Score Matching Langevin Dynamics and SDEs for Diffusion

Oscar Depp

Roadmap

Part 1. Score Matching Langevin Dynamics (SMLD)

- Why use Score Matching?
- What is a Score Function
- Langevin Dynamics

Part 2. SDEs for Diffusion

- What is an SDE
- SDE's representing Reverse and Forward diffusion
- Properties of SDEs

Part 3. Physics for Diffusion

- What is Diffusion in Physics
- Fokker-Planck Equation
- Langevin Equation

Part 1. Score Matching Langevin Dynamics (SMLD)

Score-Based Generative Models

- There are three core ingredients of Score-Based Generative Models
  1. The (Stein) score function
  2. Score matching loss functions
  3. The Langevin equation

- Will cover each in detail…

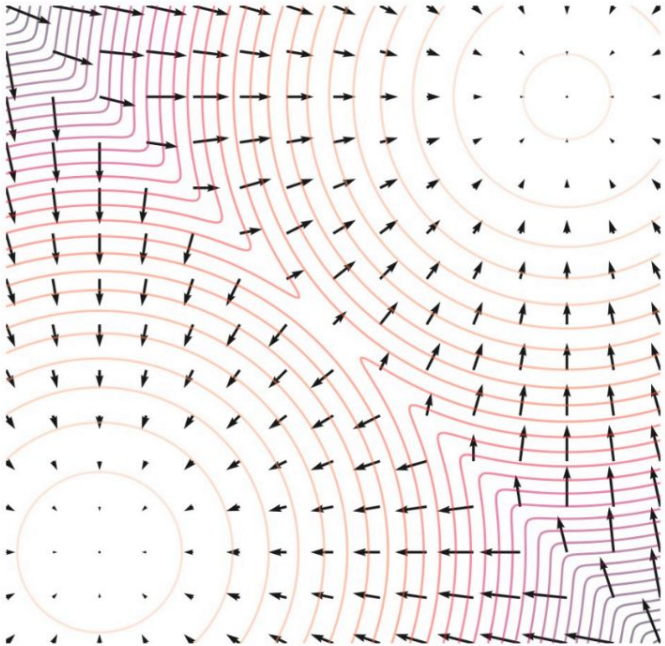
Score function

- The (Stein) score function is the gradient of the log-probability of a distribution w.r.t. to the input

$$\nabla_x \log p(x)$$

- A model $s_\theta(x)$, which models the score function explicitly, is a score-based model

$$s_\theta(x) \approx \nabla_x \log p(x)$$



The score function of a mixture of two Gaussians

https://uvadl2c.github.io/lectures/Advanced%20Generative%20&%20Energy-based%20Models/modern-based-models/lecture%204%20-%20all%20slides.pdf

Score Matching Loss Function

- Different score-based models are defined by their score matching loss functions
  - Explicit score matching (ESM) models
  - Implicit score matching (ISM) models
  - Denoising score matching (DSM) models

- Will cover each in depth

https://courses.cs.washington.edu/courses/cse599i/20au/slides/L17_score.pdf

Score Matching Loss Function

Defn (Score-Matching Loss Function): the loss functions used by a score-based model

- What is a good loss function: **MSE**

$$\mathop{\mathbb{E}}_{x\sim p}\left[\frac{1}{2}\|s_\theta(x) - \nabla_x \log p(x)\|_2^2\right]$$

- What is this quantity? It is the Fisher divergence $D_{\text{Fisher}}(p \parallel p_\theta)$

Alternatively, we are minimizing the score gradients between the two distributions? How?

What is Fisher Divergence?

Fisher Divergence intuitively measures the “distance” between two distributions

Defn: $$D_{\text{Fisher}}(p \parallel q) \equiv \mathop{\mathbb{E}}_{x\sim p}\left[\frac{1}{2}\left\|\nabla_x \log \frac{p(x)}{q(x)}\right\|^2\right]$$

$$\nabla_x \log \frac{p(x)}{q(x)} = \nabla_x \log p(x) - \nabla_x \log q(x) =: \nabla_x \log p(x) - s_\theta(x)$$

More specifically, it’s the **MSE** between **score gradients of the two distributions**!

$$D_F(p\|q_\theta) = \frac{1}{2}\mathbb{E}_{p(x)}\left[\|\nabla_x \log p(x) - s_\theta(x)\|^2\right]$$

Explicit Score Matching (ESM) Loss

Explicit Score Matching Loss Function:

$$J_{\mathrm{ESM}}(\boldsymbol{\theta}) \overset{\mathrm{def}}{=} \frac{1}{2}\mathbb{E}_{p(\mathbf{x})}\|\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}) - \nabla_{\mathbf{x}} \log p(\mathbf{x})\|^2$$

We can't access $p(x)$ → use kernel density estimation to estimate it as a plug-in estimator

$$q_h(\mathbf{x}) = \frac{1}{M}\sum_{m=1}^{M} \frac{1}{h} K\left(\frac{\mathbf{x} - \mathbf{x}^{(m)}}{h}\right),$$

What is Kernel Density Estimation?

- Kernel Density Estimation predicts response w/o knowing its distribution
  - Formula:

$$q_h(\mathbf{x}) = \frac{1}{M}\sum_{m=1}^{M}\frac{1}{h}K\left(\frac{\mathbf{x}-\mathbf{x}^{(m)}}{h}\right),$$

where K (the kernel function) is symmetric, nonnegative, and integrates to 1

- Key Idea: observations closer to $\mathbf{x}^{(m)}$ receive greater weight for q_h($\mathbf{x}^{(m)}$)
- Example: Kernel Density Estimation where K = Gaussian pdf

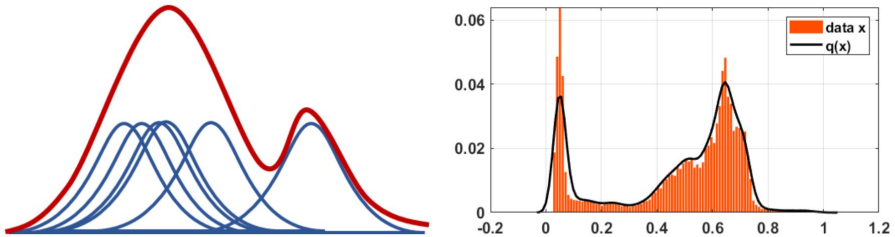


Figure 3.7: Illustration of kernel density estimation.

Why Gaussian Kernel Leads to Closed Form

Let

$$s(x) = \sum_{j=1}^{n} \alpha_j \nabla_x k(x, x_j), \quad k(x, x') = \exp\left(-\frac{\|x - x'\|^2}{2\sigma^2}\right)$$

Then, the empirical score matching loss is

$$\hat{\mathcal{L}}(\alpha) = \frac{1}{n} \sum_{i=1}^{n} \left[\frac{1}{2}\|s(x_i)\|^2 + \nabla_x \cdot s(x_i)\right]$$

Compute kernel derivatives:

$$\nabla_x k(x, x') = -\frac{1}{\sigma^2}(x - x')k(x, x'), \quad \nabla_x \cdot \nabla_x k(x, x') = \left(\frac{\|x - x'\|^2}{\sigma^4} - \frac{d}{\sigma^2}\right) k(x, x')$$

Plug into the loss:

$$\|s(x_i)\|^2 = \sum_{j,k} \alpha_j^\top \alpha_k \nabla_x k(x_i, x_j)^\top \nabla_x k(x_i, x_k)$$

$$\nabla_x \cdot s(x_i) = \sum_j \alpha_j^\top \nabla_x \cdot \nabla_x k(x_i, x_j)$$

Final closed-form:

$$\hat{\mathcal{L}}(\alpha) = \frac{1}{n} \sum_{i=1}^{n} \left[\frac{1}{2} \sum_{j,k} \alpha_j^\top \alpha_k \nabla_x k(x_i, x_j)^\top \nabla_x k(x_i, x_k) + \sum_j \alpha_j^\top \nabla_x \cdot \nabla_x k(x_i, x_j) \right]$$

Estimating Explicit Score Matching Loss Function

Given the dataset $\mathcal{X} = \{\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(M)}\}$, can estimate the loss function of ESM as:

$$
\begin{aligned}
J_{\mathrm{ESM}}(\boldsymbol{\theta}) &= \mathbb{E}_{q_h(\mathbf{x})} \|\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}) - \nabla_{\mathbf{x}} \log q_h(\mathbf{x})\|^2 \\
&= \int \|\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}) - \nabla_{\mathbf{x}} \log q_h(\mathbf{x})\|^2 q_h(\mathbf{x}) d\mathbf{x} \\
&\approx \frac{1}{M} \sum_{m=1}^{M} \int \|\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}) - \nabla_{\mathbf{x}} \log q_h(\mathbf{x})\|^2 \frac{1}{h} K\left(\frac{\mathbf{x} - \mathbf{x}^{(m)}}{h}\right) d\mathbf{x}.
\end{aligned}
$$

Implicit Score Matching (ISM)

Implicit Score Matching Loss Function:

$$J_{\mathrm{ISM}}(\boldsymbol{\theta}) \stackrel{\mathrm{def}}{=} \mathbb{E}_{p(\mathbf{x})}\left[\mathrm{Tr}(\nabla_{\mathbf{x}}\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x})) + \frac{1}{2}\|\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x})\|^2\right]$$

ISM Loss = ESM Loss + C

- It can be shown* that optimising $\mathbb{E}_{p(x)} \| \nabla_x \log p(x) - s_\theta(x) \|_2^2$ is equivalent to

$$\mathbb{E}_{p_{data}(\mathbf{x})} \left[\mathrm{tr}\left(\nabla_{\mathbf{x}} s_\theta(\mathbf{x}) \right) + \frac{1}{2} \| s_\theta(\mathbf{x}) \|_2^2 \right]$$

  up to some regularity conditions

- Still, the trace of the Jacobian is too expensive for large networks and approximations are needed

* Song et al., Sliced score matching: A scalable approach to density and score estimation, UAI 2019

https://uvadl2c.github.io/lectures/Advanced%20Generative%20&%20Energy-based%20Models/modern-based-models/lecture%204%20-%20all%20slides.pdf

Proving ISM Loss = ESM Loss + C

We want to minimize this loss function:

$$J(\theta) = \frac{1}{2}\,\mathbb{E}_{p(x)}\left\|\nabla_x \log p(x) \;-\; \nabla_x \log q_\theta(x)\right\|^2$$

Expanding out the Fisher Divergence,

$$J(\theta) = \frac{1}{2}\,\mathbb{E}_{p(x)}\left\|\nabla_x \log q_\theta(x)\right\|^2 - \mathbb{E}_{p(x)}\left[\nabla_x \log q_\theta(x) \cdot \nabla_x \log p(x)\right] + \underbrace{\tfrac{1}{2}\,\mathbb{E}_{p(x)}\|\nabla_x \log p(x)\|^2}_{\text{const w.r.t. } \theta}$$

Need to simplify,
Call this *C* in next slide

Proof (cont.): Integrating the cross term

We want to simplify the cross term, C

$$C = \mathbb{E}_{p(x)}\left[\nabla_x \log q_\theta(x) \cdot \nabla_x \log p(x)\right] = \int p(x)\, \nabla_x \log q_\theta(x) \cdot \nabla_x \log p(x)\, \mathrm{d}x$$

Note that $\nabla_x \log p(x) = \nabla_x p(x)/p(x)$

$$p(x)\, \nabla_x \log p(x) = \nabla_x p(x) \quad \Longrightarrow \quad C = \int \nabla_x p(x) \cdot \nabla_x \log q_\theta(x)\, \mathrm{d}x$$

Integrate by parts in each coordinate plane:

$$\int \frac{\partial}{\partial x_i} p(x) \left[\nabla_x \log q_\theta(x)\right]_i \mathrm{d}x = \Big[p(x) \left[\nabla_x \log q_\theta(x)\right]_i\Big]_{x_i \to \pm\infty} - \int p(x)\, \frac{\partial}{\partial x_i} \left[\nabla_x \log q_\theta(x)\right]_i \mathrm{d}x$$

Assume boundary terms cancel out

$$C = -\int p(x) \sum_{i=1}^{d} \frac{\partial}{\partial x_i} \left[\nabla_x \log q_\theta(x)\right]_i \mathrm{d}x = \boxed{-\mathbb{E}_{p(x)}\left[\nabla_x \cdot \nabla_x \log q_\theta(x)\right]}$$

Proof (cont.): Further Simplifying Cross Term

The divergence of a vector field is defined by

$$\nabla_x \cdot v(x) = \sum_{i=1}^{d} \frac{\partial v_i(x)}{\partial x_i}$$

The cross term has $v(x) = \nabla_x \log q_\theta(x)$,

So we are finding the Jacobian of the Hessian $\nabla_x^2 \log q_\theta(x)$

$$\nabla_x \cdot \nabla_x \log q_\theta(x) = \sum_{i=1}^{d} \left[\nabla_x^2 \log q_\theta(x)\right]_{ii} = \mathrm{Tr}\left[\nabla_x^2 \log q_\theta(x)\right]$$

Proof (cont.): Bringing it all together

$$J(\theta) = \frac{1}{2}\,\mathbb{E}_{p(x)}\left\|\nabla_x \log q_\theta(x)\right\|^2 - C + \text{const.}$$

$$= \frac{1}{2}\,\mathbb{E}_{p(x)}\|\nabla_x \log q_\theta(x)\|^2 + \mathbb{E}_{p(x)}\left[\nabla_x \cdot \nabla_x \log q_\theta(x)\right] + \text{const.}$$

$$= \mathbb{E}_{p(x)}\left[\tfrac{1}{2}\,\|s_\theta(x)\|^2 + \mathrm{Tr}\left[\nabla_x^2 \log q_\theta(x)\right]\right] + \text{const.} \qquad = J_{ISM}(\theta)$$

$$\boxed{J_{ESM}(\theta) = J_{ISM}(\theta) + C}$$

(Not in Stanley's notes)

This will work, in theory, but calculating the trace costs $O(d)$ calls to $s_\theta(x)$

Denoising Score Matching (DSM)

- Denoising score matching works well for small level of noise

Loss: $J_{\mathrm{DSM}}(\theta) \stackrel{\mathrm{def}}{=} \frac{1}{2}\mathbb{E}_{q_\sigma(\tilde{\mathbf{x}}|\mathbf{x})p_{data}(\mathbf{x})}\left[\left\| s_\theta(\mathbf{x}) - \nabla_{\mathbf{x}} \log q_\sigma(\mathbf{x}|\mathbf{x}') \right\|_2^2\right]$

where the data $\mathbf{x}'$ is corrupted to $\mathbf{x}$ as $q_\sigma(\mathbf{x}) = \int q_\sigma(\mathbf{x}|\mathbf{x}')p_{data}(\mathbf{x}')d\mathbf{x}'$

- First sample a training example from the training set
- Then add noise to it from a pre-specified distribution
- You can repeat the process and average with Monte Carlo simulation (or do it once)

https://uvadl2c.github.io/lectures/Advanced%20Generative%20&%20Energy-based%20Models/modern-based-models/lecture%204%20-%20all%20slides.pdf

Denoising Score Matching (cont.)

We set

$$q(\mathbf{x}|\mathbf{x}') = \mathcal{N}(\mathbf{x} \mid \mathbf{x}', \sigma^2)$$

Implies

$$\begin{aligned}
\nabla_{\mathbf{x}} \log q(\mathbf{x}|\mathbf{x}') &= \nabla_{\mathbf{x}} \log \frac{1}{(\sqrt{2\pi\sigma^2})^d} \exp\left\{-\frac{\|\mathbf{x}-\mathbf{x}'\|^2}{2\sigma^2}\right\} \\
&= \nabla_{\mathbf{x}} \left\{-\frac{\|\mathbf{x}-\mathbf{x}'\|^2}{2\sigma^2} - \log(\sqrt{2\pi\sigma^2})^d\right\} \\
&= -\frac{\mathbf{x}-\mathbf{x}'}{\sigma^2} = -\frac{\mathbf{z}}{\sigma}.
\end{aligned}$$

Therefore, loss of DSM is defined as:

$$\begin{aligned}
J_{\mathrm{DSM}}(\boldsymbol{\theta}) &\stackrel{\mathrm{def}}{=} \mathbb{E}_{q(\mathbf{x},\mathbf{x}')}\left[\frac{1}{2}\left\|\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}) - \nabla_{\mathbf{x}} \log q(\mathbf{x}|\mathbf{x}')\right\|^2\right] \\
&= \mathbb{E}_{q(\mathbf{x}')}\left[\frac{1}{2}\left\|\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}' + \sigma\mathbf{z}) + \frac{\mathbf{z}}{\sigma}\right\|^2\right].
\end{aligned}$$

Equivalence of Loss for Score Matching Methods

Before proof, we need to unify the notations:

- Data distribution $p(\mathbf{x})$ on $\mathbb{R}^d$
- Learnable score function $s_\theta(\mathbf{x}) = \nabla_{\mathbf{x}} \log q_\theta(\mathbf{x})$ approximating $\nabla_{\mathbf{x}} \log p(\mathbf{x})$
- Noising process $q(\mathbf{x}'|\mathbf{x})$ with joint distribution $q(\mathbf{x}, \mathbf{x}') = p(\mathbf{x})q(\mathbf{x}'|\mathbf{x})$
- Functions are differentiable with vanishing boundary terms

Loss Functions

Denoising Score Matching (DSM)

$$J_{\text{DSM}}(\theta) \triangleq \mathbb{E}_{q(\mathbf{x},\mathbf{x}')}\left[\frac{1}{2}\|s_\theta(\mathbf{x}) - \nabla_{\mathbf{x}} \log q(\mathbf{x}|\mathbf{x}')\|^2\right]$$

Explicit Score Matching (ESM)

$$J_{\text{ESM}}(\theta) \triangleq \mathbb{E}_{p(\mathbf{x})}\left[\frac{1}{2}\|s_\theta(\mathbf{x}) - \nabla_{\mathbf{x}} \log p(\mathbf{x})\|^2\right]$$

Implicit Score Matching (ISM)

$$J_{\text{ISM}}(\theta) \triangleq \mathbb{E}_{p(\mathbf{x})}\left[\frac{1}{2}\|s_\theta(\mathbf{x})\|^2 + \text{Tr}[\nabla_{\mathbf{x}}^2 \log q_\theta(\mathbf{x})]\right]$$

Theorem 1: ESM-ISM Equivalence

$$J_{\mathrm{ESM}}(\theta) = J_{\mathrm{ISM}}(\theta) + C \quad \text{where } C \text{ is independent of } \theta$$

Proof of ESM-ISM Equivalence (1/3)

Start with ESM objective and expand:

$$J_{\text{ESM}}(\theta) = \frac{1}{2}\mathbb{E}_{p(\mathbf{x})}\left[\|s_\theta(\mathbf{x}) - \nabla_{\mathbf{x}} \log p(\mathbf{x})\|^2\right]$$
$$= \frac{1}{2}\mathbb{E}_{p(\mathbf{x})}\left[\|s_\theta(\mathbf{x})\|^2 - 2s_\theta(\mathbf{x})^T \nabla_{\mathbf{x}} \log p(\mathbf{x}) + \|\nabla_{\mathbf{x}} \log p(\mathbf{x})\|^2\right]$$

Last term is constant w.r.t. θ (C_1). Focus on cross term:

$$C = -\mathbb{E}_{p(\mathbf{x})}\left[s_\theta(\mathbf{x})^T \nabla_{\mathbf{x}} \log p(\mathbf{x})\right]$$

Proof of ESM-ISM Equivalence (2/3)

Using $\nabla_{\mathbf{x}} \log p(\mathbf{x}) = \frac{\nabla_{\mathbf{x}} p(\mathbf{x})}{p(\mathbf{x})}$:

$$C = -\int s_\theta(\mathbf{x})^T \nabla_{\mathbf{x}} p(\mathbf{x}) d\mathbf{x}$$

Applying integration by parts:

$$\begin{aligned} C &= \int p(\mathbf{x}) \sum_{i=1}^{d} \frac{\partial [s_\theta(\mathbf{x})]_i}{\partial x_i} d\mathbf{x} \\ &= \mathbb{E}_{p(\mathbf{x})} \left[\nabla_{\mathbf{x}} \cdot s_\theta(\mathbf{x}) \right] \end{aligned}$$

Write it in matrix trace:

$$C = \mathbb{E}_{p(\mathbf{x})} \left[\mathrm{Tr}[\nabla_{\mathbf{x}}^2 \log q_\theta(\mathbf{x})] \right]$$

Proof of ESM-ISM Equivalence (3/3)

Substituting back:

$$J_{\text{ESM}}(\theta) = \frac{1}{2}\mathbb{E}_{p(\mathbf{x})}\left[\|s_\theta(\mathbf{x})\|^2\right] + \mathbb{E}_{p(\mathbf{x})}\left[\text{Tr}[\nabla_{\mathbf{x}}^2 \log q_\theta(\mathbf{x})]\right] + C_1$$

ISM objective:

$$J_{\text{ISM}}(\theta) = \mathbb{E}_{p(\mathbf{x})}\left[\frac{1}{2}\|s_\theta(\mathbf{x})\|^2 + \text{Tr}[\nabla_{\mathbf{x}}^2 \log q_\theta(\mathbf{x})]\right]$$

Combining:

$$J_{\text{ESM}}(\theta) = J_{\text{ISM}}(\theta) + C_1$$

where C_1 is independent of θ.

Theorem 2: ESM-DSM Equivalence

$$J_{\mathrm{DSM}}(\theta) = J_{\mathrm{ESM}}(\theta) + C \quad \text{where } C \text{ is independent of } \theta$$

Proof of ESM-DSM Equivalence (1/3)

Expand ESM:

$$J_{\text{ESM}}(\theta) = \mathbb{E}_{p(\mathbf{x})}\left[\frac{1}{2}\|s_\theta(\mathbf{x})\|^2 - s_\theta(\mathbf{x})^T \nabla_{\mathbf{x}} \log p(\mathbf{x}) + \frac{1}{2}\|\nabla_{\mathbf{x}} \log p(\mathbf{x})\|^2\right]$$

Last term is constant (C_1). For cross term, use conditioning:

$$\mathbb{E}_{p(\mathbf{x})}[s_\theta(\mathbf{x})^T \nabla_{\mathbf{x}} \log p(\mathbf{x})] = \int s_\theta(\mathbf{x})^T \nabla_{\mathbf{x}} \left(\int p(\mathbf{x}')q(\mathbf{x}|\mathbf{x}')d\mathbf{x}'\right) d\mathbf{x}$$

Proof of ESM-DSM Equivalence (2/3)

Moving gradient inside:

$$= \int s_\theta(\mathbf{x})^T \left(\int p(\mathbf{x}') \nabla_\mathbf{x} q(\mathbf{x}|\mathbf{x}') d\mathbf{x}' \right) d\mathbf{x}$$

$$= \int s_\theta(\mathbf{x})^T \int q(\mathbf{x}') \left(\frac{\nabla_\mathbf{x} q(\mathbf{x}|\mathbf{x}')}{q(\mathbf{x}|\mathbf{x}')} \right) q(\mathbf{x}|\mathbf{x}') d\mathbf{x}' d\mathbf{x}$$

Using $\frac{\nabla_\mathbf{x} q(\mathbf{x}|\mathbf{x}')}{q(\mathbf{x}|\mathbf{x}')} = \nabla_\mathbf{x} \log q(\mathbf{x}|\mathbf{x}')$:

$$= \int \int q(\mathbf{x}, \mathbf{x}') (s_\theta(\mathbf{x})^T \nabla_\mathbf{x} \log q(\mathbf{x}|\mathbf{x}')) d\mathbf{x}' d\mathbf{x}$$

$$= \mathbb{E}_{q(\mathbf{x},\mathbf{x}')} [s_\theta(\mathbf{x})^T \nabla_\mathbf{x} \log q(\mathbf{x}|\mathbf{x}')]$$

Proof of ESM-DSM Equivalence (3/3)

Therefore:

$$J_{\text{ESM}}(\theta) = \mathbb{E}_{p(\mathbf{x})}\left[\frac{1}{2}\|s_\theta(\mathbf{x})\|^2\right] - \mathbb{E}_{q(\mathbf{x},\mathbf{x}')}[s_\theta(\mathbf{x})^T \nabla_{\mathbf{x}} \log q(\mathbf{x}|\mathbf{x}')] + C_1$$

Expand DSM:

$$J_{\text{DSM}}(\theta) = \mathbb{E}_{q(\mathbf{x},\mathbf{x}')}\left[\frac{1}{2}\|s_\theta(\mathbf{x})\|^2 - s_\theta(\mathbf{x})^T \nabla_{\mathbf{x}} \log q(\mathbf{x}|\mathbf{x}') + \frac{1}{2}\|\nabla_{\mathbf{x}} \log q(\mathbf{x}|\mathbf{x}')\right.$$

Last term is constant (C_2). Since $\mathbf{x}$ marginal is $p(\mathbf{x})$:

$$\begin{aligned} J_{\text{DSM}}(\theta) &= \mathbb{E}_{p(\mathbf{x})}\left[\frac{1}{2}\|s_\theta(\mathbf{x})\|^2\right] - \mathbb{E}_{q(\mathbf{x},\mathbf{x}')}[s_\theta(\mathbf{x})^T \nabla_{\mathbf{x}} \log q(\mathbf{x}|\mathbf{x}')] + C_2 \\ &= J_{\text{ESM}}(\theta) - C_1 + C_2 \\ &= J_{\text{ESM}}(\theta) + C \end{aligned}$$

where $C = C_2 - C_1$ is independent of θ.

Conclusion

- All three score matching methods (ESM, ISM, DSM) are equivalent up to constants
- The result is on population distribution level
- For optimization purposes, they lead to the same optimal score function

Denoising SM objective

Vincent, 2011

$$\arg\min_{\theta} \text{Fisher}(p_\sigma, q_\theta) = \arg\min_{\theta} \mathbb{E}\left[\|s_\theta(\tilde{\mathbf{x}}) - \nabla_{\mathbf{x}} \log p_\sigma(\tilde{\mathbf{x}} \mid \mathbf{x})\|^2\right].$$

Proof:

$$\begin{aligned}\text{Fisher} &= \mathbb{E}_{\tilde{\mathbf{x}}\sim p_\sigma}\left[\frac{1}{2}\|s_\theta(\tilde{\mathbf{x}}) - \nabla_{\tilde{\mathbf{x}}} \log p_\sigma(\tilde{\mathbf{x}})\|_2^2\right] \\ &= \mathbb{E}_{\tilde{\mathbf{x}}\sim p_\sigma}\left[\frac{1}{2}\|s_\theta(\tilde{\mathbf{x}})\|^2 - s_\theta(\tilde{\mathbf{x}})^\top \nabla_{\tilde{\mathbf{x}}} \log p_\sigma(\tilde{\mathbf{x}}) + \text{Constant}\right].\end{aligned}$$

Two Applications of the Log-Derivative Trick:

$$\mathbb{E}_{\tilde{\mathbf{x}}\sim p_\sigma}\left[s_\theta(\tilde{\mathbf{x}})^\top \nabla_{\tilde{\mathbf{x}}} \log p_\sigma(\tilde{\mathbf{x}})\right] = \int s_\theta(\tilde{\mathbf{x}})^\top \nabla_{\tilde{\mathbf{x}}} p_\sigma(\tilde{\mathbf{x}}) d\tilde{\mathbf{x}} = \int s_\theta(\tilde{\mathbf{x}})^\top \nabla_{\tilde{\mathbf{x}}} p_{\text{data}}(\mathbf{x}) p_\sigma(\tilde{\mathbf{x}} \mid \mathbf{x}) d\mathbf{x} d\tilde{\mathbf{x}}.$$

Final Simplification:

$$= \mathbb{E}_{\mathbf{x}\sim p_{\text{data}}} \mathbb{E}_{\tilde{\mathbf{x}}\sim p_\sigma(\cdot\mid\mathbf{x})}\left[s_\theta(\tilde{\mathbf{x}})^\top \nabla_{\tilde{\mathbf{x}}} \log p_\sigma(\tilde{\mathbf{x}} \mid \mathbf{x})\right].$$

DSM Loss = ESM Loss + Constant

- Proposition [Vincent, 2011]:

$$\arg\min_\theta D_{\text{Fisher}}(q_\sigma \parallel p_\theta) = \arg\min_\theta \mathop{\mathbb{E}}_{\substack{x\sim p \\ \tilde{x}\sim p_\sigma(\cdot|x)}} \left[\frac{1}{2}\|s_\theta(\tilde{x}) - \nabla_{\tilde{x}} \log p_\sigma(\tilde{x}|x)\|_2^2\right].$$

Needs some work

- Proof: Expand the quadratic:

$$\arg\min_\theta \mathop{\mathbb{E}}_{\tilde{x}\sim q_\sigma} \left[\frac{1}{2}\|s_\theta(\tilde{x}) - \nabla_{\tilde{x}} \log q_\sigma(\tilde{x})\|_2^2\right] = \arg\min_\theta \mathop{\mathbb{E}}_{\tilde{x}\sim q_\sigma} \left[\frac{1}{2}\|s_\theta(\tilde{x})\|^2 - s_\theta(\tilde{x})^T \nabla_{\tilde{x}} \log q_\sigma(\tilde{x})\right].$$

Two applications of the log-derivative trick:

$$\begin{aligned}
\mathop{\mathbb{E}}_{\tilde{x}\sim q_\sigma} \left[s_\theta(\tilde{x})^T \nabla_{\tilde{x}} \log q_\sigma(\tilde{x})\right] &= \int_{\mathcal{X}} s_\theta(\tilde{x})^T \nabla_{\tilde{x}} \log q_\sigma(\tilde{x}) q_\sigma(\tilde{x})\, d\tilde{x} = \int_{\mathcal{X}} s_\theta(\tilde{x})^T \nabla_{\tilde{x}} q_\sigma(\tilde{x})\, d\tilde{x} \\
&= \int_{\mathcal{X}} s_\theta(\tilde{x})^T \nabla_{\tilde{x}} \int_{\mathcal{X}} p(x) p_\sigma(\tilde{x}|x)\, dx\, d\tilde{x} = \int_{\mathcal{X}} s_\theta(\tilde{x})^T \int_{\mathcal{X}} p(x) p_\sigma(\tilde{x}|x) \nabla_{\tilde{x}} \log p_\sigma(\tilde{x}|x)\, dx\, d\tilde{x} \\
&= \iint_{\mathcal{X}\times\mathcal{X}} p(x) p_\sigma(\tilde{x}|x) s_\theta(\tilde{x})^T \nabla_{\tilde{x}} \log p_\sigma(\tilde{x}|x)\, d(x, \tilde{x}) = \mathop{\mathbb{E}}_{\substack{x\sim p \\ \tilde{x}\sim p_\sigma(\tilde{x}|x)}} \left[s_\theta(\tilde{x})^T \nabla_{\tilde{x}} \log p_\sigma(\tilde{x}|x)\right].
\end{aligned}$$

DSM Loss = ESM Loss + Constant

Theorem 3.4. [Vincent [43]] For up to a constant C which is independent of the variable $\boldsymbol{\theta}$, it holds that

$$J_{\mathrm{DSM}}(\boldsymbol{\theta}) = J_{\mathrm{ESM}}(\boldsymbol{\theta}) + C. \tag{3.12}$$

Proof in following slides

Proof of Equivalence

Start with loss of ESM

$$
\begin{aligned}
J_{\mathrm{ESM}}(\boldsymbol{\theta}) &= \mathbb{E}_{q(\mathbf{x})}\left[\frac{1}{2}\left\|\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}) - \nabla_{\mathbf{x}} \log q(\mathbf{x})\right\|^2\right] \\
&= \mathbb{E}_{q(\mathbf{x})}\Big[\frac{1}{2}\left\|\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x})\right\|^2 - \mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x})^T \nabla_{\mathbf{x}} \log q(\mathbf{x}) + \underbrace{\frac{1}{2}\left\|\nabla_{\mathbf{x}} \log q(\mathbf{x})\right\|^2}_{\stackrel{\text{def}}{=} C_1, \text{independent of } \boldsymbol{\theta}}\Big].
\end{aligned}
$$

Discard C1, focus on the second term:

$$
\begin{aligned}
\mathbb{E}_{q(\mathbf{x})}\left[\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x})^T \nabla_{\mathbf{x}} \log q(\mathbf{x})\right] &= \int \left(\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x})^T \nabla_{\mathbf{x}} \log q(\mathbf{x})\right) q(\mathbf{x}) d\mathbf{x}, && \text{(expectation)} \\
&= \int \left(\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x})^T \frac{\nabla_{\mathbf{x}} q(\mathbf{x})}{\cancel{q(\mathbf{x})}}\right) \cancel{q(\mathbf{x})} d\mathbf{x}, && \text{(gradient)} \\
&= \int \mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x})^T \nabla_{\mathbf{x}} q(\mathbf{x}) d\mathbf{x}.
\end{aligned}
$$

To connect to DSM, we need the conditional distribution q(x|x')

$$\int \mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x})^T \nabla_{\mathbf{x}} q(\mathbf{x}) d\mathbf{x} = \int \mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x})^T \nabla_{\mathbf{x}} \underbrace{\left(\int q(\mathbf{x}') q(\mathbf{x}|\mathbf{x}') d\mathbf{x}' \right)}_{=q(\mathbf{x})} d\mathbf{x} \qquad \text{(conditional)}$$

$$= \int \mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x})^T \left(\int q(\mathbf{x}') \nabla_{\mathbf{x}} q(\mathbf{x}|\mathbf{x}') d\mathbf{x}' \right) d\mathbf{x} \qquad \text{(move gradient)}$$

$$= \int \mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x})^T \left(\int q(\mathbf{x}') \nabla_{\mathbf{x}} q(\mathbf{x}|\mathbf{x}') \times \frac{q(\mathbf{x}|\mathbf{x}')}{q(\mathbf{x}|\mathbf{x}')} d\mathbf{x}' \right) d\mathbf{x} \qquad \text{(multiple and divide)}$$

Convert to log q(x|x'), pretty genius

$$= \int \mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x})^T \int q(\mathbf{x}') \underbrace{\left(\frac{\nabla_{\mathbf{x}} q(\mathbf{x}|\mathbf{x}')}{q(\mathbf{x}|\mathbf{x}')} \right)}_{=\nabla_{\mathbf{x}} \log q(\mathbf{x}|\mathbf{x}')} q(\mathbf{x}|\mathbf{x}') d\mathbf{x}' d\mathbf{x} \qquad \text{(rearrange terms)}$$

$$= \int \mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x})^T \left(\int q(\mathbf{x}') \left(\nabla_{\mathbf{x}} \log q(\mathbf{x}|\mathbf{x}') \right) q(\mathbf{x}|\mathbf{x}') d\mathbf{x}' \right) d\mathbf{x}$$

$$= \int \int \underbrace{q(\mathbf{x}|\mathbf{x}') q(\mathbf{x}')}_{=q(\mathbf{x},\mathbf{x}')} \left(\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x})^T \nabla_{\mathbf{x}} \log q(\mathbf{x}|\mathbf{x}') \right) d\mathbf{x}' d\mathbf{x} \qquad \text{(move integration)}$$

Expectation is taken over x and x'

$$= \mathbb{E}_{q(\mathbf{x},\mathbf{x}')} \left[\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x})^T \nabla_{\mathbf{x}} \log q(\mathbf{x}|\mathbf{x}') \right].$$

Apply it to J_ESM,

$$J_{\mathrm{ESM}}(\boldsymbol{\theta}) = \mathbb{E}_{q(\mathbf{x})}\left[\frac{1}{2}\left\|\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x})\right\|^2\right] - \mathbb{E}_{q(\mathbf{x},\mathbf{x}')}\left[\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x})^T \nabla_{\mathbf{x}} \log q(\mathbf{x}|\mathbf{x}')\right] + C_1.$$

Compare to J_DSM

$$\begin{aligned} J_{\mathrm{DSM}}(\boldsymbol{\theta}) &\stackrel{\mathrm{def}}{=} \mathbb{E}_{q(\mathbf{x},\mathbf{x}')}\left[\frac{1}{2}\left\|\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}) - \nabla_{\mathbf{x}} q(\mathbf{x}|\mathbf{x}')\right\|^2\right] \\ &= \mathbb{E}_{q(\mathbf{x},\mathbf{x}')}\Big[\frac{1}{2}\left\|\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x})\right\|^2 - \mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x})^T \nabla_{\mathbf{x}} \log q(\mathbf{x}|\mathbf{x}') + \underbrace{\frac{1}{2}\left\|\nabla_{\mathbf{x}} \log q(\mathbf{x}|\mathbf{x}')\right\|^2}_{\stackrel{\mathrm{def}}{=} C_2, \text{independent of } \boldsymbol{\theta}}\Big] \\ &= \mathbb{E}_{q(\mathbf{x})}\left[\frac{1}{2}\left\|\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x})\right\|^2\right] - \mathbb{E}_{q(\mathbf{x},\mathbf{x}')}\left[\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x})^T \nabla_{\mathbf{x}} \log q(\mathbf{x}|\mathbf{x}')\right] + C_2. \end{aligned}$$

End of Proof.

This means that ESM, ISM, DSM all have same loss function up to a constant

Then why prefer DSM loss function?

Apply Loss in Real

Loss of DSM is take over q(x,x'), not easy to implement.

Need extra assumption: $q(\mathbf{x}|\mathbf{x}') = \mathcal{N}(\mathbf{x} \mid \mathbf{x}', \sigma^2) \quad \mathbf{x} = \mathbf{x}' + \sigma\mathbf{z}.$

Then gradient of log q(x|x') can be simplified as

$$\begin{aligned}
\nabla_{\mathbf{x}} \log q(\mathbf{x}|\mathbf{x}') &= \nabla_{\mathbf{x}} \log \frac{1}{(\sqrt{2\pi\sigma^2})^d} \exp\left\{-\frac{\|\mathbf{x}-\mathbf{x}'\|^2}{2\sigma^2}\right\} \\
&= \nabla_{\mathbf{x}} \left\{-\frac{\|\mathbf{x}-\mathbf{x}'\|^2}{2\sigma^2} - \log(\sqrt{2\pi\sigma^2})^d\right\} \\
&= -\frac{\mathbf{x}-\mathbf{x}'}{\sigma^2} = -\frac{\mathbf{z}}{\sigma}.
\end{aligned}$$

Thus the loss of DSM now is:

$$J_{\mathrm{DSM}}(\boldsymbol{\theta}) \stackrel{\mathrm{def}}{=} \mathbb{E}_{q(\mathbf{x},\mathbf{x}')}\left[\frac{1}{2}\left\|\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}) - \nabla_{\mathbf{x}} \log q(\mathbf{x}|\mathbf{x}')\right\|^2\right]$$
$$= \mathbb{E}_{q(\mathbf{x}')}\left[\frac{1}{2}\left\|\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}' + \sigma\mathbf{z}) + \frac{\mathbf{z}}{\sigma}\right\|^2\right].$$

Remark: Expectation over x can be taken off

- z and x' are independent
- z is regarded as non-random.

Theorem 3.3. The **Denoising Score Matching** has a loss function defined as

$$J_{\mathrm{DSM}}(\boldsymbol{\theta}) = \mathbb{E}_{p(\mathbf{x})}\left[\frac{1}{2}\left\|\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x} + \sigma\mathbf{z}) + \frac{\mathbf{z}}{\sigma}\right\|^2\right] \tag{3.11}$$

Advantages of DSM's

Recall DSM Loss definition:

$$J_{\mathrm{DSM}}(\boldsymbol{\theta}) = \mathbb{E}_{p(\mathbf{x})}\left[\frac{1}{2}\left\|\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}+\sigma\mathbf{z}) + \frac{\mathbf{z}}{\sigma}\right\|^2\right]$$

Advantages of DSM Loss:

1. Highly interpretable:
   - During training, score function learns to predict the noise of noisy image
   - Predicting noise = denoising
2. Doesn't require trace computations (as we do for ISM)
3. Doesn't have kernel density estimation's drawbacks (e.g. curse of dimensionality)

Challenges in Denoising Score Matching

- score function fit is less accurate over low density regions of $p(x)$ since we observe few samples
- increasing additive noise variance helps estimating a better score function, however we learn a noisier perturbed distribution
- sampling can get stuck at isolated modes
- **idea:** use multiple scales of noise (Song and Ermon, Generative Modeling by Estimating Gradients of the Data Distribution, 2019)

https://web.stanford.edu/class/ee364b/lectures/diffusion_models.pdf

Score matching: advantages

- We have no constraints on the form of $f_\theta(x)$ as we do not require $s_\theta(x)$ to be the score function of a normalised distribution
- We just compare our neural network output with the ground-truth data score
- The only requirement is that $s_\theta(x)$ is a vector valued function with the same input and output dimensionality

https://uvadl2c.github.io/lectures/Advanced%20Generative%20&%20Energy-based%20Models/modern-based-models/lecture%204%20-%20all%20slides.pdf

Sampling using Langevin dynamics

Inference

- During training we do not involve an explicit "sampling" mechanism
- After training the score-based model, we can sample with Langevin dynamics
- Langevin dynamics are an MCMC procedure to sample from distribution $p(x)$ using only the score function $\nabla_x \log p(x)$

$$\mathbf{x}_{t+1} \leftarrow \mathbf{x}_t + \epsilon \nabla_{\mathbf{x}} \log p(\mathbf{x}_t) + \sqrt{2\epsilon}\mathbf{z}_t, \quad t = 0, \ldots, K, \quad \mathbf{z}_t \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$$

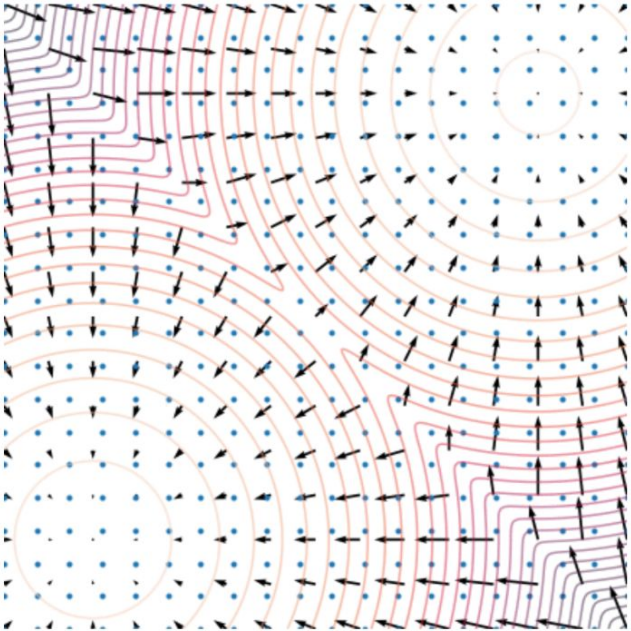
- Where for $t = 0$ we sample from an arbitrary prior distribution $\mathbf{x}_0 \sim \pi(\mathbf{x})$
- And is a sample from a standard Gaussian

https://uvadl2c.github.io/lectures/Advanced%20Generative%20&%20Energy-based%20Models/modern-based-models/lecture%204%20-%20all%20slides.pdf

Sampling using Langevin dynamics

$$\mathbf{x}_{t+1} \leftarrow \mathbf{x}_t + \epsilon \nabla_{\mathbf{x}} \log p(\mathbf{x}_t) + \sqrt{2\epsilon}\mathbf{z}_t, t = 0, \ldots, K$$

- For $\epsilon \to 0$ and $K \to \infty$ we sample from $p(\mathbf{x})$ (under conditions)
- Importantly, this is an iterative sampling procedure for which we only need to score function
- So, we can produce samples by iteratively computing $\mathbf{x}_{t+1}$ via score function $s_\theta(x) \approx \nabla_x \log p(x)$



https://uvadl2c.github.io/lectures/Advanced%20Generative%20&%20Energy-based%20Models/modern-based-models/lecture%204%20-%20all%20slides.pdf

Langevin Dynamics

$$\mathbf{x}_{t+1} \leftarrow \mathbf{x}_t + \epsilon \nabla_{\mathbf{x}} \log p(\mathbf{x}_t) + \sqrt{2\epsilon}\mathbf{z}_t, \quad t = 0, \ldots, K, \quad \mathbf{z}_t \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$$

- Originally developed to model molecular dynamics
- You can think of Langevin dynamics as something similar to stochastic gradient descent, only now we do not necessarily optimise for parameters
- Given your current position $\mathbf{x}_t$ we move to the direction of the gradient ∇ of the score function (log-likehood function) $\log p(\mathbf{x}_t)$, corrupted with some noise $\mathbf{z}_t$, scaled by ϵ (like 'learning rate') annealed over time

https://uvadl2c.github.io/lectures/Advanced%20Generative%20&%20Energy-based%20Models/modern-based-models/lecture%204%20-%20all%20slides.pdf

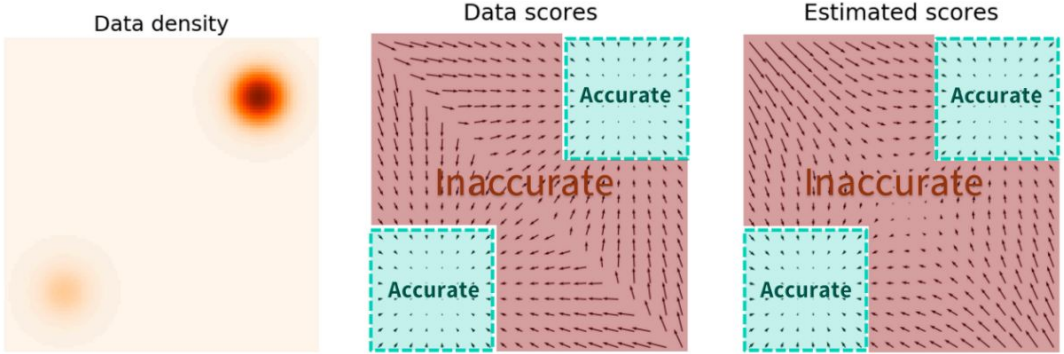
Low data density regions

Breaking down the Langevin dynamics formula

- Minimising Fisher divergence means placing more emphasis where $p(\mathbf{x})$ is high

$$\mathbb{E}_{p(\mathbf{x})}\left[\|\nabla_x \log p(\mathbf{x}) - s_\theta(\mathbf{x})\|_2^2\right] = \int p(\mathbf{x})\|\nabla_\mathbf{x} \log p(\mathbf{x}) - s_\theta(\mathbf{x})\|_2^2 d\mathbf{x}$$

- Even harder in high-dimensional spaces that are mostly empty
- The Monte Carlo sample estimates will not be accurate enough



Slow mixing of Langevin dynamics

$$\mathbf{x}_{t+1} \leftarrow \mathbf{x}_t + \epsilon \nabla_{\mathbf{x}} \log p(\mathbf{x}_t) + \sqrt{2\epsilon}\mathbf{z}_t, \quad t = 0, \ldots, K, \quad \mathbf{z}_t \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$$

- When the true density has two (or multiple) modes separated by a low-density region, it is hard for Langevin dynamics to visit them in a reasonable time
- That makes sense: the 'jumps' local around current location of score function and the added noise is unlikely to be large enough to push to far

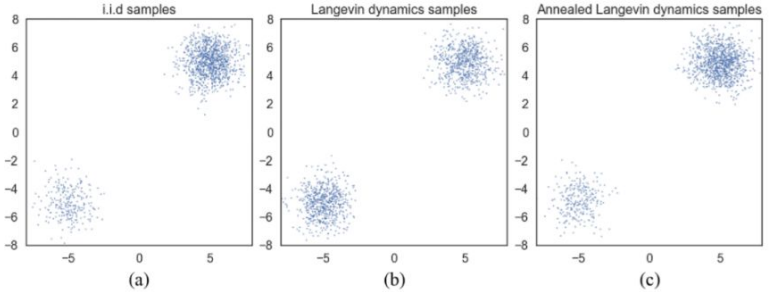
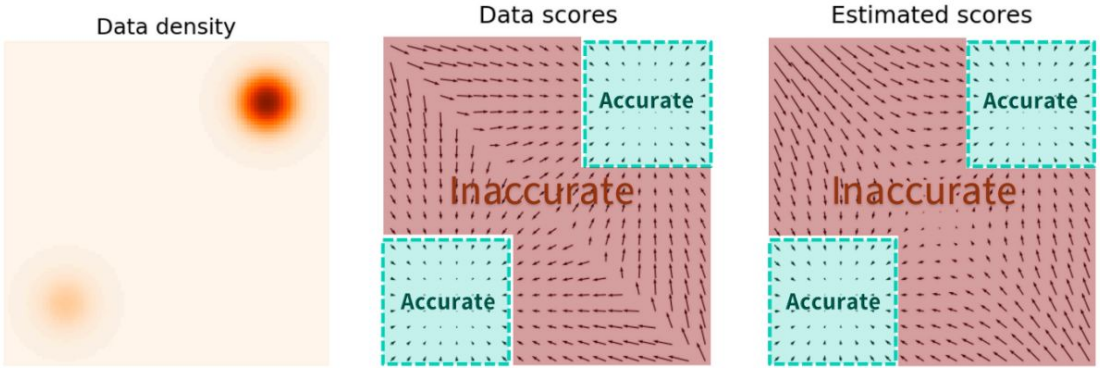


Figure 3: Samples from a mixture of Gaussian with different methods. (a) Exact sampling. (b) Sampling using Langevin dynamics with the exact scores. (c) Sampling using annealed Langevin dynamics with the exact scores. Clearly Langevin dynamics estimate the relative weights between the two modes incorrectly, while annealed Langevin dynamics recover the relative weights faithfully.

From 'Generative Modelling by Estimating Gradients of the Data Distribution', by Song and Ermon

Naive score-based ignores low-density regions

- In the naive case of training score-based methods we have innaccurate score function estimation
- And we have slow mixing of Langevin dynamics
- As a result, the Langevin chain will start from a low density region and get stuck

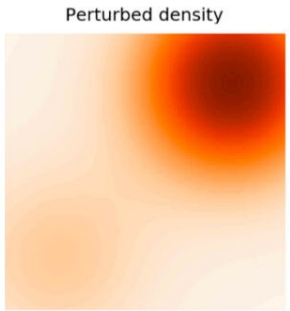


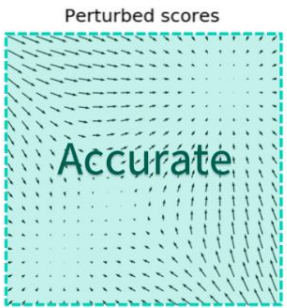
Noise perturbations

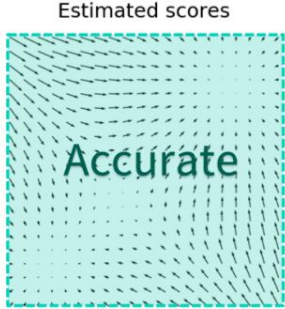
- Perturb data with noise ← Noised up data fill up the "empty" space
- Too much noise will over-corrupt the data, however, so caution is needed
- Add noise from $\mathcal{N}(0,\sigma_t)$ with more and more variance: $\sigma_1 < \sigma_2 < \ldots < \sigma_L$, specifically by marginalising out the noise variable

$$p_{\sigma_t}(\mathbf{x}) = \int p(\mathbf{x},\mathbf{y})\mathrm{d}\mathbf{y} = \int p(\mathbf{y})p(\mathbf{x}\,|\,\mathbf{y})\mathrm{d}\mathbf{y}$$

$$= \int p(\mathbf{y})\mathcal{N}(\mathbf{x}\,|\,\mathbf{y},\sigma_t^2 I)\mathrm{d}\mathbf{y}$$

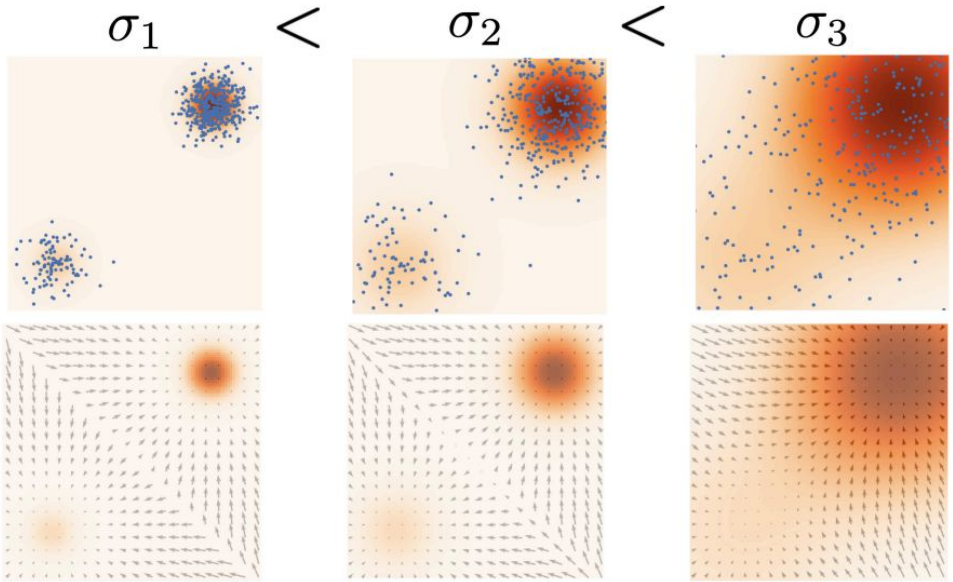






Noise-conditional Score-based Models

- Learn the score-matching function on the perturbed data points



Multiple scales of Gaussian noise to perturb data (above) so that to learn the respective score-matching function (below).

Noise-Conditional Score-based Models

- The final objective is a weighted sum of Fisher divergences

$$\sum_t \lambda(t)\mathbb{E}_{p_{\sigma_t}(\mathbf{x})}\left[\|\nabla_{\mathbf{x}}\log p_{\sigma_t}(\mathbf{x}) - s_\theta(\mathbf{x}, t])\|\right]$$

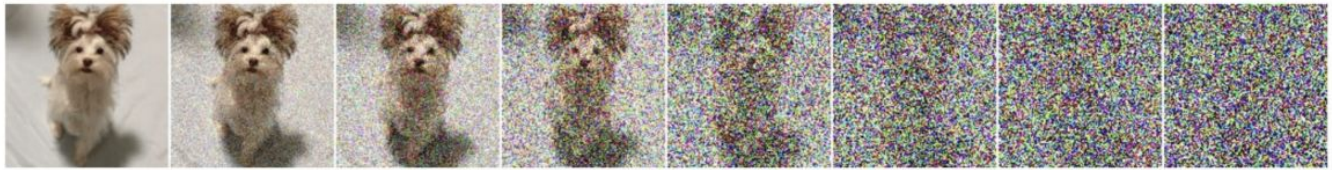
where $\lambda(t)$ is a weighting function, typical choice $\lambda(t) = \sigma_t^2$

Noising-up real images

Multiple scales of noise

- we perturb data by adding Gaussian noise of standard deviation $\sigma_1, \sigma_2, ..., \sigma_L$ such that $\sigma_1 \leq \sigma_2 \leq ... \leq \sigma_L$

  i.e, given a data sample x, we generate $x + \mathcal{N}(0, \sigma_1 I)$, $x + \mathcal{N}(0, \sigma_2 I)$, ... , $x + \mathcal{N}(0, \sigma_L I)$



- we fit a score function model $s_\theta(x, \sigma)$ which is also a function of the noise level σ

Fitting a noise conditional score function

- we minimize a weighted combination of denoising score matching losses over L noise scales

$$\min_{\theta} \sum_{i=1}^{L} \lambda_i \mathbb{E}_{x \sim p(x)} \mathbb{E}_{n \sim \mathcal{N}(0,I)} \left\| s_\theta(x + \sigma_i n, \sigma_i) - \frac{n}{\sigma_i} \right\|_2^2$$

- we can pick weights proportional to variance, $\lambda_i = \sigma_i^2$

$$\min_{\theta} \sum_{i=1}^{L} \sigma_i \mathbb{E}_{x \sim p(x)} \mathbb{E}_{n \sim \mathcal{N}(0,I)} \left\| s_\theta(x + \sigma_i n, \sigma_i) - \frac{n}{\sigma_i} \right\|_2^2$$

which simplifies to

$$\min_{\theta} \sum_{i=1}^{L} \mathbb{E}_{x \sim p(x)} \mathbb{E}_{n \sim \mathcal{N}(0,I)} \left\| \sigma_i s_\theta(x + \sigma_i n, \sigma_i) - n \right\|_2^2$$

https://web.stanford.edu/class/ee364b/lectures/diffusion_models.pdf

Generative Modeling by Estimating Gradients of the Data Distribution Y. Song, S. Ermon. NeurIPS 2019

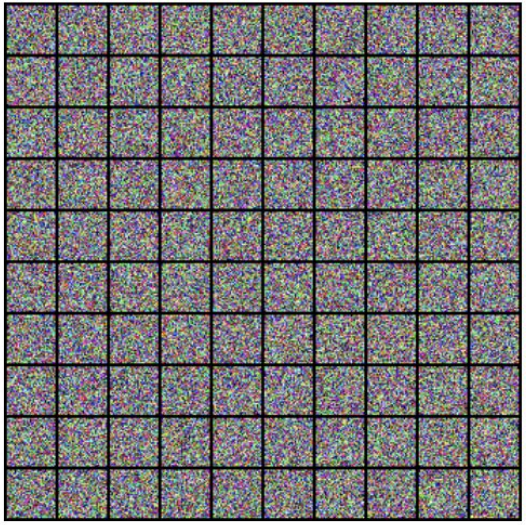
Annealed Langevin Dynamics

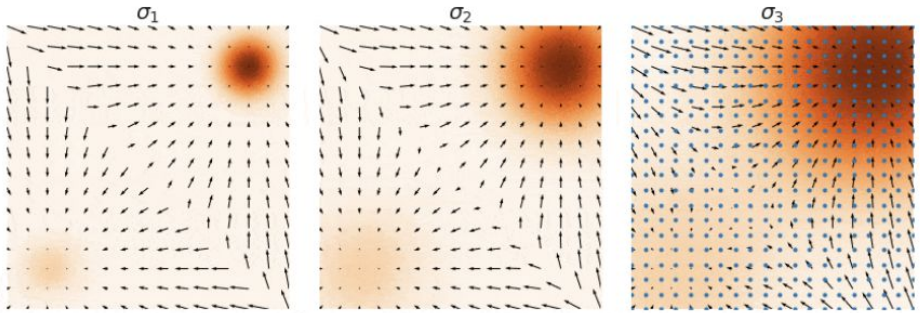
- Like before, but we start sampling from larger noise, which we gradually decrease

Algorithm 1 Annealed Langevin dynamics.

Require: $\{\sigma_i\}_{i=1}^{L}, \epsilon, T$.

Initialize $\tilde{\mathbf{x}}_0$
for $i \leftarrow 1$ to L **do**
  $\alpha_i \leftarrow \epsilon \cdot \sigma_i^2/\sigma_L^2$ ▷ α_i is the step size.
  for $t \leftarrow 1$ to T **do**
    Draw $\mathbf{z}_t \sim \mathcal{N}(0, I)$
    $\tilde{\mathbf{x}}_t \leftarrow \tilde{\mathbf{x}}_{t-1} + \frac{\alpha_i}{2}\mathbf{s}_{\boldsymbol{\theta}}(\tilde{\mathbf{x}}_{t-1}, \sigma_i) + \sqrt{\alpha_i}\,\mathbf{z}_t$
  end for
  $\tilde{\mathbf{x}}_0 \leftarrow \tilde{\mathbf{x}}_T$
end for
return $\tilde{\mathbf{x}}_T$





Practical tips

- Pick σ_t in geometric progression where σ_L is comparable to max distance between samples in the training set
- L is typical in the order of hundreds or thousands

Part 2. SDE’s for Diffusion

Score-based models with SDEs

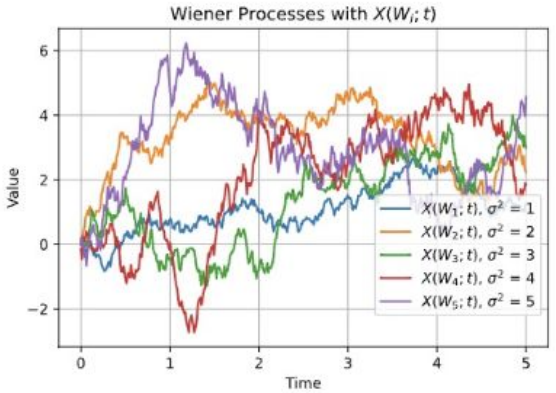
- Adding noise is important, but why ‘hardcode’?
- By generalising with infinite noise scales, we can
  - get higher quality samples
  - exact log-likelihood computation
  - controllable generation with inverse problem solving
- A stochastic process defines a process of generating infinite noise scales

2.1 What are SDE’s (and ODE’s)?

Wiener Process (Brownian Motion)

A real-valued continuous-time stochastic process characterized by the following properties:

1. Initial condition: $\boldsymbol{w}(0) = 0$ almost surely.
   Probability is 1.
2. Normally distributed increments:
   For every $0 \leq s < t, \boldsymbol{w}(t) - \boldsymbol{w}(s) \sim \mathcal{N}(0, t - s)$.
3. Independent increments:
   For every $0 \leq s < t, \boldsymbol{w}(t) - \boldsymbol{w}(s)$ is independent of the past values.
4. Continuity: $\boldsymbol{w}(t)$ is continuous almost surely.



Ordinary differential equation

Consider the ordinary differential equation (ODE)

$$\frac{dX}{dt}(t) = f(X(t), t)$$

which we also express as

$$dX_t = f(X_t, t)dt$$

where $X(t), f(X(t), t) \in \mathbb{R}^d$. Then, $\{X(t)\}_t$ is a deterministic curve.

We can think of the ODE as the limit

$$X_{k+1} = X_k + \Delta t f(X_k, k\Delta t), \quad k = 0, 1, \dots$$

under $\Delta t \to 0$, where $t = k\Delta t$.

Solution for ODE

$\{X(t)\}_{t=0}^{T}$ solves ODE if it satisfies the

- differential form of the ODE

$$\frac{dX}{dt}(t) = f(X(t), t)$$

- or the integral form of the ODE

$$X(t) = X_0 + \int_0^t f(X(s), s)ds$$

- Example:

$$\frac{dX}{dt} = -\beta X$$
$$X(t) = X(0)e^{-\beta t}$$

https://ernestryu.com/courses/FM/diffusion1.pdf

Stochastic differential equation

Consider the stochastic differential equation (SDE)

$$dX_t = f(X_t, t)dt + g(t)dW_t$$

where $X_t(t), f(X_t, t) \in \mathbb{R}^d$, $g(t) \in \mathbb{R}^{d\times d}$, and W_t is a d-dimensional Brownian motion or Wiener process. $\{X_t\}_t$ is a random process. (We can allow g to also depend on X_t, but this makes the equations more complicated.)

We can think of the SDE as the limit

$$X_{k+1} = X_k + \Delta t f(X_k, k\Delta t) + g(k\Delta t)\sqrt{\Delta t}Z_k, \qquad k = 0, 1, \dots$$

under $\Delta t \to 0$, where $t = k\ \Delta t$ and $Z_0, Z_1, \dots \sim \mathcal{N}(0, I)$. Precisely, $\{X_{\lfloor t/\Delta t \rfloor}\}_t \xrightarrow{\mathcal{D}} \{X_t\}_t$ on compact intervals.

https://ernestryu.com/courses/FM/diffusion1.pdf

B. Øksendal, *Stochastic Differential Equations: An Introduction with Applications*, 6th edition, Springer, 2003.

Solution for SDE

$\{X_t\}_{t=0}^T$ is a solution path for SDE if $\{X_t\}_{t=0}^T$ is nice# with probability distribution defined by

$$X_t = X_0 + \int_0^t f(X_s, s)ds + \int_0^t g(X_s, s)dW_s$$

where the Itô stochastic integral is defined as

$$\int_0^t g(X_s, s)dW_s = \lim_{\Delta t \to 0} \sum_{k=0}^{\lfloor t/\Delta t \rfloor} g(X_{k\Delta t}, \varepsilon k)\sqrt{\Delta t} Z_k$$

where $Z_1, Z_2, \ldots \sim \mathcal{N}(0, I)$ are IID.

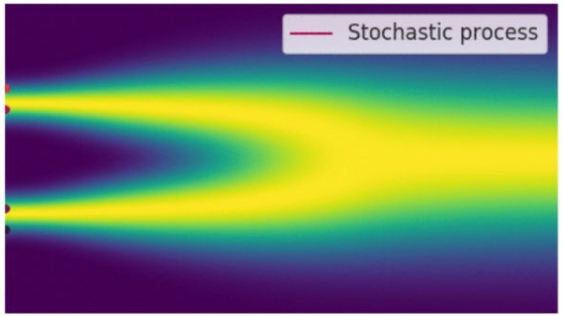
https://ernestryu.com/courses/FM/diffusion1.pdf

#right-continuous with left limits (càdlàg)

Stochastic processes via SDEs

- A stochastic process can be defined in terms of (solution of) a stochastic differential equation

$$\mathrm{d}\mathbf{x} = f(\mathbf{x}, t)\mathrm{d}t + g(t)\mathrm{d}\mathbf{w}$$



- The change in our random variable is governed a function of the variable itself and time (drift coefficient) plus stochastic perturbation (noise) whose scale is a function of time (diffusion coefficient)
- $f(\cdot, t) : \mathbb{R}^d \to \mathbb{R}^d, g \in \mathbb{R}$, $\mathbf{w}$ is Brownian motion, and $\mathrm{d}\mathbf{w}$ is infinitesimal white noise

Solutions to SDEs

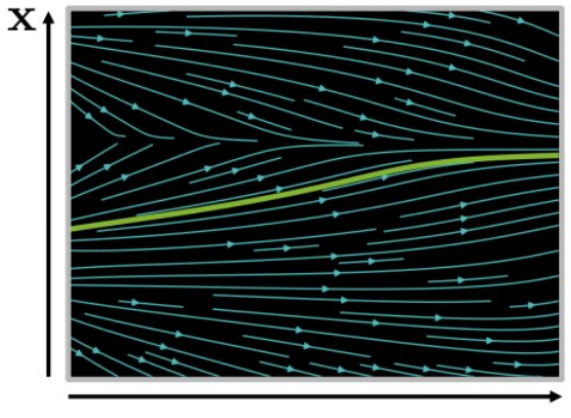
$$\mathbf{dx} = f(\mathbf{x}, t)\mathrm{dt} + g(t)\mathrm{dw}$$

- Solutions to the SDEs are stochastic random variables $\{\mathbf{x}(t)\}_{t\in[0,T]}$
- These random variables are stochastic trajectories over time
- The probability density of $\mathbf{x}(t)$ is $p_t(\mathbf{x})$ (analogous to $p_{\sigma_i}(\mathbf{x})$ for the discrete case)
- $p_0(\mathbf{x})$ means the distribution in the data space, i.e., $p_0(\mathbf{x}) = p(\mathbf{x})$
- $p_T(\mathbf{x})$ is the distribution after all the noising up for period T until we end up to our prior distribution for our data generation process, i.e., $p_T(\mathbf{x}) = \pi(\mathbf{x})$

Crash Course in Differential Equations

Ordinary Differential Equation (ODE):

$$\frac{\mathrm{d}\mathbf{x}}{\mathrm{d}t} = \mathbf{f}(\mathbf{x}, t) \quad \text{or} \quad \mathrm{d}\mathbf{x} = \mathbf{f}(\mathbf{x}, t)\mathrm{d}t$$



Analytical Solution:

$$\mathbf{x}(t) = \mathbf{x}(0) + \int_0^t \mathbf{f}(\mathbf{x}, \tau)\mathrm{d}\tau$$

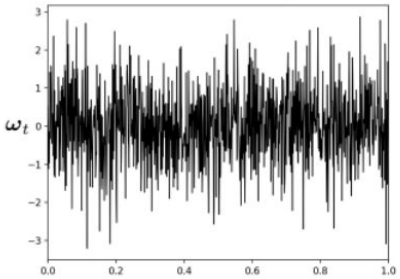
Iterative Numerical Solution:

$$\mathbf{x}(t + \Delta t) \approx \mathbf{x}(t) + \mathbf{f}(\mathbf{x}(t), t)\Delta t$$

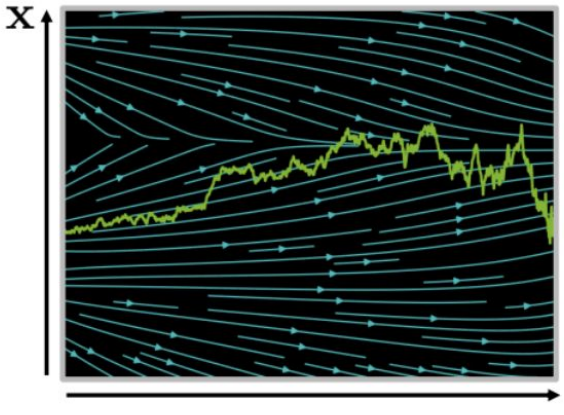
Stochastic Differential Equation (SDE):

$$\frac{\mathrm{d}\mathbf{x}}{\mathrm{d}t} = \underbrace{\mathbf{f}(\mathbf{x}, t)}_{\text{drift coefficient}} + \underbrace{\sigma(\mathbf{x}, t)\boldsymbol{\omega}_t}_{\text{diffusion coefficient}}$$

Wiener Process (Gaussian White Noise)



$$\left(\mathrm{d}\mathbf{x} = \mathbf{f}(\mathbf{x}, t)\mathrm{d}t + \sigma(\mathbf{x}, t)\mathrm{d}\boldsymbol{\omega}_t \right)$$

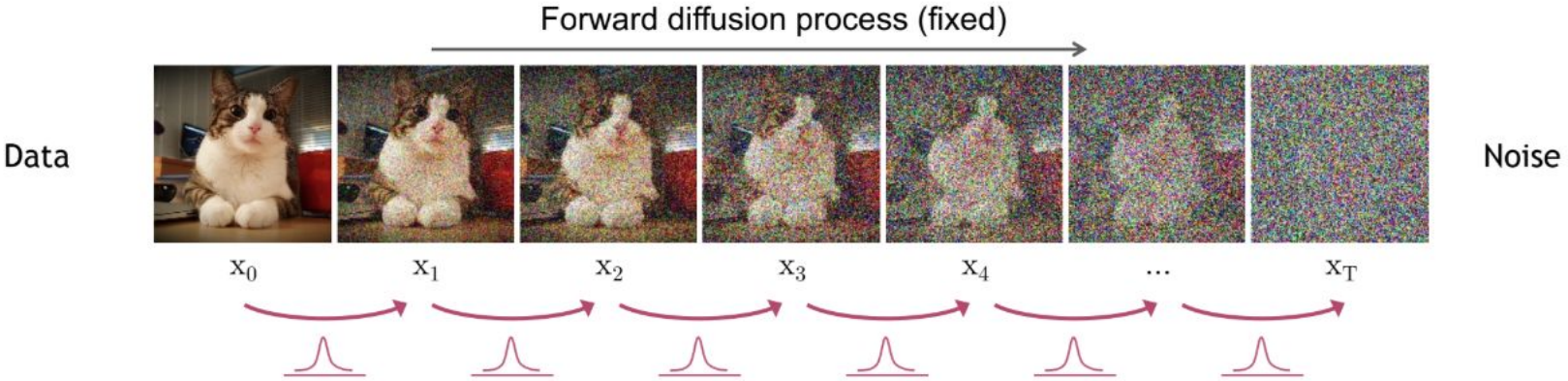


$$\mathbf{x}(t + \Delta t) \approx \mathbf{x}(t) + \mathbf{f}(\mathbf{x}(t), t)\Delta t + \sigma(\mathbf{x}(t), t)\sqrt{\Delta t}\,\mathcal{N}(\mathbf{0}, \mathbf{I})$$

2.2 SDE’s for DDPM Forward + Reverse Processes

Forward Diffusion Process

Consider the forward diffusion process again:

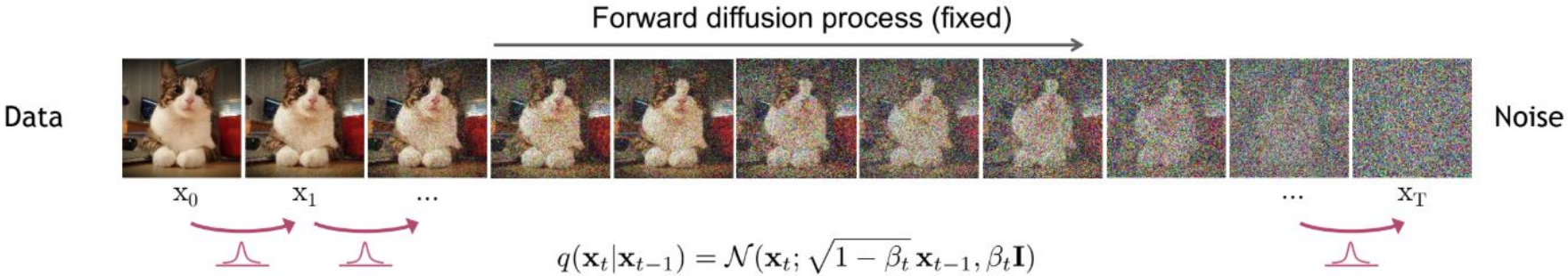


$$q(\mathbf{x}_t|\mathbf{x}_{t-1}) = \mathcal{N}(\mathbf{x}_t; \sqrt{1-\beta_t}\,\mathbf{x}_{t-1}, \beta_t\mathbf{I})$$

https://drive.google.com/file/d/1DYHDbt1tSl9oqm3O333biRYzSCOtdtmn/view

Forward Diffusion Process

Consider the limit of **many small steps**:



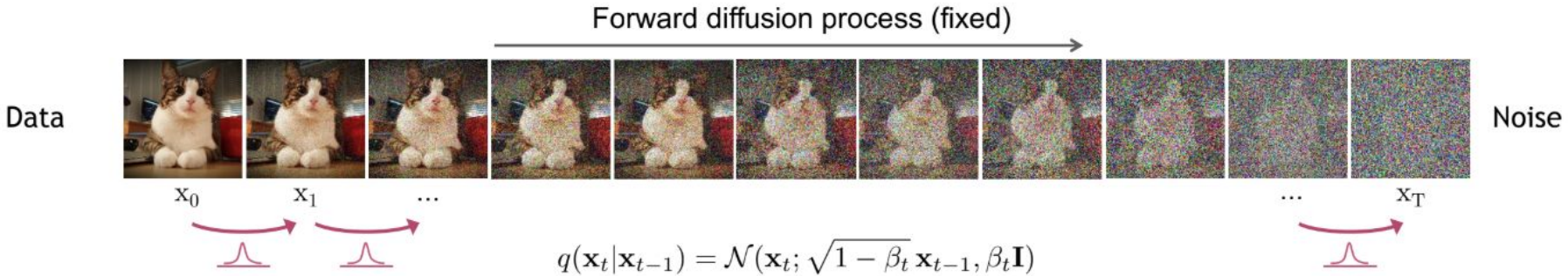
$$q(\mathbf{x}_t|\mathbf{x}_{t-1}) = \mathcal{N}(\mathbf{x}_t; \sqrt{1-\beta_t}\,\mathbf{x}_{t-1}, \beta_t\mathbf{I})$$

$$\Rightarrow \quad \mathbf{x}_t = \sqrt{1-\beta_t}\,\mathbf{x}_{t-1} + \sqrt{\beta_t}\,\mathcal{N}(\mathbf{0}, \mathbf{I})$$

https://drive.google.com/file/d/1DYHDbt1tSl9oqm3O333biRYzSCOtdtmn/view

Forward Diffusion Process

Consider the limit of **many small steps:**



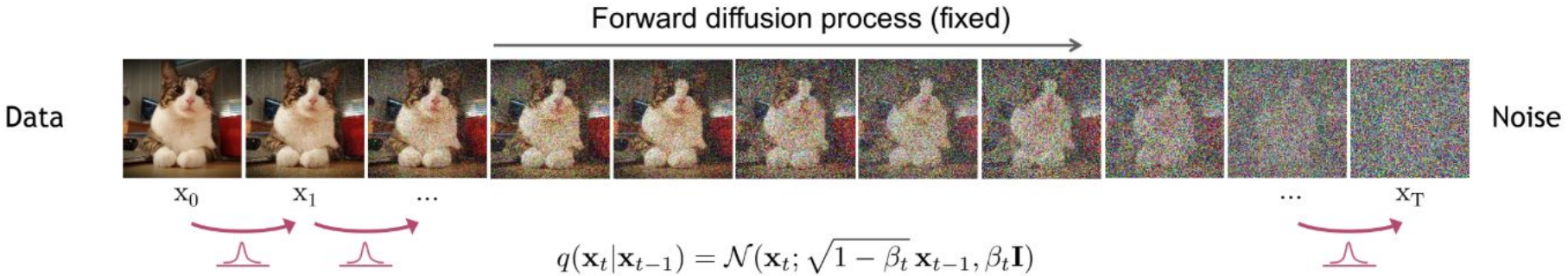
$$q(\mathbf{x}_t|\mathbf{x}_{t-1}) = \mathcal{N}(\mathbf{x}_t; \sqrt{1-\beta_t}\,\mathbf{x}_{t-1}, \beta_t\mathbf{I})$$

$$\begin{aligned}\mathbf{x}_t &= \sqrt{1-\beta_t}\,\mathbf{x}_{t-1} + \sqrt{\beta_t}\,\mathcal{N}(\mathbf{0},\mathbf{I}) \\ \Rightarrow \quad &= \sqrt{1-\beta(t)\Delta t}\,\mathbf{x}_{t-1} + \sqrt{\beta(t)\Delta t}\,\mathcal{N}(\mathbf{0},\mathbf{I}) \qquad (\beta_t := \beta(t)\Delta t)\end{aligned}$$

https://drive.google.com/file/d/1DYHDbt1tSl9oqm3O333biRYzSCOtdtmn/view

Forward Diffusion Process

Consider the limit of **many small steps:**



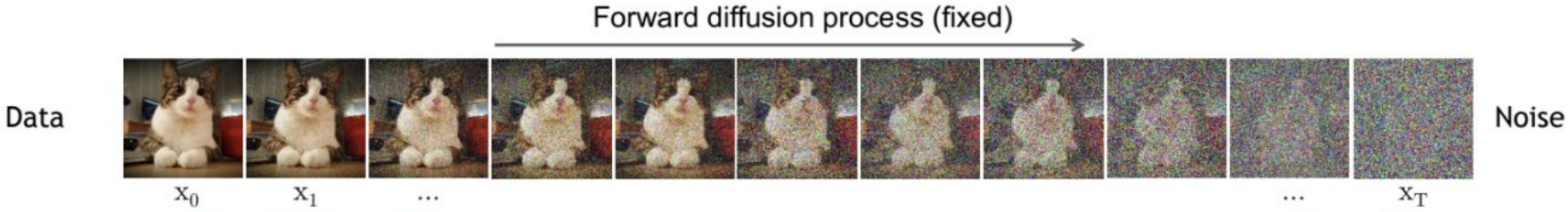
$$q(\mathbf{x}_t|\mathbf{x}_{t-1}) = \mathcal{N}(\mathbf{x}_t; \sqrt{1-\beta_t}\,\mathbf{x}_{t-1}, \beta_t\mathbf{I})$$

$$\begin{aligned}
\mathbf{x}_t &= \sqrt{1-\beta_t}\,\mathbf{x}_{t-1} + \sqrt{\beta_t}\,\mathcal{N}(\mathbf{0},\mathbf{I}) \\
&= \sqrt{1-\beta(t)\Delta t}\,\mathbf{x}_{t-1} + \sqrt{\beta(t)\Delta t}\,\mathcal{N}(\mathbf{0},\mathbf{I}) && (\beta_t := \beta(t)\Delta t) \\
&\approx \mathbf{x}_{t-1} - \frac{\beta(t)\Delta t}{2}\mathbf{x}_{t-1} + \sqrt{\beta(t)\Delta t}\,\mathcal{N}(\mathbf{0},\mathbf{I}) && \text{(Taylor expansion)}
\end{aligned}$$

https://drive.google.com/file/d/1DYHDbt1tSl9oqm3O333biRYzSCOtdtmn/view

Forward Diffusion Process as Stochastic Differential Equation

Consider the limit of **many small steps:**



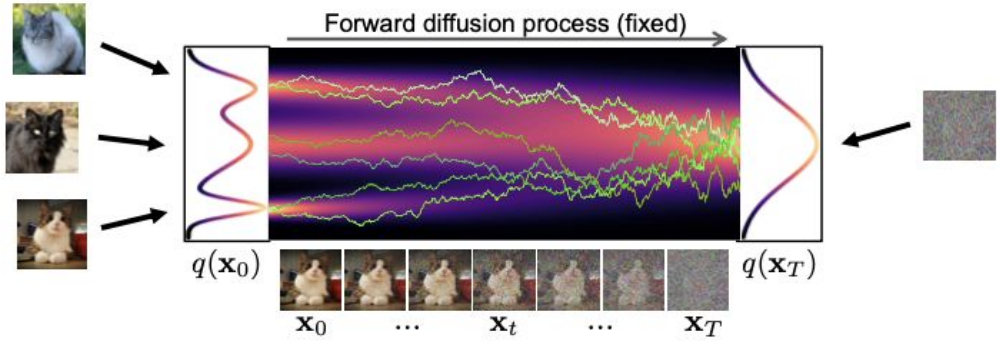
$$\mathbf{x}_t \approx \mathbf{x}_{t-1} - \frac{\beta(t)\Delta t}{2}\mathbf{x}_{t-1} + \sqrt{\beta(t)\Delta t}\,\mathcal{N}(\mathbf{0}, \mathbf{I})$$

$$\mathrm{d}\mathbf{x}_t = -\frac{1}{2}\beta(t)\mathbf{x}_t\,\mathrm{d}t + \sqrt{\beta(t)}\,\mathrm{d}\boldsymbol{\omega}_t$$

Stochastic Differential Equation (SDE)
describing the diffusion in infinitesimal limit

https://drive.google.com/file/d/1DYHDbt1tSl9oqm3O333biRYzSCOtdtmn/view

Forward Diffusion Process as Stochastic Differential Equation

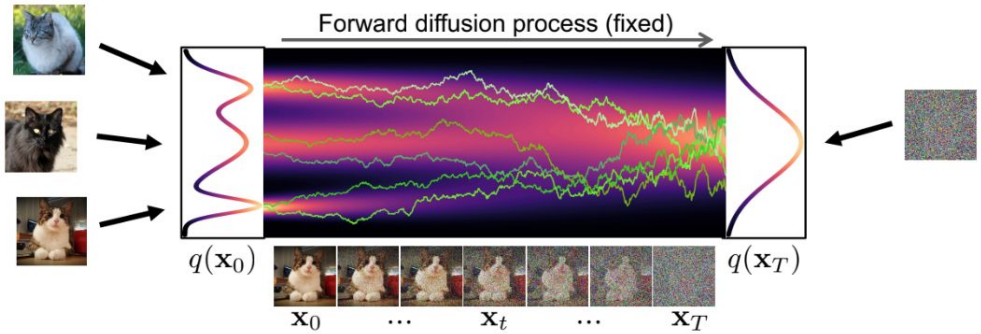


Forward Diffusion SDE:

$$\mathrm{d}\mathbf{x}_t = \underbrace{-\frac{1}{2}\beta(t)\mathbf{x}_t\,\mathrm{d}t}_{\text{drift term (pulls towards mode)}} + \underbrace{\sqrt{\beta(t)}\,\mathrm{d}\boldsymbol{\omega}_t}_{\text{diffusion term (injects noise)}}$$

Song et al., *ICLR*, 2021

The Generative Reverse Stochastic Differential Equation



Forward Diffusion SDE:

$$\mathrm{d}\mathbf{x}_t = -\frac{1}{2}\beta(t)\mathbf{x}_t\,\mathrm{d}t + \sqrt{\beta(t)}\,\mathrm{d}\boldsymbol{\omega}_t$$

But what about the reverse direction, necessary for generation?

https://drive.google.com/file/d/1DYHDbt1tSl9oqm3O333biRYzSCOtdtmn/view

Deriving SDE for Reverse Process in Diffusion

Theorem: Given a forward diffusion process

$$d\mathbf{x} = \underbrace{\mathbf{f}(\mathbf{x}, t)}_{\text{drift}}\, dt + \underbrace{g(t)}_{\text{diffusion}}\, d\mathbf{w}.$$

where $d\mathbf{w} = \boldsymbol{\xi}(t)dt.$

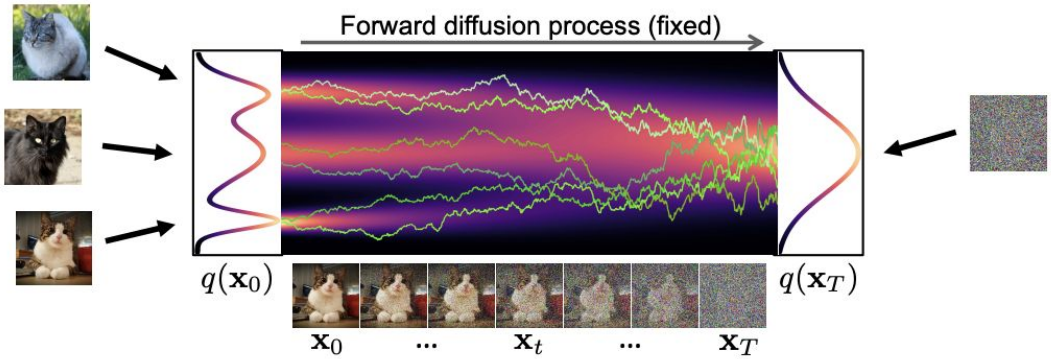
We get the corresponding SDE for the reverse process (Anderson, derived in physics section)

$$d\mathbf{x} = [\underbrace{\mathbf{f}(\mathbf{x}, t)}_{\text{drift}} - g(t)^2 \underbrace{\nabla_{\mathbf{x}} \log p_t(\mathbf{x})}_{\text{score function}}]\, dt \quad + \quad \underbrace{g(t)d\overline{\mathbf{w}}}_{\text{reverse-time diffusion}}$$

where $p_t(\mathbf{x})$ is the probability distribution of x at time t,

and $\overline{\mathbf{w}}$ is Wiener process when time flows backward.

The Generative Reverse Stochastic Differential Equation



Forward Diffusion SDE:

$$\mathrm{d}\mathbf{x}_t = -\frac{1}{2}\beta(t)\mathbf{x}_t\,\mathrm{d}t + \sqrt{\beta(t)}\,\mathrm{d}\boldsymbol{\omega}_t$$

Reverse Generative Diffusion SDE:

$$\mathrm{d}\mathbf{x}_t = \underbrace{\left[-\frac{1}{2}\beta(t)\mathbf{x}_t - \beta(t)\nabla_{\mathbf{x}_t}\log q_t(\mathbf{x}_t)\right]}_{\text{drift term}}\mathrm{d}t + \underbrace{\sqrt{\beta(t)}\,\mathrm{d}\bar{\boldsymbol{\omega}}_t}_{\text{diffusion term}}$$

"Score Function"

➡ **Simulate reverse diffusion process: Data generation from random noise!**

https://drive.google.com/file/d/1DYHDbt1tSl9oqm3O333biRYzSCOtdtmn/view

The Generative Reverse Stochastic Differential Equation

But how to get the score function $\nabla_{\mathbf{x}_t} \log q_t(\mathbf{x}_t)$?

Forward Diffusion SDE: $\mathrm{d}\mathbf{x}_t = -\frac{1}{2}\beta(t)\mathbf{x}_t\,\mathrm{d}t + \sqrt{\beta(t)}\,\mathrm{d}\omega_t$

Use Denoising Score Matching (DSM) to learn score

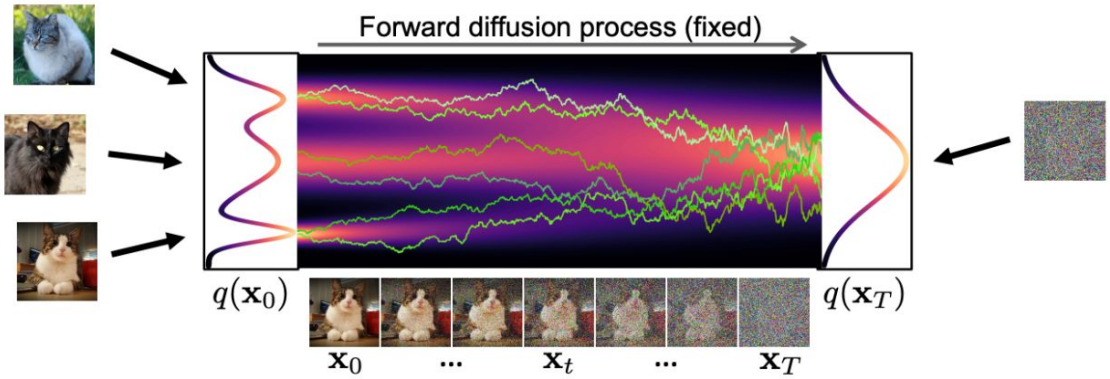
Reverse Generative Diffusion SDE:

$$\mathrm{d}\mathbf{x}_t = \left[-\frac{1}{2}\beta(t)\mathbf{x}_t - \beta(t)\nabla_{\mathbf{x}_t}\log q_t(\mathbf{x}_t)\right]\mathrm{d}t + \sqrt{\beta(t)}\,\mathrm{d}\bar{\omega}_t$$

drift term — diffusion term — "Score Function"

➡ Simulate reverse diffusion process: Data generation from random noise!

Score Matching



Naïve idea, learn model for the score function by direct regression?

$$\min_{\boldsymbol{\theta}} \underbrace{\mathbb{E}_{t\sim\mathcal{U}(0,T)}}_{\substack{\text{diffusion}\\ \text{time } t}} \underbrace{\mathbb{E}_{\mathbf{x}_t\sim q_t(\mathbf{x}_t)}}_{\substack{\text{diffused}\\ \text{data } \mathbf{x}_t}} \underbrace{\tilde{w}(t)}_{\substack{\text{weighting}\\ \text{function}}} \cdot \underbrace{||\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}_t, t)}_{\substack{\text{neural}\\ \text{network}}} - \underbrace{\nabla_{\mathbf{x}_t} \log q_t(\mathbf{x}_t)||_2^2}_{\substack{\text{score of}\\ \text{diffused data}\\ \text{(marginal)}}}$$

➡ **But $\nabla_{\mathbf{x}_t} \log q_t(\mathbf{x}_t)$ (score of the *marginal diffused density* $q_t(\mathbf{x}_t)$) is not tractable!**

Variational Diffusion Models

Variational diffusion models have a general form with an arbitrary forward diffusion (forward jump) distribution $q(\boldsymbol{x}_t|\boldsymbol{x}_0)$:

$$q(\boldsymbol{x}_t|\boldsymbol{x}_0) = \mathcal{N}(\boldsymbol{x}_t; \alpha(t)\boldsymbol{x}_0, \sigma^2(t)I)$$

Now we follow the notations of Kingma et al.

which satisfies the condition that **Signal-to-Noise Ratio (SNR)** $\frac{\alpha^2(t)}{\sigma^2(t)}$ monotonically decreases over time.

Lu et al., DPM-Solver: A Fast ODE Solver for Diffusion Probabilistic Model Sampling in Around 10 Steps, NeurIPS 2022.

Diffusion Model → SDE

Given a forward diffusion distribution:

$$q(\boldsymbol{x}_t|\boldsymbol{x}_0) = \mathcal{N}(\boldsymbol{x}_t; \alpha(t)\boldsymbol{x}_0, \sigma^2(t)I),$$

its corresponding SDE becomes:

$$dx(t) = f(t)x(t)dt + g(t)d\boldsymbol{w}$$

- $f(t) = \frac{d \log \alpha(t)}{dt} = \frac{d \log \alpha_t}{dt}$
- $g^2(t) = \frac{d\sigma^2(t)}{dt} - 2\frac{d \log \alpha(t)}{dt}\sigma^2(t) = \frac{d\sigma_t^2}{dt} - 2\frac{d \log \alpha_t}{dt}\sigma_t^2$

2.2 SDE’s for SMLD

Recall: VP-SDE vs. VE-SDE

- **DDPM** is a special case (Variance Preserving) when the forward transitional distribution $q(\boldsymbol{x}_t|\boldsymbol{x}_{t-1})$ is defined as

$$q(\boldsymbol{x}_t|\boldsymbol{x}_{t-1}) = \mathcal{N}\left(\boldsymbol{x}_t; \sqrt{1-\beta_t}\boldsymbol{x}_{t-1}, \beta_t I\right)$$

- **SMLD** (Score Matching with Langevin Dynamics) is another case (Variance Exploding) when the forward transitional distribution $q(\boldsymbol{x}_t|\boldsymbol{x}_{t-1})$ is:

$$q(\boldsymbol{x}_t|\boldsymbol{x}_{t-1}) = \mathcal{N}\left(\boldsymbol{x}_t; \boldsymbol{x}_{t-1}, \left(\sigma_i^2 - \sigma_{i-1}^2\right)I\right)$$

Karras et al., Elucidating the Design Space of Diffusion-Based Generative Models, NeurIPS 2022. Appendix B.

SDE → Diffusion Model

For a forward process SDE:

$$dx(t) = f(t)x(t)dt + g(t)d\boldsymbol{w},$$

its corresponding forward diffusion distribution is defined as:

$$q(\boldsymbol{x}_t|\boldsymbol{x}_0) = \mathcal{N}(\boldsymbol{x}_t; \alpha(t)\boldsymbol{x}_0, \sigma^2(t)I)$$

- $\alpha(t) = e^{\int_0^t f(s)ds}$
- $\sigma^2(t) = \int_0^t \frac{g(s)^2}{\alpha(s)^2} ds$

VE and VP noise as SDEs

Noise perturbations used in SMLD and DDPM can be interpreted as discretizations of two different SDEs.[4]

$$x_i = x_{i-1} + \sqrt{\sigma_i^2 - \sigma_{i-1}^2} z_{i-1}$$

$$\mathbf{dx} = \sqrt{\frac{d[\sigma^2(t)]}{\mathbf{dt}}}\,\mathbf{dw}$$

Continuous process corresponding to SMLD gives rise to **VE SDE**.

$$x_i = \sqrt{1-\beta_i} x_{i-1} + \sqrt{\beta_i} z_{i-1}$$

$$\mathbf{dx} = -\frac{1}{2}\beta(t)x\,\mathbf{dt} + \sqrt{\beta(t)}\,\mathbf{dw}$$

Continuous process corresponding to DDPM gives rise to **VP SDE**.

https://jimut123.github.io/blogs/IITB_Courses/CS726/CS_726_Presentation.pdf

[4]Song, Y., Sohl-Dickstein, J., Kingma, D. P., Kumar, A., Ermon, S., & Poole, B. (2020). Score-based generative modeling through stochastic differential equations.

Variance Preserving (VP) SDE

- In the continuous setup of Variance Preserving (VP) SDE:

$$q(\boldsymbol{x}_t|\boldsymbol{x}_{t-1}) = \mathcal{N}\left(\boldsymbol{x}_t; \sqrt{1-\beta(t)}\boldsymbol{x}_{t-1}, \beta(t)I\right)$$

- The forward jump distribution becomes:

$$q(\boldsymbol{x}_t|\boldsymbol{x}_0) = \mathcal{N}(\boldsymbol{x}_t; \alpha(t)\boldsymbol{x}_0, \sigma^2(t)I)$$

where $\alpha(t) = e^{-\frac{1}{2}\int_0^t \beta(s)ds}$ and $\sigma^2(t) = 1 - \alpha^2(t)$.

Variance Preserving (VP) SDE

$$\alpha(t) = e^{-\frac{1}{2}\int_0^t \beta(s)ds} \qquad \sigma^2(t) = 1 - \alpha^2(t)$$

Note that $0 \leq \beta_1 \leq \beta_2 \leq \cdots \leq \beta_T$.

Q. Does the Signal-to-Noise Ratio (SNR) $\frac{\alpha^2(t)}{\sigma^2(t)}$ decrease over time?

Variance Preserving (VP) SDE

$$\alpha(t) = e^{-\frac{1}{2}\int_0^t \beta(s)ds} \qquad \sigma^2(t) = 1 - \alpha^2(t)$$

- $f(t) = \frac{d \log \alpha(t)}{dt} = \frac{d \log \alpha_t}{dt}$

- $g^2(t) = \frac{d\sigma^2(t)}{dt} - 2\frac{d \log \alpha(t)}{dt}\sigma^2(t)$

Variance Preserving (VP) SDE

Given the forward transitional distribution of the VP-SDE case:

$$q(\boldsymbol{x}_t|\boldsymbol{x}_{t-1}) = \mathcal{N}\left(\boldsymbol{x}_t; \sqrt{1-\beta(t)}\boldsymbol{x}_{t-1}, \beta(t)\mathbf{I}\right)$$

The corresponding SDE for the forward process is defined as:

$$dx(t) = \underbrace{-\frac{1}{2}\beta(t)}_{f(t)}x(t)dt + \underbrace{\sqrt{\beta(t)}}_{g(t)}d\boldsymbol{w},\ x(0)\sim q(\boldsymbol{x}_0)$$

Forward Sampling Equation of SMLD

Intuition

- gradually add more noise to data along variance schedule $\sigma(t)$ that increases smoothly $t \in [0, 1]$
- Discrete update at time t with step Δt

$$x(t + \Delta t) = x(t) + \sqrt{\sigma(t + \Delta t)^2 - \sigma(t)^2}\ z(t) \quad , \quad z(t) \sim \mathcal{N}(0, I)$$

$$\sqrt{\sigma(t + \Delta t)^2 - \sigma(t)^2} \approx \sqrt{\frac{d\left[\sigma(t)^2\right]}{dt}\,\Delta t}$$

- In the continuous limit $\Delta t \to 0$ this becomes an SDE

$$dx = \sqrt{\frac{d\left[\sigma(t)^2\right]}{dt}}\, dW_t$$

- This SDE captures the instantaneous rate of variance injection
- Equivalent to the discrete forward noising process in the limit of infinitesimal steps

Reverse Sampling Equation of SMLD

Defined as an SDE:

$$d\mathbf{x} = -\left(\frac{d[\sigma(t)^2]}{dt}\nabla_{\mathbf{x}}\log p_t(\mathbf{x})\right)dt + \sqrt{\frac{d[\sigma(t)^2]}{dt}}\,d\overline{\mathbf{w}}.$$

Proof. We recognize that

$$\mathbf{f}(\mathbf{x}, t) = 0, \qquad \text{and} \qquad g(t) = \sqrt{\frac{d[\sigma(t)^2]}{dt}}.$$

As a result, if we write the reverse equation Eqn (4.11), we should have

$$\begin{aligned} d\mathbf{x} &= [\mathbf{f}(\mathbf{x}, t) - g(t)^2\nabla_{\mathbf{x}}\log p_t(\mathbf{x})]\,dt \;+\; g(t)d\overline{\mathbf{w}} \\ &= -\left(\frac{d[\sigma(t)^2]}{dt}\nabla_{\mathbf{x}}\log p_t(\mathbf{x}(t))\right)dt + \sqrt{\frac{d[\sigma(t)^2]}{dt}}\,d\overline{\mathbf{w}}. \end{aligned}$$

Perturbing data with noise from SDEs

- This SDE is the generalisation of the finite scaling $\sigma_0, \ldots, \sigma_L$
- Earlier we were perturbing according to a geometric progression of scales
- Now, we perturb with noise controlled by the SDE
- We select manually which SDE to model the process with
- If we were to select $\mathrm{d}\mathbf{x} = e^t \mathrm{d}\mathbf{w}$, we would add Gaussian noise $\mathrm{d}\mathbf{w}$ with a scale e^t that grows exponentially with time

Score-based Generative Models (SGM)

Mila

1. Training

$$\mathbb{E}_{t,\ \mathbf{x}_0 \sim p_{\text{data}},\, \epsilon \sim \mathcal{N}(\mathbf{0},\mathbf{I})} \left|\left| \mathbf{s}_\theta(\mathbf{x}_t, t) - \frac{\mathbf{x}_0 - \mathbf{x}_t}{\sigma_t^2} \right|\right|_2^2$$

$$(\mathbf{x}_t = \mathbf{x}_0 + \sigma_t \epsilon)$$

2. Generation

$$\mathbf{x}_{t-1} = \mathbf{x}_t + \lambda_t \, \mathbf{s}_\theta(\mathbf{x}_t, t) + \sqrt{2\lambda_t}\, \mathbf{z}_t \qquad ; \lambda_t = \lambda \sigma_t^2 / \sigma_1^2;\ \mathbf{z}_t \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$$

Solving the reverse SDE

- Once we have trained the score-matching function, we can solve the reverse SDE from the prior π all the way to our data distribution p_0 to generate new data
- We can use any numerical solver, e.g., the Euler-Maruyama, for a small negative Δt

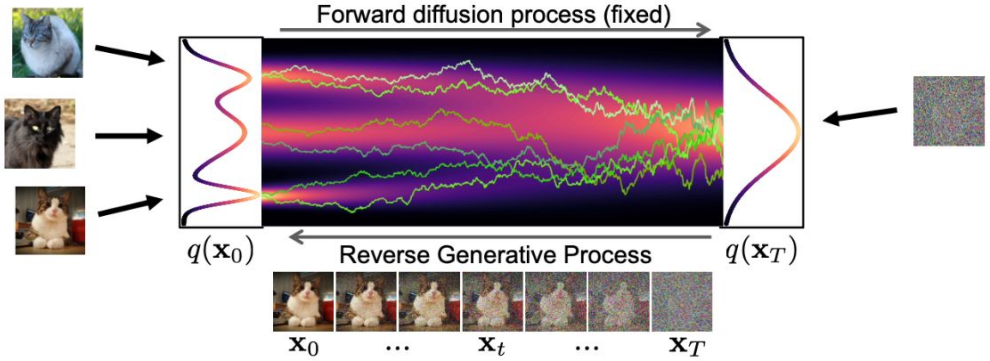
$$\Delta \mathbf{x} \leftarrow \left[f(\mathbf{x}, t) - g^2(t) s_\theta(\mathbf{x}, t) \right] \Delta t + g(t) \sqrt{|\Delta t|}\, \mathbf{z}_t, \quad \mathbf{z}_t \sim \mathcal{N}(0, I)$$

$$\mathbf{x} \leftarrow \mathbf{x} + \Delta \mathbf{x}$$

$$t \leftarrow t + \Delta t$$

- With better sampling procedures for SDEs and better architectures, one gets state-of-the-art in generated samples

Diffusion Models as Energy-based Models



- Assume an Energy-based Model (EBM): $p_{\boldsymbol{\theta}}(\mathbf{x}, t) = \dfrac{e^{-E_{\boldsymbol{\theta}}(\mathbf{x},t)}}{\mathcal{Z}_{\boldsymbol{\theta}}(t)}$
- Sample EBM via Langevin dynamics: $\mathbf{x}_{i+1} = \mathbf{x}_i - \eta \nabla_{\mathbf{x}} E_{\boldsymbol{\theta}}(\mathbf{x}_i, t) + \sqrt{2\eta}\, \mathcal{N}(\mathbf{0}, \mathbf{I})$
- Requires only gradient of energy $-\nabla_{\mathbf{x}} E_{\boldsymbol{\theta}}(\mathbf{x}, t)$, not $E_{\boldsymbol{\theta}}(\mathbf{x}, t)$ itself, nor $\mathcal{Z}_{\boldsymbol{\theta}}(t)$!

In diffusion models, we learn "energy gradients" for all diffused distributions directly:

$$\nabla_{\mathbf{x}} \log q_t(\mathbf{x}) \approx \mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}, t) =: \nabla_{\mathbf{x}} \log p_{\boldsymbol{\theta}}(\mathbf{x}, t) = -\nabla_{\mathbf{x}} E_{\boldsymbol{\theta}}(\mathbf{x}, t) - \underbrace{\nabla_{\mathbf{x}} \log \mathcal{Z}_{\boldsymbol{\theta}}(t)}_{=0} = -\nabla_{\mathbf{x}} E_{\boldsymbol{\theta}}(\mathbf{x}, t)$$

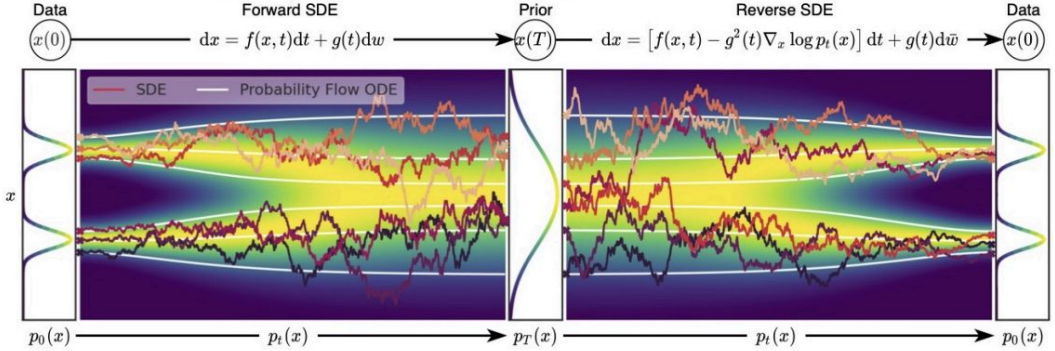
➡ Diffusion Models model energy gradient directly, along entire diffusion process, and avoid modeling partition function. Different noise levels along diffusion are analogous to annealed sampling in EBMs.

Probability flow ODE

- With Langevin MCMC samplers and SDE solvers we can't get exact log-likelihoods
- We can convert the SDE to a corresponding ODE without changing the marginal distributions $\{p_t(\mathbf{x})\}_{t\in[0,T]}$, name the probability flow ODE

$$\mathrm{d}\mathbf{x} = \left[f(\mathbf{x}, t) - g^2(t)\,\nabla_{\mathbf{x}} \log p_t(\mathbf{x})\right]\mathrm{d}t$$

- Solving the ODE, we can get the exact log-likelihood



Recall: Gaussians and Score Function

Let $p(\boldsymbol{z}) = \mathcal{N}(\mathbf{0}, \mathbf{I})$ and $\boldsymbol{x} = \boldsymbol{\mu} + \Sigma^{\frac{1}{2}}\boldsymbol{z}$. Then,

- $p(\boldsymbol{x}) = \mathcal{N}(\boldsymbol{\mu}, \Sigma)$.
- $\log p(\boldsymbol{x}) = -\frac{1}{2}(\boldsymbol{x} - \boldsymbol{\mu})^T \Sigma^{-1}(\boldsymbol{x} - \boldsymbol{\mu}) + c$
- $\nabla_{\boldsymbol{x}} \log p(\boldsymbol{x}) = -\Sigma^{-1}(\boldsymbol{x} - \boldsymbol{\mu})$

Recall: Forward Jump

Let $\boldsymbol{x}_t$ be the noisy data point at time step t from $\boldsymbol{x}_0$ in the diffusion process.

$$p(\boldsymbol{x}_t|\boldsymbol{x}_0) = \mathcal{N}\left(\sqrt{\bar{\alpha}_t}\boldsymbol{x}_0, (1-\bar{\alpha}_t)\mathbf{I}\right)$$

Recall: Tweedie's Formula

How to estimate the true mean of a normal distribution from samples drawn from it?

$$\text{For a } \boldsymbol{x} \sim p(\boldsymbol{x}) = \mathcal{N}(\boldsymbol{x}; \boldsymbol{\mu}, \boldsymbol{\Sigma}),$$

$$\mathbb{E}[\boldsymbol{\mu}|\boldsymbol{x}] = \boldsymbol{x} + \boldsymbol{\Sigma}\nabla_{\boldsymbol{x}} \log p(\boldsymbol{x})$$

Q3. Given a sample $\boldsymbol{x}_t$, what is the expected value of $\boldsymbol{x}_0$, $\boldsymbol{x}_{0|t} = \mathbb{E}[\boldsymbol{x}_0|\boldsymbol{x}_t]$?

$$p(\boldsymbol{x}_t|\boldsymbol{x}_0) = \mathcal{N}\left(\sqrt{\bar{\alpha}_t}\boldsymbol{x}_0, (1-\bar{\alpha}_t)\mathbf{I}\right)$$

From Tweedie's formula:

$$\mathbb{E}\left[\sqrt{\bar{\alpha}_t}\boldsymbol{x}_0|\boldsymbol{x}_t\right] = \sqrt{\bar{\alpha}_t}\mathbb{E}[\boldsymbol{x}_0|\boldsymbol{x}_t] = \boldsymbol{x}_t + (1-\bar{\alpha}_t)\nabla_{\boldsymbol{x}} \log p(\boldsymbol{x}_t|\boldsymbol{x}_0)$$

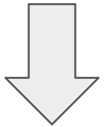
$$\therefore \boldsymbol{x}_{0|t} = \mathbb{E}[\boldsymbol{x}_0|\boldsymbol{x}_t] = \frac{1}{\sqrt{\bar{\alpha}_t}}(\boldsymbol{x}_t + (1-\bar{\alpha}_t)\nabla_{\boldsymbol{x}} \log p(\boldsymbol{x}_t|\boldsymbol{x}_0))$$

Part 3. Physics for Diffusion

Table of Contents

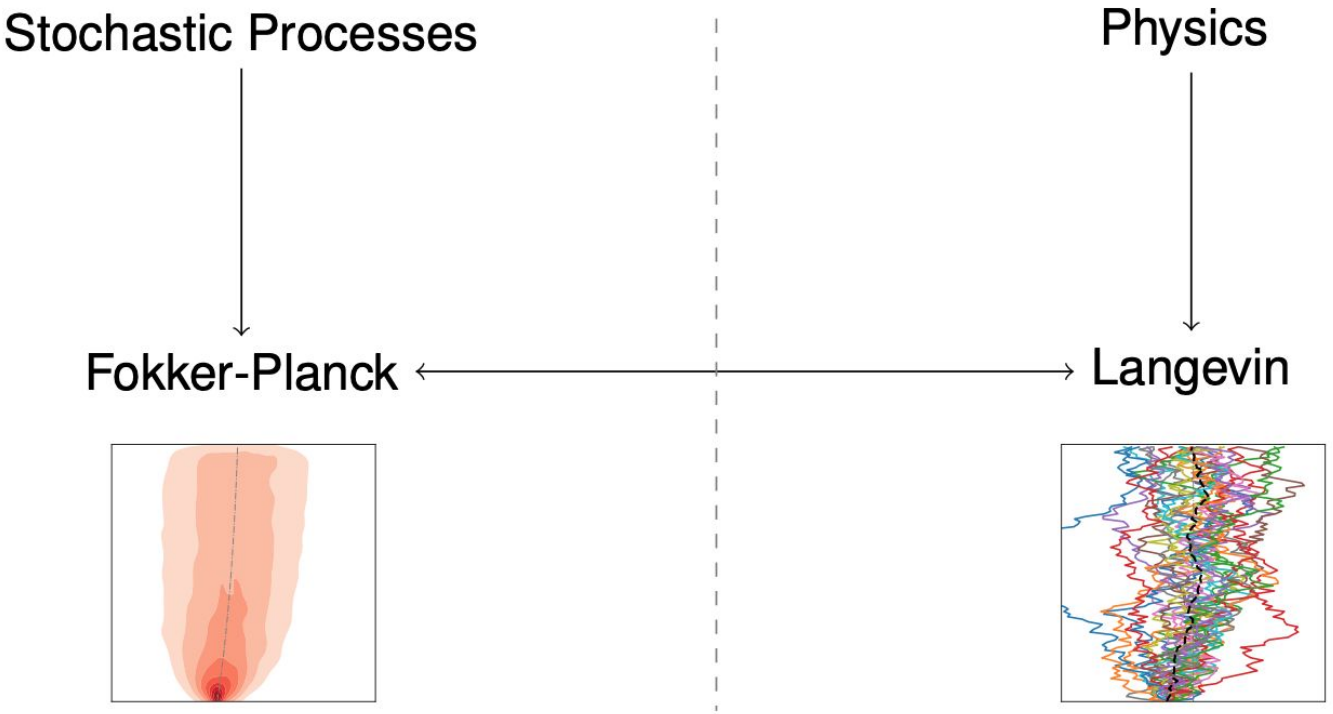
Motivation for Learning Physics of Diffusion

Having physical interpretation of SDE's associated with diffusion models



Gain deeper understanding of what these equations mean

High-Level Connection between SDE's and Physics



3.1 :Linear Langevin Equation for Forward Diffusions

Intro to Langevin Equation

The **Langevin Equation** is an SDE of the form

$$\dot{\xi} = h(\xi, t) + g(\xi, t)\Gamma(t), \quad \text{where } \xi := \xi(t) \text{ is a R.V. at time } t$$

where $\Gamma(t)$ is a stochastic process called the **Langevin Force,** which satisfies:

(i) $\mathbb{E}[\Gamma(t)] = 0$ for all t so that its mean function is a constant zero;

(ii) $\mathbb{E}[\Gamma(t)\Gamma(t')] = q\delta(t - t')$ for all t and t', i.e., the autocorrelation function is a delta function with an amplitude q.

NOTE: Typically choose $\Gamma(t)$ to be **Gaussian White Noise** b/c it can describe many physical phenomena

Linear Langevin Equation & Wiener Process

The ***Linear* Langevin Equation** is a Langevin Equation with constant h and g = 1

$$\dot{\xi} + \gamma\xi = \Gamma(t),$$

where gamma is a constant

A ***Wiener Process*** is a Linear Langevin Equation where gamma = 1.

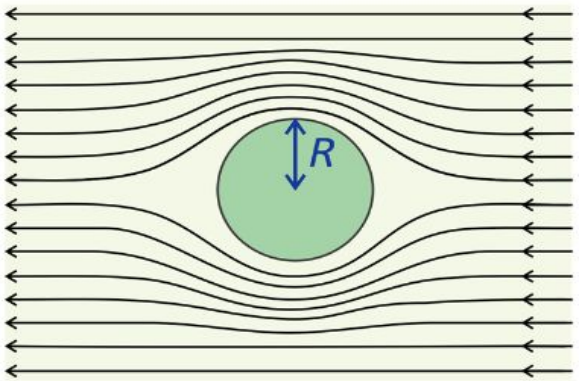
Linear Langevin Equation can describe certain physical phenomena in fluid dynamics!!! (more on next slide)

Application of Linear Langevin Equation to Physics

Suppose we have a particle suspended in a fluid. Langevin-Stokes' equation models its microscopic movement (called Brownian motion) as:

$$F(t) = -\alpha v(t) + F_f(t) \qquad \textit{where}$$

- $F(t)$ is friction, or drag force, applied to a particle suspended in a fluid, as a function of time
- $v(t)$ is the velocity of the fluid
- $\alpha = 6\pi\mu R$ where R is the radius of the particle and μ is the viscosity of the fluid
- $F_f(t)$ is a stochastic term as function of time



Application of Linear Langevin Equation to Physics

Combining this with Newton's 2nd Law of Motion $F(t) = m\dot{v}(t)$,

and letting $\Gamma(t) := \frac{F_f(t)}{m}$ and $\gamma \overset{\text{def}}{=} \frac{\alpha}{m}$, we get

$$\frac{dv(t)}{dt} + \gamma v(t) = \Gamma(t) \quad \iff \quad \dot{v} + \gamma v = \Gamma(t)$$

where it is assumed that $\mathbb{E}[\Gamma(t)] = 0$ for all t and $\mathbb{E}[\Gamma(t)\Gamma(t')] = q\delta(t - t')$ for all t and t'

This takes form of Linear Langevin Equation (which recall, is $\dot{\xi} + \gamma\xi = \Gamma(t),$)

THUS, we see that the Linear Langevin Equation can model a particle's microscopic movement in a fluid!

Interpretation of $\Gamma(t)$ in this Example

The stochastic term $\Gamma(t)$ in the Linear Langevin Equation

$$\dot{v} + \gamma v = \Gamma(t)$$

represents the acceleration that the particle gains from collisions with water molecules

*** If particle is too big, it is unaffected by water molecules -> zero Langevin force

Can it also be applied to Diffusion? YES

Application of Linear Langevin Equation to Diffusion

We can also use the Langevin Equation to model the Forward Diffusion Process!

Example 5.1. Forward DDPM. Recall that a DDPM forward diffusion equation is given by

$$d\mathbf{x} = \underbrace{-\frac{\beta(t)}{2}}_{=f(t)} \mathbf{x}\, dt + \underbrace{\sqrt{\beta(t)}}_{=g(t)} d\mathbf{w}.$$

Expressing it in the Langevin equation form, we can write it as

$$\dot{\boldsymbol{\xi}}(t) + f(t)\boldsymbol{\xi}(t) = g(t)\Gamma(t), \qquad \text{where} \quad \Gamma(t) \sim \mathcal{N}(0, \mathbf{I}).$$

Solution to Forward Diffusion SDE

Theorem 5.2. Consider the following differential equation

$$\dot{\xi}(t) + \gamma\xi(t) = \Gamma(t),$$

with an initial condition $\xi(0) = \xi_0$. The solution is given by

$$\xi(t) = \xi_0 e^{-\gamma t} + \int_0^t e^{-\gamma(t-t')}\Gamma(t')\,dt'. \tag{5.8}$$

PROOF: Solution to Forward Diffusion SDE

(Employ Integrating Factor Method for Solving Linear 1st-Order SDE's):

First, find solution to homogeneous version of this SDE (i.e. $\dot{\xi}(t) + \gamma\xi(t) = 0$,), which is $\xi(t) = \xi_0 e^{-\gamma t}$

Step 1: Multiply by the Integrating Factor. Let $\mu(t) = e^{\gamma t}$. Then:

$$e^{\gamma t}\frac{d\xi(t)}{dt} + \gamma e^{\gamma t}\xi(t) = e^{\gamma t}\Gamma(t),$$

which simplifies to:

$$\frac{d}{dt}\left[e^{\gamma t}\xi(t)\right] = e^{\gamma t}\Gamma(t).$$

Step 2: Integrate Both Sides. Integrating from 0 to t:

$$e^{\gamma t}\xi(t) = \xi(0) + \int_0^t e^{\gamma s}\Gamma(s)\,ds.$$

Step 3: Solve for $\xi(t)$.

$$\xi(t) = \xi(0)e^{-\gamma t} + \int_0^t e^{-\gamma(t-s)}\Gamma(s)\,ds.$$

$$\boxed{\xi(t) = \xi_0\, e^{-\gamma t} + \int_0^t e^{-\gamma(t-s)}\Gamma(s)\,ds}$$

Shortcoming of Solution to (Linear) Langevin Equation

- HOWEVER, the solution $\xi(t)$ is not consistent because it depends on a random process $\Gamma(t)$
- Hence, better to find distribution of $\xi(t)$ (when t -> infinity)

$$p(x) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{x^2}{2\sigma^2}}, \tag{5.10}$$

where $\sigma = \sqrt{\frac{q}{2\gamma}}$. In other words, the solution $\xi(t) = x$ at equilibrium is a zero-mean Gaussian random variable.

(assuming Langevin Force has white Gaussian distr.)

PROOF: Distribution of Trajectory in Forward Diffusion

Step 1: Solve the SDE

The general solution is:

$$\xi(t) = \xi_0 e^{-\gamma t} + \int_0^t e^{-\gamma(t-t')}\Gamma(t')dt'$$

As $t \to \infty$, the first term vanishes (since $e^{-\gamma t} \to 0$), leaving:

$$\xi(t) = \int_0^\infty e^{-\gamma\tau}\Gamma(t-\tau)d\tau$$

This expresses $\xi(t)$ as a **linear functional of white noise**.

PROOF (pt. 2)

Step 2: Compute the Moments of $\xi(t)$

Because $\Gamma(t)$ is zero-mean Gaussian, $\xi(t)$ is also Gaussian. So we only need its **second moment** to characterize it fully.

Use results from Risken:

- Odd moments are zero.
- Even moments (specifically the second moment) are:

$$\mathbb{E}[\xi(t)^2] = \int_0^\infty \int_0^\infty e^{-\gamma(\tau_1+\tau_2)} \mathbb{E}[\Gamma(t-\tau_1)\Gamma(t-\tau_2)] d\tau_1 d\tau_2$$

$$= q \int_0^\infty e^{-2\gamma\tau} d\tau = \frac{q}{2\gamma}$$

So $\xi(t) \sim \mathcal{N}(0, \sigma^2)$ with $\sigma^2 = \frac{q}{2\gamma}$

PROOF (pt. 3)

Step 3: Write the PDF

This is just the density of a zero-mean Gaussian with variance σ^2:

$$p(x) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{x^2}{2\sigma^2}\right) = \sqrt{\frac{\gamma}{\pi q}} \exp\left(-\frac{\gamma x^2}{q}\right)$$

Example: Distribution of Trajectory of Wiener Process

Theorem 5.4. Wiener Process. Consider the Wiener process

$$\dot{\xi} = \Gamma(t), \tag{5.15}$$

where $\Gamma(t)$ is the Gaussian white noise with $\mathbb{E}[\Gamma(t)] = 0$ and $\mathbb{E}[\Gamma(t)\Gamma(t')] = q\delta(t - t')$. The probability distribution $p(x, t)$ of the solution $\xi(t)$ where $\xi(t) = x$ is

$$p(x, t) = \frac{1}{\sqrt{2\pi qt}} e^{-\frac{(x-\xi_0)^2}{2qt}}. \tag{5.16}$$

Proof. The main difference between this result and Theorem 5.3 is that here we are interested in the distribution at any time t. To do so, we notice that $\xi(t) = \xi_0 + \int_0^t \Gamma(t')dt'$. So, to eliminate the non-zero mean, we can consider $\xi(t) - \xi_0$ instead. Substituting this into Eqn (5.13), we can show that

$$\mathbb{E}[(\xi(t) - \xi_0)^{2n+1}] = 0$$

$$\mathbb{E}[(\xi(t) - \xi_0)^{2n}] = \frac{(2n)!}{2^n n!}\left[q\int_0^t e^0 d\tau_2\right]^n = \frac{(2n)!}{2^n n!}(qt)^n.$$

This implies that the characteristic function of $\xi(t) - \xi_0$ is

$$C(u) = \sum_{n=0}^{\infty} \frac{1}{n!}\left(-\frac{u^2 qt}{2}\right)^n = e^{-\frac{u^2 qt}{2}}. \tag{5.17}$$

Taking the inverse Fourier transform will give us the probability distribution for $\xi(t)$:

$$p(x,t) = \frac{1}{\sqrt{2\pi qt}} e^{-\frac{(x-\xi_0)^2}{2qt}}. \tag{5.18}$$

Equilibrium Solution to Langevin Equation for Forward Diffusions

For simplicity, we assume a constant learning rate in the DDPM equation:

$$dx = -\frac{\beta}{2} x\, dt + \sqrt{\beta}\, dw.$$

The corresponding Langevin equation is:

$$\dot{\xi}(t) + \frac{\beta}{2}\xi(t) = \sqrt{\beta}\,\Gamma(t).$$

According to our previous theorem, as $t \to \infty$, we have:

$$p(x) = \sqrt{\frac{\gamma}{\pi q}}\, e^{-\frac{\gamma x^2}{q}} = \frac{1}{\sqrt{2\pi}}\, e^{-\frac{x^2}{2}},$$

where we substitute $\gamma = \frac{\beta}{2}$ and $q = (\sqrt{\beta})^2 = \beta$.

What about Reverse Diffusions?

- Recall the SDE for Reverse DDPM:

$$d\mathbf{x} = \underbrace{-\beta(t)\left[\frac{\mathbf{x}}{2} + \nabla_{\mathbf{x}} \log p_t(\mathbf{x})\right]}_{=\mathbf{f}(\boldsymbol{\xi},t)} dt + \underbrace{\sqrt{\beta(t)}}_{=g(t)} d\overline{\mathbf{w}}.$$

- Letting xi = x, can rewrite as

$$\dot{\boldsymbol{\xi}}(t) = \mathbf{f}(\boldsymbol{\xi}, t) + g(t)\Gamma(t), \qquad \text{where} \quad \Gamma(t) \sim \mathcal{N}(0, \mathbf{I}).$$

- Notice this isn't a Linear Langevin Equation
- It is a *NONLINEAR* Langevin equation (defined in next slide)

3.2 Nonlinear Langevin Equation for Reverse Diffusions

Nonlinear Langevin Equations (Defn.)

Definition 5.2. A **Nonlinear Langevin Equation** takes the form of

$$\dot{\xi} = h(\xi, t) + g(\xi, t)\Gamma(t), \tag{5.20}$$

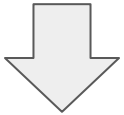
where $h(\xi, t)$ and $g(\xi, t)$ are functions denoting the drift and diffusion, respectively. Like before, we assume that $\Gamma(t)$ is a Gaussian white noise so that $\mathbb{E}[\Gamma(t)] = 0$ for all t, and $\mathbb{E}[\Gamma(t)\Gamma(t')] = 2\delta(t - t')$.

Can easily see that reverse diffusion SDE is a nonlinear Langevin Equation

Reverse Diffusion SDE: $\dot{\boldsymbol{\xi}}(t) = \mathbf{f}(\boldsymbol{\xi}, t) + g(t)\Gamma(t), \quad$ where $\quad \Gamma(t) \sim \mathcal{N}(0, \mathbf{I})$.

The Problem with Nonlinear Langevin Equations

Nonlinear Langevin Equations do NOT have simple, closed-form solutions



Need more complex mathematical tools

(Will formulate them for the rest of this presentation, in this order:)

1. Chapman-Kolmogorov Equation
2. Master's Equation
3. Kramers-Moyal Expansion
4. Fokker-Planck Equation

What is a Markov Process?

Let $t_1 \leq t_2 ... \leq t_n$ and $\xi_(t)$ be a random process that has value $x_i = \xi(t_i)$ at time t_i

Definition (Markov Property)**.** $\xi(t)$ is said to have the Markov Property if it is memoryless, i.e.

$$\mathbb{P}(x_n = \xi(t_n)|x_{n-1}, t_{n-1}, x_{n-2}, t_{n-2}, ..., x_1, t_1) = \mathbb{P}(x_n = \xi(t_n)|x_{n-1}, t_{n-1})$$

Definition (Markov Process)**.** A random process is a Markov process if it satisfies the Markov Property

3.2.1: Chapman-Kolmogorov Equation

Theorem 5.5. Chapman-Kolmogorov Equation. Let $\xi(t)$ be a Markov process, and let $x_n = \xi(t_n)$ be the state of $\xi(t)$ at time t_n. Then

$$p(x_3, t_3 \mid x_1, t_1) = \int p(x_3, t_3 | x_2, t_2) p(x_2, t_2 | x_1, t_1) dx_2, \tag{5.22}$$

assuming $t_1 \leq t_2 \leq t_3$.

- Term x2 and t2 can be marginalized with Bayes Theorem and Markov Property

Analytical Proof

Chapman–Kolmogorov Equation. For a Markov process the memoryless property gives

$$p(x_n, t_n \mid x_{n-1}, t_{n-1}, \ldots, x_1, t_1) = \underbrace{p(x_n, t_n \mid x_{n-1}, t_{n-1})}_{\text{memoryless}}.$$

Hence the joint density factorizes as

$$p(x_n, t_n, \ldots, x_1, t_1) = \prod_{i=2}^{n} p(x_i, t_i \mid x_{i-1}, t_{i-1})\, p(x_1, t_1).$$

Marginalizing over x_2 yields

$$p(x_3, t_3, x_1, t_1) = \int p(x_3, t_3, x_2, t_2, x_1, t_1)\, \mathrm{d}x_2 = \int p(x_3, t_3 \mid x_2, t_2)\, p(x_2, t_2 \mid x_1, t_1)\, p(x_1, t_1)\, \mathrm{d}x_2.$$

Writing $p(x_3, t_3, x_1, t_1) = p(x_3, t_3 \mid x_1, t_1)\, p(x_1, t_1)$ and cancelling $p(x_1, t_1)$ gives

$$p(x_3, t_3 \mid x_1, t_1) = \int p(x_3, t_3 \mid x_2, t_2)\, p(x_2, t_2 \mid x_1, t_1)\, \mathrm{d}x_2,$$

which is exactly the Chapman–Kolmogorov equation.

3.2.2: Masters Equation

Can use C-K Equation to derive another tool: The Master's Equation.

Defined as:

Theorem 5.6. Let $\xi(t)$ be a Markov process. The **Masters Equation** states that

$$\frac{\partial}{\partial t}p(x,t) = \int \Big[W(x|x')p(x',t) - W(x'|x)p(x,t)\Big]\, dx', \tag{5.25}$$

where $W(x|x')$ is the probability density function per unit time.

where

- **p(x,t)**: probability density in state x at time t
- **W(x | x′)**: the rate (per unit time) of jumping from x′ into x
- **∫W(x | x′) p(x′,t) dx′**: Gain term, total **inflow** into x.
- **∫W(x′ | x) p(x,t) dx′**: Loss term, total **outflow** from x.

Physics Interpretation of Masters Eq

- Terms W(x, t|x', t) and W(x', t|x, t) are transition rates (the transition probability per unit time)
- View p(x, t) as the density of particles in a room at time t, and
  - $\int W(x,t \mid x',t)\, p(x',t)\, dx'$: in-flow to x
  - $\int W(x',t \mid x,t)\, p(x,t)\, dx'$: out-flow from x

  The change of the density is the difference between the in-flow and the out-flow of the particles

$$\underbrace{\frac{\partial}{\partial t}p(x,t)}_{\text{rate of change}} = \underbrace{\int \Big[W(x,t \mid x',t)p(x',t)\Big]dx'}_{\text{in-flow of probability}} - \underbrace{\int \Big[W(x',t \mid x,t)p(x,t)\Big]dx'}_{\text{out-flow of probability}}.$$

- Relate time t to state x', helpful to Fokker-Planck equation

Physical Intuition Proof (Proof 1)

Intuitive Proof. Consider first a particle that can occupy only two states, x_1 or x_2, with

$$p(x_1, t) + p(x_2, t) = 1.$$

Over a short time interval $[t,\, t + \mathrm{d}t]$, each state either retains the particle or loses it to the other state. Hence

$$\begin{aligned} p(x_1, t + \mathrm{d}t) &= p(x_1, t)\, \underbrace{\big(1 - \Pr[1 \to 2]\big)}_{\text{stay in } x_1} + p(x_2, t)\, \underbrace{\Pr[2 \to 1]}_{\text{jump to } x_1} \\ &= p(x_1, t)\big(1 - W(x_2 \,|\, x_1)\, \mathrm{d}t\big) + p(x_2, t)\, W(x_1 \,|\, x_2)\, \mathrm{d}t + O(\mathrm{d}t^2), \end{aligned}$$

where

$$\Pr[1 \to 2] = W(x_2 \,|\, x_1)\, \mathrm{d}t, \qquad \Pr[2 \to 1] = W(x_1 \,|\, x_2)\, \mathrm{d}t.$$

Proof 1 cont.

Subtract $p(x_1, t)$, divide by $\mathrm{d}t$, and send $\mathrm{d}t \to 0$:

$$\frac{\mathrm{d}p(x_1,t)}{\mathrm{d}t} = -\underbrace{W(x_2\,|\,x_1)\,p(x_1,t)}_{\text{outflow rate from } x_1} + \underbrace{W(x_1\,|\,x_2)\,p(x_2,t)}_{\text{inflow rate into } x_1}.$$

Generalizing to N discrete states $\{x_j\}$, the same argument yields

$$\frac{\mathrm{d}p(x_i,t)}{\mathrm{d}t} = \sum_{j\neq i}\big[-W(x_j\,|\,x_i)\,p(x_i,t) \;+\; W(x_i\,|\,x_j)\,p(x_j,t)\big].$$

Finally, letting the index j run over a continuous variable x' turns the sum into an integral and recovers the Master Equation:

$$\frac{\partial p(x,t)}{\partial t} = \int\big[W(x\,|\,x')\,p(x',t) \;-\; W(x'\,|\,x)\,p(x,t)\big]\,\mathrm{d}x'.$$

Analytical Proof (Proof 2)

Proof. Recall the Chapman–Kolmogorov equation

$$p(x_3, t_3 \mid x_1, t_1) = \int p(x_3, t_3 \mid x_2, t_2)\, p(x_2, t_2 \mid x_1, t_1)\, \mathrm{d}x_2.$$

Set

$$(x_1, t_1) = (x_0, t_0), \quad (x_2, t_2) = (x, t), \quad (x_3, t_3) = (x, t + \Delta t),$$

so that E.q. (1) becomes

$$p(x, t + \Delta t \mid x_0, t_0) = \int p(x, t + \Delta t \mid x', t)\, p(x', t \mid x_0, t_0)\, \mathrm{d}x' \,.$$

Then

$$\begin{aligned} \frac{\partial}{\partial t} p(x, t \mid x_0, t_0) &= \lim_{\Delta t \to 0} \frac{p(x, t + \Delta t \mid x_0, t_0) \;-\; p(x, t \mid x_0, t_0)}{\Delta t} \\ &= \lim_{\Delta t \to 0} \frac{1}{\Delta t} \Big[\int p(x, t + \Delta t \mid x', t)\, p(x', t \mid x_0, t_0)\, \mathrm{d}x' \;-\; p(x, t \mid x_0, t_0) \Big]. \end{aligned}$$

Proof 2 cont.

Since

$$\int p(x', t+\Delta t \mid x, t)\,\mathrm{d}x' = 1, \quad \Longrightarrow \quad p(x, t \mid x_0, t_0) = \int p(x', t+\Delta t \mid x, t)\, p(x, t \mid x_0, t_0)\,\mathrm{d}x',$$

we may subtract this term inside the bracket to get

$$\frac{\partial}{\partial t}\, p(x, t \mid x_0, t_0) = \lim_{\Delta t \to 0} \frac{1}{\Delta t} \int \Big[p(x, t+\Delta t \mid x', t)\, p(x', t \mid x_0, t_0) \;-\; p(x', t+\Delta t \mid x, t)\, p(x, t \mid x_0, t_0) \Big] \mathrm{d}x'.$$

Define the transition rates

$$W(x, t \mid x', t) = \lim_{\Delta t \to 0} \frac{p(x, t+\Delta t \mid x', t)}{\Delta t}, \qquad W(x', t \mid x, t) = \lim_{\Delta t \to 0} \frac{p(x', t+\Delta t \mid x, t)}{\Delta t}.$$

It follows that

$$\frac{\partial}{\partial t}\, p(x, t \mid x_0, t_0) = \int \big[W(x, t \mid x', t)\, p(x', t \mid x_0, t_0) \;-\; W(x', t \mid x, t)\, p(x, t \mid x_0, t_0) \big] \mathrm{d}x'.$$

Dropping the conditioning on (x_0, t_0) yields the Master Equation

$$\frac{\partial}{\partial t}\, p(x, t) = \int \big[W(x \mid x')\, p(x', t) - W(x' \mid x)\, p(x, t) \big] \mathrm{d}x' \,.$$

3.3.3: Kramers-Moyal Expansion

Theorem 5.7. Let $\xi(t)$ be a Markov process and let $p(x,t)$ be the probability distribution of $\xi(t)$ taking a value x at time t. The **Kramers-Moyal Expansion** states that

$$\frac{\partial}{\partial t}p(x,t) = \sum_{m=1}^{\infty}\left[-\frac{\partial^m}{\partial x^m}D^{(m)}(x,t)p(x,t)\right],$$

where the Kramers-Moyal expansion coefficients are defined as

$$D^{(m)}(x,t) = \frac{1}{m!}\lim_{\Delta t\to 0}\left[\frac{1}{\Delta t}\mathbb{E}\left[(\xi(t+\Delta t)-x)^m\right]\Big|_{\xi(t)=x}\right].$$

Proof in following slides…

Proof of Kramers-Moyal Expansion

Recall the Master's Equation

$$\underbrace{\frac{\partial}{\partial t}p(x,t)}_{\text{rate of change}} = \underbrace{\int \Big[W(x,t \mid x',t)p(x',t)\Big]dx'}_{\text{in-flow of probability}} - \underbrace{\int \Big[W(x',t \mid x,t)p(x,t)\Big]dx'}_{\text{out-flow of probability}}.$$

where *W* is the probability density function per unit time, i.e.,

$$W(x,t \mid x',t) = \lim_{\Delta t \to 0} \frac{1}{\Delta t} p(x,t+\Delta t \mid x',t)$$
$$W(x',t \mid x,t) = \lim_{\Delta t \to 0} \frac{1}{\Delta t} p(x',t+\Delta t \mid x,t)$$

By conditioning on x_0 and t_0 and via substitution, we get

$$\frac{\partial}{\partial t}p(x,t \mid x_0,t_0) = \lim_{\Delta t \to 0} \frac{1}{\Delta t}\Bigg[\int p(x,t+\Delta t \mid x',t)p(x',t \mid x_0,t_0)dx' - \int p(x',t+\Delta t \mid x,t)p(x,t \mid x_0,t_0)dx'\Bigg].$$

Proof of Kramers-Moyal Expansion (cont.)

Inject a test function s.t.

$$\frac{\partial}{\partial t}\int \varphi(x)p(x,t\mid x_0,t_0)dx = \lim_{\Delta t\to 0}\frac{1}{\Delta t}\left\{\int \varphi(x)\int p(x,t+\Delta t\mid x',t)p(x',t\mid x_0,t_0)dx'dx - \int \varphi(x)\int p(x',t+\Delta t\mid x,t)p(x,t\mid x_0,t_0)dx'dx\right\}.$$

Taylor expand the test function on RHS of equality, to get

$$\begin{aligned}&\frac{\partial}{\partial t}\int \varphi(x)p(x,t\mid x_0,t_0)dx\\ &= \lim_{\Delta t\to 0}\frac{1}{\Delta t}\left[\iint \varphi(x')p(x,t+\Delta t\mid x',t)p(x',t\mid x_0,t_0)dx'dx\right.\\ &\quad + \iint \sum_{m=1}^{\infty}\frac{(x-x')^m}{m!}\frac{\partial^m}{\partial x^m}\varphi(x)\Big|_{x=x'}p(x,t+\Delta t\mid x',t)p(x',t\mid x_0,t_0)dx'dx\\ &\quad \left.- \iint \varphi(x)p(x',t+\Delta t\mid x,t)p(x,t\mid x_0,t_0)dx'dx\right]\end{aligned}$$

Note the Taylor expansion is: $$\varphi(x) = \varphi(x') + \sum_{m=1}^{\infty}\frac{(x-x')^m}{m!}\frac{\partial^m}{\partial x^m}\varphi(x)\Big|_{x=x'}$$

1st and 3rd terms are equal, $\frac{\partial}{\partial t}\int \varphi(x)p(x,t\mid x_0,t_0)dx$

So cancel them out to get: $$= \lim_{\Delta t\to 0}\frac{1}{\Delta t}\iint \sum_{m=1}^{\infty}\frac{(x-x')^m}{m!}\frac{\partial^m}{\partial x^m}\varphi(x)\Big|_{x=x'}p(x,t+\Delta t\mid x',t)p(x',t\mid x_0,t_0)dx'dx$$

Proof of Kramers-Moyal Expansion (cont.)

Define $$D^{(m)}(x',t) = \frac{1}{m!}\lim_{\Delta t\to 0}\frac{1}{\Delta t}\int (x-x')^m p(x,t+\Delta t \mid x',t)\,dx$$

Substitute into Master's, and get

$$\begin{aligned}
&\frac{\partial}{\partial t}\int \varphi(x)p(x,t\mid x_0,t_0)dx \\
&= \int \sum_{m=1}^{\infty} D^{(m)}(x',t)\frac{\partial^m}{\partial x^m}\varphi(x)\Big|_{x=x'} p(x',t\mid x_0,t_0)dx' && \text{Substitute } D^{(m)}(x',t) \\
&= \sum_{m=1}^{\infty}\int D^{(m)}(x',t)p(x',t\mid x_0,t_0)\frac{\partial^m}{\partial x^m}\varphi(x)\Big|_{x=x'} dx' && \text{Switch summation and integration}
\end{aligned}$$

Because generalized integration by parts states that for any cont. *f* and *g:*

$$\int g\cdot\frac{\partial^m f}{\partial x^m}dx = (-1)^m\int f\frac{\partial^m g}{\partial x^m}dx.$$

We further simplify the Master's, and get that expression above equals

$$\sum_{m=1}^{\infty}(-1)^m\int \varphi(x')\frac{\partial^m}{\partial x^m}\Big[D^{(m)}(x,t)p(x,t,x_0,t_0)\Big]dx'$$

Proof of Kramers-Moyal Expansion (cont.)

Note that our result so far:

$$\frac{\partial}{\partial t}\int \varphi(x)p(x,t \mid x_0,t_0)dx \;=\; \sum_{m=1}^{\infty}(-1)^m \int \varphi(x')\frac{\partial^m}{\partial x^m}\Big[D^{(m)}(x,t)p(x,t,x_0,t_0)\Big]dx'$$

holds for any test function phi, so this implies

$$\frac{\partial}{\partial t}p(x,t \mid x_0,t_0) = \sum_{m=1}^{\infty}(-1)^m \frac{\partial^m}{\partial x^m}\Big[D^{(m)}(x,t)p(x,t,x_0,t_0)\Big]$$

Drop condition on initial states to get

$$\frac{\partial}{\partial t}p(x,t) = \sum_{m=1}^{\infty}\frac{1}{m!}(-1)^m \frac{\partial^m}{\partial x^m}\Big[D^{(m)}(x,t)p(x,t)\Big]$$

END OF PROOF.

Approximation of Kramers-Moyal Expansion

- Notice: K-M expansion creates infinite series
- Can approximate this by retaining terms up to order 1 or 2 ONLY
- Else, cannot approximate -
- This is Pawula's Theorem

Pawula's Theorem: The Kramers-Moyal Expansion has m = 1, 2, or infinity.

3.3.4: Fokker-Planck (FP) Equation (Defn.)

The Fokker-Planck Equation is just KM expansion using m=2…

Definition 5.4. The **Fokker-Planck Equation** is obtained by truncating the Kramers-Moyal expansion to $m = 2$. That is, for any Markov process $\xi(t)$, the probability distribution $p(x,t)$ of $\xi(t) = x$ at time t will satisfy the following partial differential equation:

$$\frac{\partial}{\partial t}p(x,t) = -\frac{\partial}{\partial x}D^{(1)}(x,t)p(x,t) + \frac{\partial^2}{\partial x^2}D^{(2)}(x,t)p(x,t). \tag{5.35}$$

Where recall:

$$D^{(m)}(x',t) = \frac{1}{m!}\lim_{\Delta t \to 0}\frac{1}{\Delta t}\int (x-x')^m p(x, t+\Delta t \mid x', t)\, dx$$

FP Equation Motivation

Define:

p is a probability density on $\mathbb{R}^d$: goal of gen. modelling is twofold:
given samples $x^1, ..., x^n$ from p, we want to estimate and generate new samples from p

OU and SDE representation,having the same marginals as p_t are not necessarily unique or special

What matters are two features:

1. p_t provides a path connecting p and $p_T \sim N(0, I)$
2. marginals are easy to sample

FP Equation: PDE to SDE

Consider the SDE

$$dX_t = f(X_t, t)\,dt + g(X_t, t)\,dW_t$$

With coefficients

- drift $f : \mathbb{R}^d \times \mathbb{R} \to \mathbb{R}^d$
- diffusion coefficient $g : \mathbb{R}^d \times \mathbb{R} \to \mathbb{R}^{d \times d}$

- Initial distribution: $X_0 \sim \mu$
- Let $p_t(x)$ denote the density of X_t, i.e. $X_t \sim p_t$

FP Motivation: PDE to SDE Cont.

Key is that all p_t satisfy the Fokker-Planck Equation:

- Describes how $p_t(x)$ evolves over time Time evolution of FP
- The change in probability density is expressed entirely through spatial derivatives

$$\frac{\partial p_t(x)}{\partial t} = -\sum_{i=1}^{d} \frac{\partial}{\partial x_i}\left[f_i(x,t)\, p_t(x,t)\right] + \tfrac{1}{2}\sum_{i=1}^{d}\sum_{j=1}^{d} \frac{\partial^2}{\partial x_i\, \partial x_j}\left[\left(g(x,t)\, g(x,t)^\top\right)_{ij} p_t(x,t)\right]$$

$$\frac{\partial p_t(x)}{\partial t} = -\nabla\cdot\left[f(x,t)\, p_t(x,t)\right] + \tfrac{1}{2}\,\nabla^2\cdot\left[g(x,t)\, g(x,t)^\top\, p_t(x,t)\right]$$

∇: gradient $\rho : \mathbb{R}^d \to \mathbb{R}^d$
$\nabla \cdot \rho(x)$: divergence (measure of contraction or expansion) $\sum_{i=1}^{d} \partial_{x_i} \rho(x_1, ..., x_d)$
$\nabla \cdot \nabla = \nabla^2 = \sum_{i=1}^{d} \partial_{x_i}^2$: Laplacian

Transition: Kramers–Moyal to SDE Coefficients

- Kramers–Moyal expansion (truncate at $m = 2$) gives

$$\partial_t p(x,t) = -\partial_x \left[D^{(1)}(x,t)\, p(x,t) \right] + \partial_x^2 \left[D^{(2)}(x,t)\, p(x,t) \right]$$

- Here

$$D^{(m)}(x',t) = \frac{1}{m!} \lim_{\Delta t \to 0} \frac{1}{\Delta t} \int (x - x')^m \, p(x, t + \Delta t \mid x', t)\, dx$$

- For the Itô SDE

$$dX_t = f(X_t, t)\, dt + g(X_t, t)\, dW_t$$

  one identifies

$$\boxed{D^{(1)}(x,t) = f(x,t) \quad , \quad D^{(2)}(x,t) = \tfrac{1}{2}\, g(x,t)\, g(x,t)^\top}$$

- Substituting into the Fokker–Planck equation yields

$$\partial_t p(x,t) = -\nabla \cdot \left[f(x,t)\, p(x,t) \right] + \tfrac{1}{2} \nabla^2 \cdot \left[g(x,t)\, g(x,t)^\top \, p(x,t) \right]$$

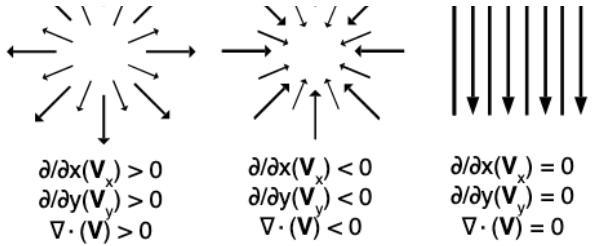
FP equation: Left-hand side intuition

Interpretation

- If probability mass is flowing *into* point x, then $\frac{d}{dt}p_t(x) > 0$
- If probability mass is flowing *out of* point x, then $\frac{d}{dt}p_t(x) < 0$
- Intuitively:

$$\frac{d}{dt}p_t(x) = \text{“Net inflow of change of probability mass to point } x\text{”}$$

- This net inflow is called the **divergence** in physics



Rewriting FP into a continuity equation

$$\begin{aligned}\frac{d}{dt}p_t(x) &= \text{"Net inflow of change of probability mass to point } x\text{"} \\ &= -\text{"Net outflow of change of probability mass to point } x\text{"} \\ &= -\operatorname{div}(\text{"Change of probability mass at point } x\text{"}) \\ &= -\nabla\cdot(\text{"Change of probability mass at point } x\text{"})\end{aligned}$$

=Probability current in Stanley's Notes

- This form naturally leads to a continuity equation

$$\frac{d}{dt}p_t(x) = -\nabla\cdot\Big[\underbrace{f(x,t)\,p_t(x)}_{\text{drift}} + \underbrace{\tfrac{1}{2}\,\nabla^T\big(g(x,t)\,g(x,t)^T\,p_t(x)\big)}_{\text{diffusion}}\Big]$$

- This is a form of the continuity equation from fluid dynamics
- Incompressibility here means the probability mass can shift/transport but not vanish (can transport)
- The two components inside the divergence describe two sources of change

Drift Component of FP Equation

Drift component

- The vector field $f(x, t)$ moves mass in the direction of the drift
- If $p_t(x) = 0$, no change occurs because there's no mass to move
- So the actual mass flux is $f(x, t)p_t(x)$

"Change of probability mass at x by drift" $= f(x, t)p_t(x)$

Diffusion Component of FP Equation

Diffusion component

- The diffusion matrix $g(x,t)$ perturbs positions with noise
- For a small step s, the displacement is $\sqrt{s}\, g(x,t)\, \xi \sim \mathcal{N}(0,\, s\, g\, g^T)$
- Probability mass dispersed at x is proportional to $p_t(x)$ and the rate of dispersion $g(x,t)\, g(x,t)^T$
- Initial rate of dispersion $\propto p_t(x)\, g(x,t)\, g(x,t)^T$
- Dispersion alone does not change net mass—neighbouring points disperse similarly
- So we compute the **change in dispersion rate**:

$$\text{“Change of probability mass at } x \text{ by diffusion”} = \tfrac{1}{2}\, \nabla \cdot \left[g(x,t)\, g(x,t)^T\, p_t(x) \right]$$

Summary

In essence, the Fokker–Planck equation describes the change of probability mass at x over time as the net inflow of change of probability mass at x, governed by a vector field induced by the SDE

FP application: Turning SDEs into ODEs

- Consider the SDE:

$$dX_t = f(X_t, t)\,dt + g(t)\,dW_t$$

- For simplicity, let g be independent of position (depends only on time)
- A natural question arises:

Can we reproduce the same evolution using an ODE instead of an SDE, as long as we start from the same initial distribution?

Turning SDEs into ODEs pt. 2

- More precisely, if $X_0 \sim p_0$, **can we find an ODE of the form**:

$$dZ_t = F(Z_t, t)\, dt$$

- such that $Z_t \sim X_t$ for all t, i.e., both share the same marginal distributions
- let $X_t \sim p_t$ and assume $Z_0 \sim p_0$
- then compute the Fokker–Planck for X_t:

$$\frac{d}{dt}p_t(x) = -\sum_{i=1}^{n} \frac{\partial}{\partial x_i}[\, f_i(x,t)\, p_t(x)\,] + \tfrac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} \frac{\partial^2}{\partial x_i \partial x_j}\left[(g(t)g(t)^\top)_{ij}\, p_t(x)\right]$$

- use linearity of derivatives, as g is independent of x
- can be represented as $\nabla_x^2(\, g(t)g(t)^\top p_t(x) = \nabla_x\left(g(t)g(t)^\top \nabla_x p_t(x)\right)$

$$= -\sum_{i=1}^{n} \frac{\partial}{\partial x_i}[\, f_i(x,t)\, p_t(x)\,] + \tfrac{1}{2}\sum_{i=1}^{n} \frac{\partial}{\partial x_i}\left[\sum_{j=1}^{n}(g(t)g(t)^\top)_{ij}\frac{\partial}{\partial x_j}p_t(x)\right]$$

Replace this term by a score

ASIDE: Gradient–Logarithm Identity

- Show that

$$\nabla p_t(x) = p_t(x)\, \nabla \log p_t(x)$$

Derivation

- work component-wise for $i = 1, \ldots, d$

$$\frac{\partial}{\partial x_i} \log p_t(x) = \frac{1}{p_t(x)} \frac{\partial p_t(x)}{\partial x_i}$$

- multiply both sides by $p_t(x)$

$$p_t(x) \frac{\partial}{\partial x_i} \log p_t(x) = \frac{\partial p_t(x)}{\partial x_i}$$

- collect components into the gradient vector

$$\nabla p_t(x) = \left(\tfrac{\partial p_t}{\partial x_1}, \ldots, \tfrac{\partial p_t}{\partial x_d}\right) = p_t(x) \left(\tfrac{\partial}{\partial x_1} \log p_t, \ldots, \tfrac{\partial}{\partial x_d} \log p_t\right) = p_t(x)\, \nabla \log p_t(x)$$

Turning SDEs into ODEs pt. 3

- Rewrite the second term using $\nabla p_t = p_t \, \nabla \log p_t$

$$= -\sum_{i=1}^{n} \frac{\partial}{\partial x_i} \left[f_i(x,t)\, p_t(x) \right] + \tfrac{1}{2} \sum_{i=1}^{n} \frac{\partial}{\partial x_i} \left[\sum_{j=1}^{n} (g(t) g(t)^\top)_{ij} \left(\tfrac{\partial}{\partial x_j} \log p_t(x) \right) p_t(x) \right]$$

$$= -\sum_{i=1}^{n} \frac{\partial}{\partial x_i} \left[\left[f_i(x,t) - \tfrac{1}{2} \sum_{j=1}^{n} (g(t) g(t)^\top)_{ij} \left[\tfrac{\partial}{\partial x_j} \log p_t(x) \right] \right] p_t(x) \right]$$

- Define the drift F such that the ODE reproduces the same marginals:

$$F(x,t) = f(x,t) \; - \; \tfrac{1}{2}\, g(t)\, g(t)^\top \, \nabla \log p_t(x) \qquad (\text{SDE} \; \Rightarrow \; \text{ODE drift})$$

Turning SDEs into ODEs pt. 4

- Substituting F back gives

$$\frac{d}{dt}p_t(x) = -\nabla \cdot \left[F(x,t)\, p_t(x)\right]$$

- This is exactly the continuity equation for the flow field F
- Therefore the deterministic ODE

$$dZ_t = F(Z_t, t)\, dt$$

has solution Z_t whose marginal density at time t is p_t ($Z_t \sim p_t$), matching the SDE's marginals

We have just shown that we can turn an SDE into a ODE using the FP Equation!

Application: Deterministic sampling process in Diffusion models

- In diffusion models, sampling proceeds by running the *reverse-time SDE*

$$d\bar{X}_t = \left[f(\bar{X}_t, t) \; - \; g^2(t) \, \nabla \log p_t(\bar{X}_t) \right] dt \; + \; g(t) \, d\bar{W}_t$$

- time runs from $t = T$ back to $t = 0$
- f and g are the forward SDE coefficients
- $\nabla \log p_t$ (the score) is learned during training
- to obtain a deterministic sampler, convert the SDE into an ODE with identical marginals

$$d\bar{X}_t = \left[f(\bar{X}_t, t) \; - \; \tfrac{1}{2} \, g^2(t) \, \nabla \log p_t(\bar{X}_t) \right] dt$$

- the $\frac{1}{2} \, g^2 \, \nabla \log p_t$ term arises from time reversal in the Fokker–Planck/ODE conversion
- this deterministic ODE is called the *probability-flow ODE*
- integrating this ODE from $T \to 0$ yields samples with the same final distribution as the reverse SDE

Reverse SDE Proof:

Identical proof before but with different coefficients

Continuity-Equation Trick on Reverse-Time SDE

- Start with the reverse-time SDE's Fokker–Planck equation

$$\partial_t p_t(x) = -\nabla \cdot \left[\left(f(x,t) \; - \; g^2(t)\, \nabla \log p_t(x) \right) p_t(x) \right] \; + \; \tfrac{1}{2}\, \nabla^2 \cdot \left(g^2(t)\, p_t(x) \right)$$

- Rewrite the Laplacian term as a divergence of a gradient

$$\nabla^2 \cdot \left(g^2\, p_t \right) = \nabla \cdot \left[g^2\, \nabla p_t \right]$$

- Apply the score identity $\nabla p_t = p_t\, \nabla \log p_t$

$$\nabla \cdot \left[g^2\, \nabla p_t \right] = \nabla \cdot \left[g^2 \left(p_t\, \nabla \log p_t \right) \right]$$

Reverse SDE Proof 2

- Substitute back into the PDE

$$\partial_t p_t = -\nabla \cdot \left[(f - g^2 \nabla \log p_t)\, p_t \right] \; + \; \tfrac{1}{2}\, \nabla \cdot \left[g^2\, p_t\, \nabla \log p_t \right]$$

- Combine into a single divergence

$$\partial_t p_t = -\nabla \cdot \left[(f - g^2 \nabla \log p_t)\, p_t \; - \; \tfrac{1}{2}\, g^2\, p_t\, \nabla \log p_t \right] = -\nabla \cdot \left[F(x,t)\, p_t(x) \right]$$

$$F(x,t) = f(x,t) \; - \; \tfrac{1}{2}\, g^2(t)\, \nabla \log p_t(x)$$

- Therefore the deterministic ODE

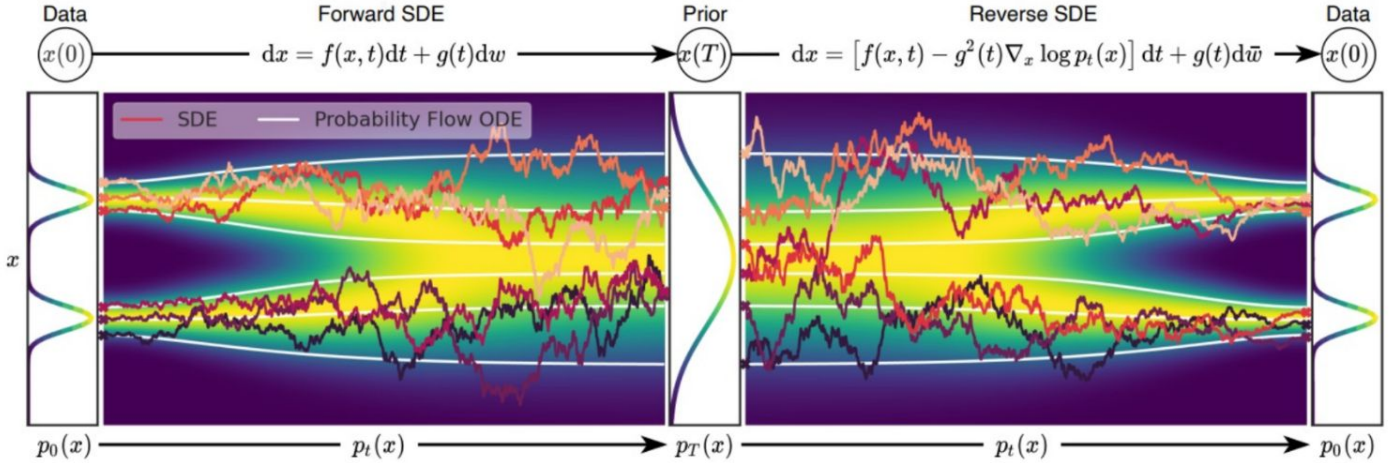
$$dZ_t = F(Z_t, t)\, dt$$

has solution Z_t whose marginals coincide with those of the reverse-time SDE

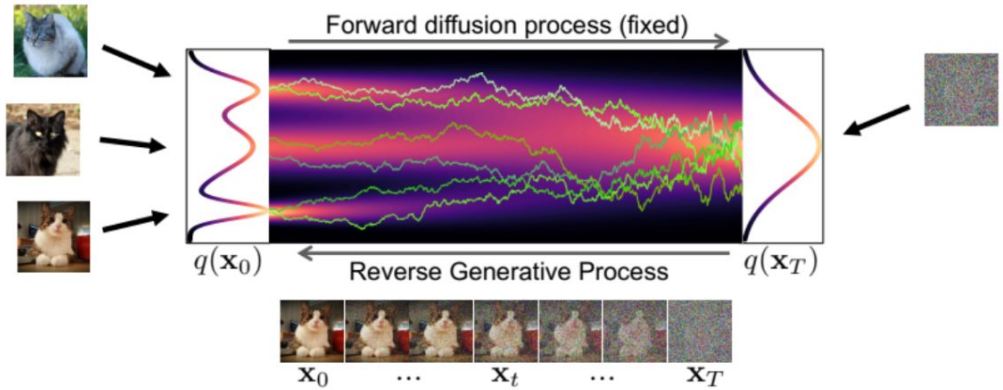
Probability Flow ODE

A SDE has a corresponding ODE (no Wiener process) that has same marginal distribution at every time step.

- Red curves: SDE
- White curves: ODE



Probability Flow ODE



- Consider reverse generative diffusion SDE:

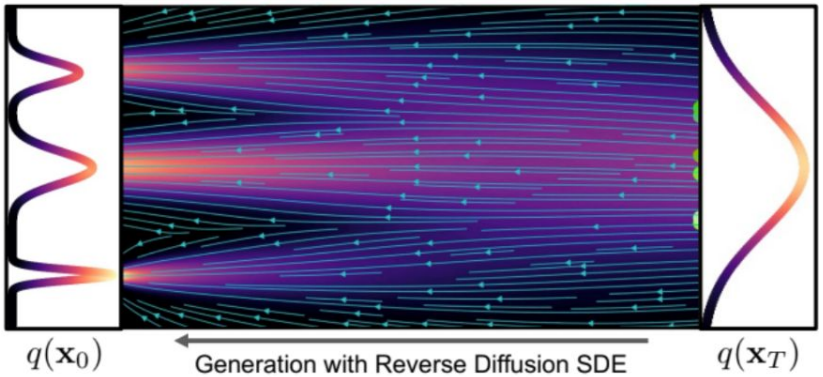
$$\mathrm{d}\mathbf{x}_t = -\frac{1}{2}\beta(t)\left[\mathbf{x}_t + 2\nabla_{\mathbf{x}_t}\log q_t(\mathbf{x}_t)\right]\mathrm{d}t + \sqrt{\beta(t)}\,\mathrm{d}\bar{\boldsymbol{\omega}}_t$$

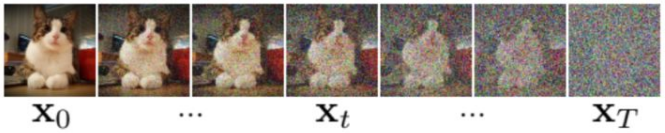
- In distribution equivalent to **"Probability Flow ODE"**:

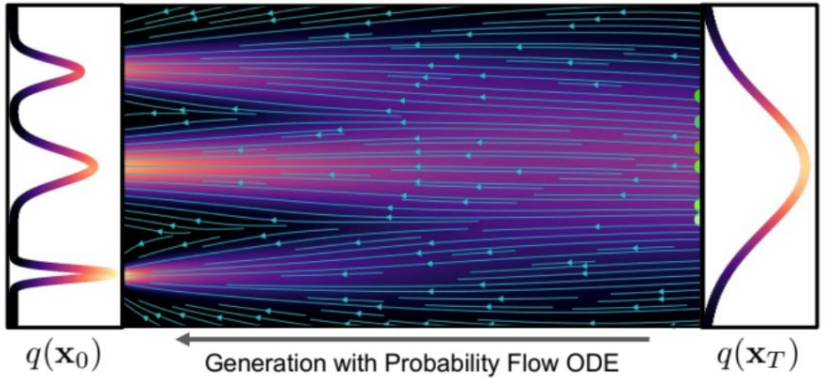
$$\mathrm{d}\mathbf{x}_t = -\frac{1}{2}\beta(t)\left[\mathbf{x}_t + \nabla_{\mathbf{x}_t}\log q_t(\mathbf{x}_t)\right]\mathrm{d}t$$

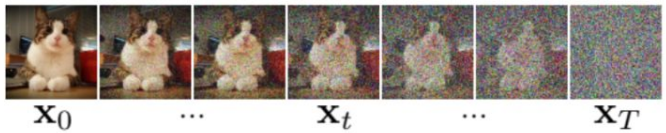
(solving this ODE results in the same $q_t(\mathbf{x}_t)$ when initializing $q_T(\mathbf{x}_T) \approx \mathcal{N}(\mathbf{x}_T; \mathbf{0}, \mathbf{I})$)

Synthesis with SDE vs. ODE









- **Generative Reverse Diffusion SDE (stochastic):**

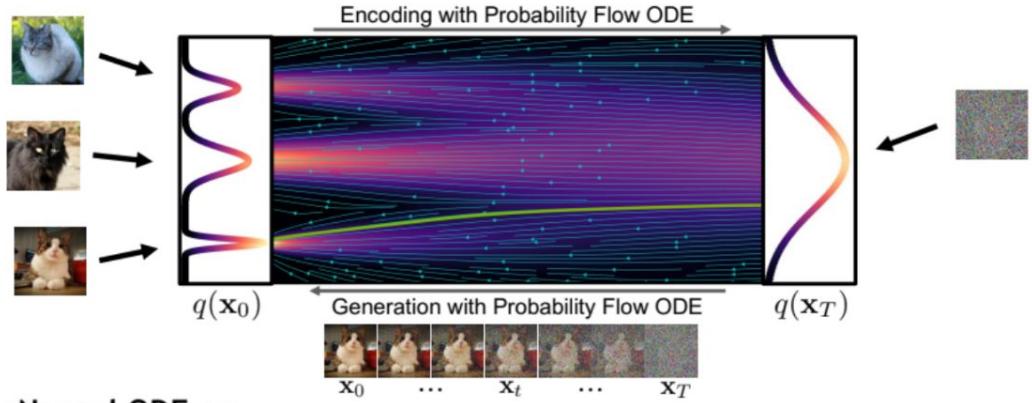
$$\mathrm{d}\mathbf{x}_t = -\frac{1}{2}\beta(t)\left[\mathbf{x}_t + 2\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}_t, t)\right]\mathrm{d}t + \sqrt{\beta(t)}\,\mathrm{d}\bar{\boldsymbol{\omega}}_t$$

- **Generative Probability Flow ODE (deterministic):**

$$\mathrm{d}\mathbf{x}_t = -\frac{1}{2}\beta(t)\left[\mathbf{x}_t + \mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}_t, t)\right]\mathrm{d}t$$

Probability Flow ODE

Diffusion Models as Continuous Normalizing Flows



- **Probability Flow ODE as Neural ODE or Continuous Normalizing Flow (CNF):**

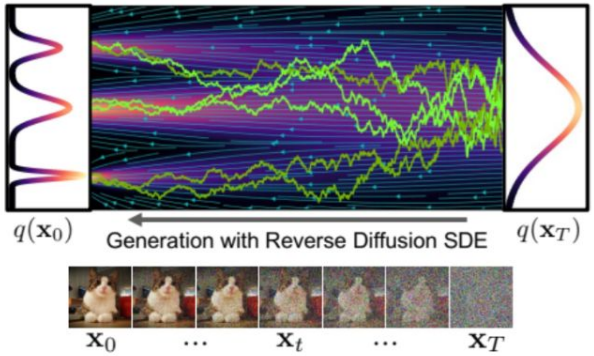
$$\mathrm{d}\mathbf{x}_t = -\frac{1}{2}\beta(t)\left[\mathbf{x}_t + \mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}_t, t)\right]\mathrm{d}t$$

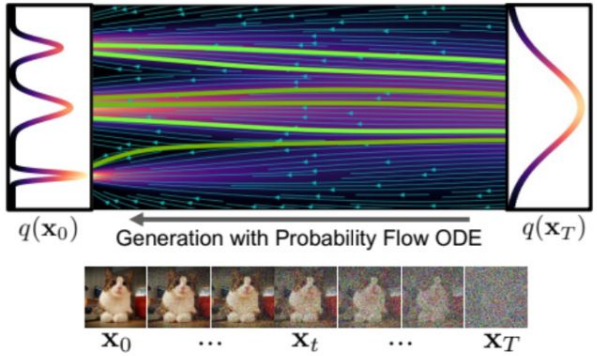
$$\left(\frac{\mathrm{d}\mathbf{x}_t}{\mathrm{d}t} = -\frac{1}{2}\beta(t)\left[\mathbf{x}_t + \mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}_t, t)\right]\right)$$

- ➡ Enables use of **advanced ODE solvers**
- ➡ **Deterministic encoding and generation** (semantic image interpolation, etc.)

Sampling from "Continuous-Time" Diffusion Models

How to solve the generative SDE or ODE in practice?





Generative Diffusion SDE:

$$\mathrm{d}\mathbf{x}_t = -\frac{1}{2}\beta(t)\left[\mathbf{x}_t + 2\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}_t, t)\right]\mathrm{d}t + \sqrt{\beta(t)}\,\mathrm{d}\bar{\boldsymbol{\omega}}_t$$

- *Euler-Maruyama*:

$$\mathbf{x}_{t-1} = \mathbf{x}_t + \frac{1}{2}\beta(t)\left[\mathbf{x}_t + 2\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}_t, t)\right]\Delta t + \sqrt{\beta(t)\Delta t}\,\mathcal{N}(\mathbf{0}, \mathbf{I})$$

- *Ancestral Sampling* (Part 1) is also a generative SDE sampler!

Probability Flow ODE:

$$\mathrm{d}\mathbf{x}_t = -\frac{1}{2}\beta(t)\left[\mathbf{x}_t + \mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}_t, t)\right]\mathrm{d}t$$

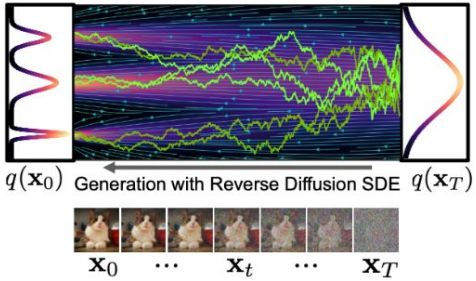
- *Euler's Method*:

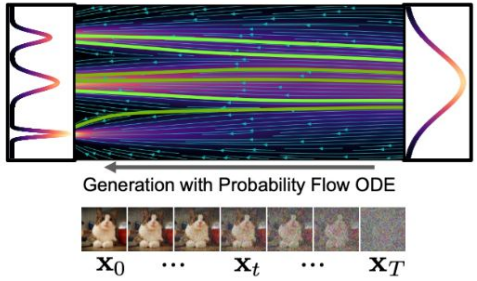
$$\mathbf{x}_{t-1} = \mathbf{x}_t + \frac{1}{2}\beta(t)\left[\mathbf{x}_t + \mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}_t, t)\right]\Delta t$$

- In practice: Higher-Order ODE solvers (*Runge-Kutta, linear multistep methods, exponential integrators, …*)

Sampling from "Continuous-Time" Diffusion Models

SDE vs. ODE Sampling: Pro's and Con's





Generative Diffusion SDE:

$$\mathrm{d}\mathbf{x}_t = -\frac{1}{2}\beta(t)\left[\mathbf{x}_t + 2\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}_t, t)\right]\mathrm{d}t + \sqrt{\beta(t)}\,\mathrm{d}\bar{\boldsymbol{\omega}}_t$$

$$\mathrm{d}\mathbf{x}_t = \underbrace{-\frac{1}{2}\beta(t)\left[\mathbf{x}_t + \mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}_t, t)\right]\mathrm{d}t}_{\text{Probability Flow ODE}} \underbrace{-\frac{1}{2}\beta(t)\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}_t, t)\mathrm{d}t + \sqrt{\beta(t)}\,\mathrm{d}\bar{\boldsymbol{\omega}}_t}_{\text{Langevin Diffusion SDE}}$$

- *Pro:* Continuous noise injection can help to compensate errors during diffusion process (Langevin sampling actively pushes towards correct distribution).
- *Con:* Often slower, because the stochastic terms themselves require fine discretization during solve.

Probability Flow ODE:

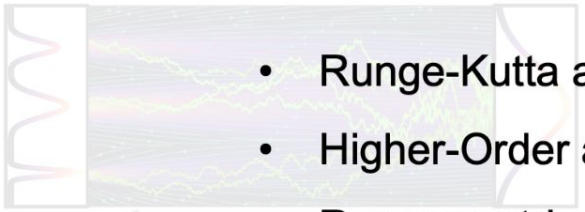
$$\mathrm{d}\mathbf{x}_t = -\frac{1}{2}\beta(t)\left[\mathbf{x}_t + \mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}_t, t)\right]\mathrm{d}t$$

- *Pro:* Can leverage fast ODE solvers. Best when targeting very fast sampling.
- *Con:* No "stochastic" error correction, often slightly lower performance than stochastic sampling.

Karras et al., "Elucidating the Design Space of Diffusion-Based Generative Models", *arXiv*, 2022

Sampling from "Continuous-Time" Diffusion Models

How to solve the generative SDE or ODE in practice?



- Runge-Kutta adaptive step-size ODE solver [1]
- Higher-Order adaptive step-size SDE solver [2]
- Reparametrized, smoother ODE [3]
- Higher-Order ODE solver with linear multistepping [4]
- Exponential ODE Integrators [5,6]
- Higher-Order ODE solver with Heun's Method [7]

[1] Song et al., "Score-Based Generative Modeling through Stochastic Differential Equations", *ICLR*, 2021
[2] Jolicoeur-Martineau et al., "Gotta Go Fast When Generating Data with Score-Based Models", *arXiv*, 2021
[3] Song et al., "Denoising Diffusion Implicit Models", *ICLR*, 2021
[4] Liu et al., "Pseudo Numerical Methods for Diffusion Models on Manifolds", *ICLR*, 2022
[5] Zhang and Chen, "Fast Sampling of Diffusion Models with Exponential Integrator", *arXiv*, 2022
[6] Lu et al., "DPM-Solver: A Fast ODE Solver for Diffusion Probabilistic Model Sampling in Around 10 Steps", *arXiv*, 2022
[7] Karras et al., "Elucidating the Design Space of Diffusion-Based Generative Models", *arXiv*, 2022

Summary

1. The forward diffusion process can be described by an SDE in the continuous domain.
2. The reverse process of an SDE can also be formulated as another SDE.
3. A diffusion process SDE has a corresponding ODE that has the same marginal distribution at each time step.
4. Hence, the reverse diffusion (generation) process can be conducted by solving the corresponding ODE.

Numerical Methods for ODE

- General-purpose ODE solvers require many steps for accurate results.

- Is there a dedicated solver for the following diffusion model ODE that can solve it accurately in just a few steps?

$$\frac{dx(t)}{dt} = \left(\frac{d\log\alpha_t}{dt}\right)x(t) - \sigma_t \frac{d\lambda_t}{dt}\hat{\boldsymbol{\varepsilon}}_\theta(\boldsymbol{x}_t, t)$$

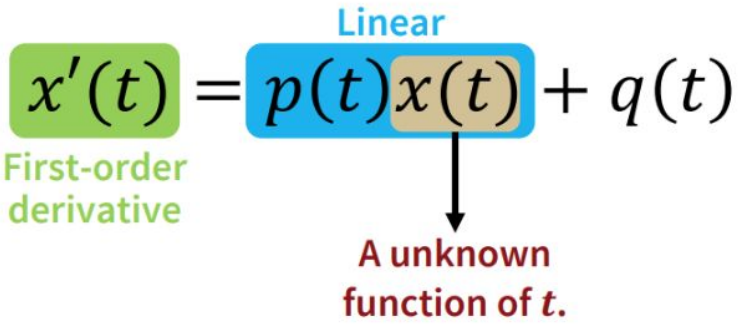
How to solve these ODEs?

With DPM Solver- DDIM Is a first order approx of DPM Solver

Ordinary Differential Equation

First-Order Linear Differential Equation

First-order linear differential equation has the following general form:



Note that “linear” does NOT mean that $p(t)$ is linear, but the equation is linear in x and its derivative with respect to t.

First-Order Linear Differential Equation

Let's first think about a special when $q(t) = 0$.

$$x'(t) = p(t)x(t)$$

How to find a function $x(t)$ that satisfies this equation?

First-Order Linear Differential Equation

When assuming $x(t) \neq 0$ for all t,

$$\frac{1}{x(t)} x'(t) = p(t)$$

What is the simplified form of the expression $\frac{1}{x(t)} x'(t)$ based on calculus techniques?

First-Order Linear Differential Equation

When assuming $x(t) \neq 0$ for all t,

$$\frac{1}{x(t)} x'(t) = \frac{d \ln x(t)}{dt} = p(t)$$

Integrate both sides of the equation, and then take the exponential.

First-Order Linear Differential Equation

$$\ln x(t) = \ln x(t_0) + \int_{t_0}^{t} p(\tau)d\tau$$

$$\Longrightarrow x(t) = e^{\ln x(t_0) + \int_{t_0}^{t} p(\tau)d\tau}$$

$$= e^{\ln x(t_0)} \cdot e^{\int_{t_0}^{t} p(\tau)d\tau}$$

$$= C \cdot e^{\int p(\tau)d\tau}$$

First-Order Linear Differential Equation

Q. Find a function $x(t)$ that satisfies the following ordinary differential equation:

$$x'(t) = -\frac{1}{2}x(t)$$

and the boundary condition:

$$x(0) = 1$$

First-Order Linear Differential Equation

Based on the solution for the special case:

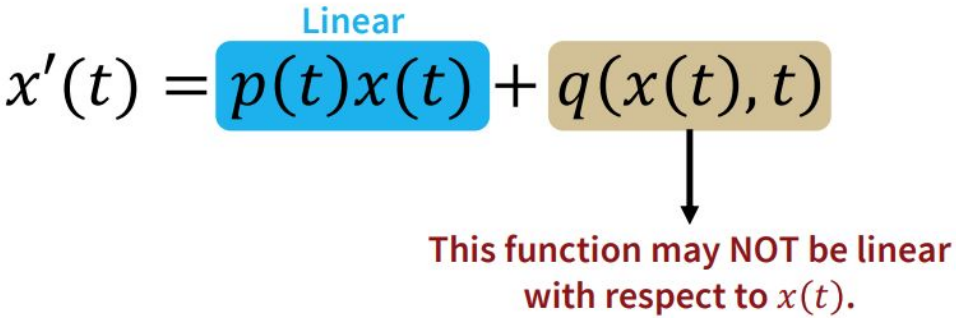
$$x'(t) = p(t)x(t)$$

can we find the solution for the general case:

$$x'(t) = p(t)x(t) + q(t)$$

First-Order Semilinear Differential Equation

Let's think about an even more general case:



Such a function is called a first-order semilinear ordinary differential equation.

First-Order Semlinear Differential Equation

How to solve it?

Recall that the solution of the following ODE:

$$x'(t) = p(t)x(t)$$

is given by:

$$x(t) = C \cdot e^{\int p(\tau)d\tau}$$

First-Order Semlinear Differential Equation

Exponential Integrator

Given first-order semlinear ODE:

$$x'(t) - p(t)x(t) = q(x(t), t)$$

Inspired by the solution of the special case, multiply $e^{-\int_{t_0}^{t} p(r)dr}$ both sides:

$$e^{-\int_{t_0}^{t} p(r)dr} x'(t) - e^{-\int_{t_0}^{t} p(r)dr} p(t)x(t) = e^{-\int_{t_0}^{t} p(r)dr} q(x(t), t)$$

First-Order Semlinear Differential Equation

$$e^{-\int_{t_0}^{t} p(r)dr} x'(t) - e^{-\int_{t_0}^{t} p(r)dr} p(t)x(t) = e^{-\int_{t_0}^{t} p(r)dr} q(x(t), t)$$

Note that $\frac{d}{dt}\left(e^{-\int_{t_0}^{t} p(r)dr}\right) = -p(t).$

Simplify the left side.

Hint. $\frac{d}{dt}(f(t)g(t)) = f'(t)g(t) + f(t)g'(t)$

First-Order Semlinear Differential Equation

$$e^{-\int_{t_0}^{t} p(r)dr} x'(t) - e^{-\int_{t_0}^{t} p(r)dr} p(t)x(t)$$

$$= \frac{d}{dt}\left(e^{-\int_{t_0}^{t} p(r)dr} x(t)\right) = e^{-\int_{t_0}^{t} p(r)dr} q(x(t), t)$$

Integrate both sides.

Hint. $\frac{d}{dt}\big(p(t)\big) = q(s) \;\Rightarrow\; p(t) = p(s) + \int_s^t q(\tau)\,\mathrm{d}\tau$

First-Order Semlinear Differential Equation

$$\frac{d}{dt}\left(e^{-\int_{t_0}^{t} p(r)dr}x(t)\right) = e^{-\int_{t_0}^{t} p(r)dr}q(x(t),t)$$

$$\Rightarrow e^{-\int_{t_0}^{t} p(r)dr}x(t) = e^{-\int_{t_0}^{s} p(r)dr}x(s) + \int_{s}^{t} e^{-\int_{t_0}^{\tau} p(r)dr}q(x(\tau),\tau)d\tau$$

$t < s$

$$\Rightarrow x(t) = e^{\int_{t_0}^{t} p(r)dr - \int_{t_0}^{s} p(r)dr}x(s) + \int_{s}^{t} e^{\int_{t_0}^{t} p(r)dr - \int_{t_0}^{\tau} p(r)dr}q(x(\tau),\tau)d\tau$$

First-Order Semlinear Differential Equation

$$x(t) = e^{\int_{t_0}^{t} p(r)dr - \int_{t_0}^{s} p(r)dr} x(s) + \int_{s}^{t} e^{\int_{t_0}^{t} p(r)dr - \int_{t_0}^{\tau} p(r)dr} q(x(\tau), \tau)d\tau$$

$$x(t) = e^{\int_{s}^{t} p(r)dr} x(s) + \int_{s}^{t} e^{\int_{\tau}^{t} p(r)dr} q(x(\tau), \tau)d\tau$$

First-Order Linear Differential Equation

Q. Find a function $x(t)$ that satisfies the following ordinary differential equation:

$$x'(t) = -\frac{1}{2}x(t) + e^{-\frac{1}{2}t}$$

and the boundary condition:

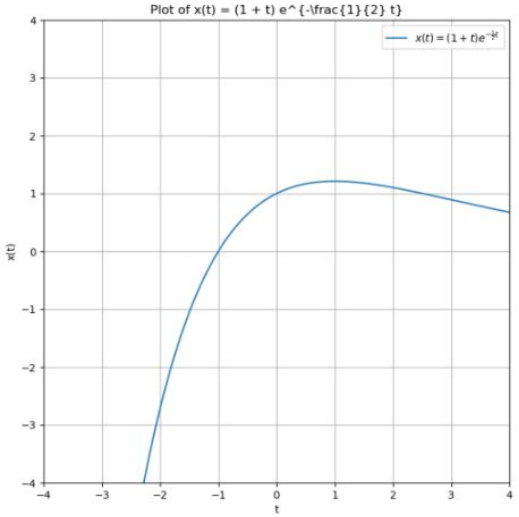
$$x(0) = 1$$

First-Order Linear Differential Equation

A. Let $s = 0$. $x(s) = 1$.

$$x(t) = e^{-\frac{1}{2}t} + \int_s^t e^{-\frac{1}{2}(t-\tau)} e^{-\frac{1}{2}\tau} d\tau$$

$$= (1 + t)e^{-\frac{1}{2}t}$$



DPM-Solver

Lu et al., DPM-Solver: A Fast ODE Solver for Diffusion Probabilistic Model Sampling in Around 10 Steps, NeurIPS 2022.

Probability Flow ODE

Reverse process **SDE**:

$$dx(t) = \left[f(t)x(t) + \frac{g^2(t)}{\sqrt{1-\bar{\alpha}_t}} \hat{\boldsymbol{\varepsilon}}_\theta(x(t), t) \right] dt + g(t) d\bar{\boldsymbol{w}}_t$$

σ_t

The corresponding probabiltiy flow **ODE**:

$$\frac{dx(t)}{dt} = f(t)x(t) + \frac{g^2(t)}{2\sigma_t} \hat{\boldsymbol{\varepsilon}}_\theta(x(t), t)$$

Probability Flow ODE

Plug $f(t) = \frac{d \log \alpha_t}{dt}$ and $g^2(t) = 2\sigma_t^2 \left(-\frac{d\lambda_t}{dt}\right)$ for the variance preserving case:

$$\frac{dx(t)}{dt} = \left(\frac{d \log \alpha_t}{dt}\right) x(t) - \sigma_t \frac{d\lambda_t}{dt} \hat{\boldsymbol{\varepsilon}}_\theta(\boldsymbol{x}_t, t)$$

(probability flow ODE)

Coming Back to PF-ODE…

The reverse process ODE is a first-order semlinear ODE:

$$x'(t) = \underbrace{\left(\frac{d \log \alpha_t}{dt}\right)}_{p(t)} x(t) \underbrace{- \sigma_t \frac{d\lambda_t}{dt} \hat{\boldsymbol{\varepsilon}}_\theta(x(t), t)}_{q(x(t),t)}$$

DPM-Solver

$$x(t) = e^{\int_s^t p(r)dr} x(s) + \int_s^t e^{\int_\tau^t p(r)dr} q(x(\tau), \tau) d\tau$$

Plug $p(t) = \frac{d \log \alpha_t}{dt}$ and $q(x(t), t) = -\sigma_t \frac{d\lambda_t}{dt} \hat{\boldsymbol{\varepsilon}}_\theta(x(t), t)$:

$$x(t) = \frac{\alpha_t}{\alpha_s} x(s) - \int_s^t \left(\frac{\alpha_t}{\alpha_\tau}\right) \sigma_\tau \frac{d\lambda_\tau}{d\tau} \hat{\boldsymbol{\varepsilon}}_\theta(x(\tau), \tau) d\tau$$

$$= \frac{\alpha_t}{\alpha_s} x(s) - \alpha_t \int_s^t \left(\frac{d\lambda_\tau}{d\tau}\right) e^{-\lambda_\tau} \hat{\boldsymbol{\varepsilon}}_\theta(x(\tau), \tau) d\tau$$

$\because \frac{\alpha_\tau}{\sigma_\tau} = e^{-\lambda_\tau}$

DPM-Solver

$\lambda_t = \lambda(t)$ is a strictly decreasing function of t.

$\Longrightarrow$ There exists an inverse function $t_\lambda(\lambda_t) = t$.

For the second integral term, change the parameter $t \to \lambda$.

$$x(t) = \frac{\alpha_t}{\alpha_s} x(s) - \alpha_t \int_s^t \left(\frac{d\lambda_\tau}{d\tau}\right) e^{-\lambda_\tau} \hat{\boldsymbol{\varepsilon}}_\theta(x(\tau), \tau) d\tau$$

$$= \frac{\alpha_t}{\alpha_s} x(s) - \alpha_t \int_{\lambda_s}^{\lambda_t} \left(\frac{d\lambda_t}{d\lambda_\tau}\right) e^{-\lambda_\tau} \hat{\boldsymbol{\varepsilon}}_\theta(x(\lambda_\tau), \lambda_\tau) d\lambda_\tau$$

DPM-Solver

In a continuous setup,

$$x(t) = \frac{\alpha_t}{\alpha_s} x(s) - \alpha_t \int_{\lambda_s}^{\lambda_t} e^{-\lambda} \hat{\boldsymbol{\varepsilon}}_\theta(x(\lambda), \lambda) d\lambda$$

$t < s$

In a discrete setup (let $s = t + 1$),

$$\boldsymbol{x}_t = \frac{\alpha_t}{\alpha_{t+1}} \boldsymbol{x}_{t+1} - \alpha_t \int_{\lambda_{t+1}}^{\lambda_t} e^{-\lambda} \hat{\boldsymbol{\varepsilon}}_\theta(x(\lambda), \lambda) d\lambda$$

How can this be approximated with discretized time steps?

DPM-Solver-1

First-Order Approximation

In the discrete setup, assume that $\hat{\boldsymbol{\varepsilon}}_\theta(x(\lambda), \lambda)$ is constant as $\hat{\boldsymbol{\varepsilon}}_\theta(x(\lambda_{t+1}), \lambda_{t+1})$ in the range of $(\lambda_t, \lambda_{t+1}]$:

$$\boldsymbol{x}_t \approx \frac{\alpha_t}{\alpha_{t+1}} \boldsymbol{x}_{t-1} - \alpha_t \hat{\boldsymbol{\varepsilon}}_\theta(x(\lambda_{t+1}), \lambda_{t+1}) \int_{\lambda_{t+1}}^{\lambda_t} e^{-\lambda} d\lambda$$

$$= \frac{\alpha_t}{\alpha_{t+1}} \boldsymbol{x}_{t+1} - \alpha_t \left(-e^{-\lambda_t} + e^{-\lambda_{t+1}}\right) \hat{\boldsymbol{\varepsilon}}_\theta(x(\lambda_{t+1}), \lambda_{t+1})$$

DPM-Solver-1

- Let $h_t := \lambda_t - \lambda_{t+1}$

- $-e^{-\lambda_t} + e^{-\lambda_{t+1}} = e^{\lambda_t}\left(e^{\lambda_t - \lambda_{t+1}} - 1\right) = \frac{\sigma_t}{\alpha_t}\left(e^{h_t} - 1\right)$

$\because \frac{\sigma_t}{\alpha_t} = e^{\lambda_t}$

$$
\begin{aligned}
\boldsymbol{x}_t &\approx \frac{\alpha_t}{\alpha_{t+1}}\boldsymbol{x}_{t+1} - \alpha_t\left(-e^{-\lambda_t} + e^{-\lambda_{t+1}}\right)\hat{\boldsymbol{\varepsilon}}_\theta(x(\lambda_{t+1}), \lambda_{t+1}) \\
&= \frac{\alpha_t}{\alpha_{t+1}}\boldsymbol{x}_{t+1} - \alpha_t\left(\frac{\sigma_{t+1}}{\alpha_{t+1}} - \frac{\sigma_t}{\alpha_t}\right)\hat{\boldsymbol{\varepsilon}}_\theta(x(\lambda_{t+1}), \lambda_{t+1}) \\
&= \frac{\alpha_t}{\alpha_{t+1}}\boldsymbol{x}_{t+1} - \left(\alpha_t\frac{\sigma_{t+1}}{\alpha_{t+1}} - \sigma_t\right)\hat{\boldsymbol{\varepsilon}}_\theta(x(\lambda_{t+1}), \lambda_{t+1}) \\
&= \frac{\alpha_t}{\alpha_{t+1}}\boldsymbol{x}_{t+1} - \sigma_t\left(e^{h_t} - 1\right)\hat{\boldsymbol{\varepsilon}}_\theta(x(\lambda_{t+1}), \lambda_{t+1})
\end{aligned}
$$

Variance Preserving (VP) SDE

$$\alpha(t) = e^{-\frac{1}{2}\int_0^t \beta(s)ds} \qquad \sigma^2(t) = 1 - \alpha^2(t)$$

$$\boldsymbol{x}_t \approx \frac{\alpha_t}{\alpha_{t+1}}\boldsymbol{x}_{t+1} - \left(\alpha_t \frac{\sigma_{t+1}}{\alpha_{t+1}} - \sigma_t\right)\hat{\boldsymbol{\varepsilon}}_\theta(x(\lambda_{t+1}), \lambda_{t+1})$$

Q. What is $\boldsymbol{x}_t$ with respect to $\boldsymbol{x}_{t+1}$ and $\beta(t)$?

Recall: DDIM Deterministic Reverse Process

$$q_\sigma(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t,\boldsymbol{x}_0)=\mathcal{N}\left(\sqrt{\bar{\alpha}_{t-1}}\boldsymbol{x}_0+\sqrt{1-\bar{\alpha}_{t-1}-\sigma_t^2}\cdot\frac{\boldsymbol{x}_t-\sqrt{\bar{\alpha}_t}\boldsymbol{x}_0}{\sqrt{1-\bar{\alpha}_t}},\sigma_t^2\mathbf{I}\right)$$

($\sigma_t^2 \to 0$, $\sigma_t^2 \to 0$)

For each time step $t = T, \dots, 1$, repeat:

1. $\boldsymbol{\varepsilon}_t = \hat{\boldsymbol{\varepsilon}}_\theta(\boldsymbol{x}_t, t)$

2. $\boldsymbol{x}_{0|t} = \frac{1}{\sqrt{\bar{\alpha}_t}}\left(\boldsymbol{x}_t - \sqrt{1-\bar{\alpha}_t}\boldsymbol{\varepsilon}_t\right)$

3. $\boldsymbol{x}_{t-1} = \sqrt{\bar{\alpha}_{t-1}}\boldsymbol{x}_{0|t} + \sqrt{1-\bar{\alpha}_{t-1}}\boldsymbol{\varepsilon}_t$

Recall: DDIM Deterministic Reverse Process

$$\boldsymbol{x}_t = \sqrt{\bar{\alpha}_t}\left(\frac{1}{\sqrt{\bar{\alpha}_{t+1}}}\left(\boldsymbol{x}_{t+1} - \sqrt{1-\bar{\alpha}_{t+1}}\hat{\boldsymbol{\varepsilon}}_\theta(\boldsymbol{x}_{t+1}, t+1)\right)\right) + \sqrt{1-\bar{\alpha}_t}\hat{\boldsymbol{\varepsilon}}_\theta(\boldsymbol{x}_{t+1}, t+1)$$

$$= \frac{\sqrt{\bar{\alpha}_t}}{\sqrt{\bar{\alpha}_{t+1}}}\boldsymbol{x}_{t+1} - \left(\frac{\sqrt{\bar{\alpha}_t}}{\sqrt{\bar{\alpha}_{t+1}}}\sqrt{1-\bar{\alpha}_{t+1}} - \sqrt{1-\bar{\alpha}_t}\right)\hat{\boldsymbol{\varepsilon}}_\theta(\boldsymbol{x}_{t+1}, t+1)$$

$$= \frac{\alpha_t}{\alpha_{t+1}}\boldsymbol{x}_{t+1} + \left(\alpha_t\frac{\sigma_{t+1}}{\alpha_{t+1}} - \sigma_t\right)\hat{\boldsymbol{\varepsilon}}_\theta(\boldsymbol{x}_{t+1}, t+1)$$

$$\alpha_t = \sqrt{\bar{\alpha}_t}$$
$$\sigma_t = \sqrt{1-\bar{\alpha}_t}$$

The first-order approximation is identical to DDIM!

DPM-Solver-2

First-Order Approximation:

$$\boldsymbol{x}_t \approx \frac{\alpha_t}{\alpha_{t+1}}\boldsymbol{x}_{t+1} - \sigma_t\left(e^{h_t} - 1\right)\hat{\boldsymbol{\varepsilon}}_\theta(x(t+1), t+1)$$

How about the second-order approximation?

DPM-Solver-2

1. Split the λ interval $(\lambda_t, \lambda_{t+1}]$ in half:

$$\tilde{\lambda} = \frac{\lambda_t + \lambda_{t+1}}{2}$$

2. Perform the first-order approximation for the first half:

$$\boldsymbol{u} \leftarrow \frac{\alpha_t}{\alpha_{t+1}} \boldsymbol{x}_{t+1} - \sigma_{t_\lambda(\tilde{\lambda})} \left(e^{\frac{h_t}{2}} - 1\right) \hat{\boldsymbol{\varepsilon}}_\theta(x(t+1), t+1)$$

3. Use the approximation to update $\hat{\boldsymbol{\varepsilon}}_\theta$ and compute the final $\boldsymbol{x}_t$:

$$\boldsymbol{x}_t \approx \frac{\alpha_t}{\alpha_{t+1}} \boldsymbol{x}_{t+1} - \sigma_t \left(e^{h_t} - 1\right) \hat{\boldsymbol{\varepsilon}}_\theta \left(\boldsymbol{u}, \tilde{\lambda}\right)$$

How about the third-order approximation?

DPM-Solver-n

Higher-Order Approximation

Taylor expansion of $\hat{\boldsymbol{\varepsilon}}_\theta(x(\lambda), \lambda)$:

$$\hat{\boldsymbol{\varepsilon}}_\theta(x(\lambda), \lambda) = \sum_{k=0}^{n-1} \frac{(\lambda - \lambda_{t+1})^k}{k!} \hat{\boldsymbol{\varepsilon}}_\theta^{(k)}(x(\lambda_{t+1}), \lambda_{t+1}) + \eta$$

DPM-Solver-n

Plug it into the discrete setup formulation:

$$\boldsymbol{x}_t = \frac{\alpha_t}{\alpha_{t+1}}\boldsymbol{x}_{t+1} - \alpha_t \int_{\lambda_{t+1}}^{\lambda_t} e^{-\lambda}\hat{\boldsymbol{\varepsilon}}_\theta(x(\lambda),\lambda)d\lambda$$

$$= \frac{\alpha_t}{\alpha_{t+1}}\boldsymbol{x}_{t+1} - \alpha_t \int_{\lambda_{t+1}}^{\lambda_t} e^{-\lambda}\left(\sum_{k=0}^{n-1}\frac{(\lambda-\lambda_{t+1})^k}{k!}\hat{\boldsymbol{\varepsilon}}_\theta^{(k)}(x(\lambda_{t+1}),\lambda_{t+1}) + \eta\right)d\lambda$$

$$= \frac{\alpha_t}{\alpha_{t+1}}\boldsymbol{x}_{t+1} - \alpha_t \sum_{k=0}^{n-1}\hat{\boldsymbol{\varepsilon}}_\theta^{(n)}(x(\lambda_{t+1}),\lambda_{t+1})\int_{\lambda_{t+1}}^{\lambda_t} e^{-\lambda}\frac{(\lambda-\lambda_{t+1})^k}{k!}d\lambda + \eta$$

Can be computed analytically

Further Readings

- Lu et al., DPM-Solver++: Fast Solver for Guided Sampling of Diffusion Probabilistic Models, 2023.

- Zheng et al., DPM-Solver-v3: Improved Diffusion ODE Solver with Empirical Model Statistics, NeurIPS 2023.

- Karras et al., Elucidating the Design Space of Diffusion-Based Generative Models, NeurIPS 2022.

extra

All SDEs for reverse diffusions follow the FP equation!

Equivalence of Marginal Densities

- **SDE dynamics:**

$\nabla \cdot \nabla = \Delta$: Laplacian

$$\partial_t p_t(x) = -\nabla \cdot \big[b(t,x)\, p_t(x)\big] \;+\; \tfrac{c}{2}\, \Delta\, p_t(x)$$

- **ODE dynamics:** define drift $F(t,x) = b(t,x) - \frac{c}{2}\, \nabla \log p_t(x)$

$$\partial_t p_t(x) = -\nabla \cdot \big[F(t,x)\, p_t(x)\big] = -\nabla \cdot \Big[\big(b(t,x) - \tfrac{c}{2} \nabla \log p_t(x)\big) p_t(x)\Big]$$

- **Algebraic check:** show the two RHS agree

$$\begin{aligned} -\nabla \cdot \Big[\big(b - \tfrac{c}{2} \nabla \log p\big) p\Big] &= -\nabla \cdot (b\, p) \;+\; \tfrac{c}{2}\, \nabla \cdot \big[(\nabla \log p)\, p\big] \\ &= -\nabla \cdot (b\, p) \;+\; \tfrac{c}{2}\, \nabla \cdot (\nabla p) \\ &= -\nabla \cdot (b\, p) \;+\; \tfrac{c}{2}\, \Delta\, p \end{aligned}$$

- Hence both the SDE and the ODE yield the *same* PDE for p_t
- Therefore at each t, their marginal $X_t \sim p_t$ agree

Recap on the Fokker-Planck equation

- We have the **Fokker-Planck** equation

$$\boxed{\partial_t p_t(x) = -\mathrm{div}(b(t,\cdot)p_t)(x) + \tfrac{1}{2}\sum_{i,j=1}^{d} \partial_{i,j}\{\Sigma_{i,j}(t,\cdot)p_t\}(x)\ ,}$$

where $\mathrm{div}(a(t,x)) := \sum_{i=1}^{d} \partial_{x_i} a_i(t,x)$.

- This equation describes the **evolution of the density**.
- Some special cases:
  - Case $\sigma = 0$ (**deterministic dynamics**)

$$\boxed{\partial_t p_t(x) = -\mathrm{div}(b(t,\cdot)p_t)(x)\ .}$$

  - Case $\sigma = c^{1/2}\,\mathrm{Id}$ $(c > 0)$ (**Langevin dynamics**)

$$\boxed{\partial_t p_t(x) = -\mathrm{div}(b(t,\cdot)p_t)(x) + \tfrac{c}{2}\Delta p_t(x) = -\mathrm{div}(\{b(t,\cdot) - \tfrac{c}{2}\nabla \log p_t\}p_t)(x)\ .}$$

- As a consequence the two following dynamics have the **same** marginal densities.
  - $\mathrm{d}\mathbf{X}_t = b(t,\mathbf{X}_t)\mathrm{d}t + c^{1/2}\mathrm{d}\mathbf{B}_t$
  - $\mathrm{d}\mathbf{X}_t = \{b(t,\mathbf{X}_t) - \tfrac{c}{2}\nabla \log p_t(\mathbf{X}_t)\}\mathrm{d}t.$
  - One is **deterministic**, the other is **stochastic**.

https://vdeborto.github.io/project/generative_modeling/session_3.pdf

Example SDE: Ornstein–Uhlenbeck process

Example:

$$dX_t = -\beta X_t dt + \sigma dW_t$$

$$X_t \,|\, X_0 \sim \mathcal{N}\left(e^{-\beta t} X_0, \frac{\sigma^2}{2\beta}(1 - e^{-2\beta t})\right)$$

If $X_0 \sim \mathcal{N}(0, \sigma^2/\beta)$

$$X_t \sim \mathcal{N}\left(0, \frac{\sigma^2}{2\beta}\right)$$

$$p_t(X_t) = \frac{1}{\sqrt{\pi\sigma^2/\beta}} \exp\left[-\frac{\beta}{\sigma^2}(X_t)^2\right]$$

With direct calculations, we can verify that p_t satisfies the FP equation.

$$\begin{aligned} 0 = \partial_t p_t(x) &= -\partial_x(f p_t) + \frac{g^2}{2}\partial_x^2(p_t) \\ &= \partial_x(\beta x p_t(x)) + \frac{\sigma^2}{2}\partial_x^2(p_t(x)) \\ &= 0 \end{aligned}$$

From score-based models to normalizing flows

- Recall that the **continuous**-time version of **diffusion model** is given by

$$\boxed{\mathrm{d}\mathbf{Y}_t = \{\mathbf{Y}_t + 2\nabla \log p_{T-t}(\mathbf{Y}_t)\}\mathrm{d}t + \sqrt{2}\mathrm{d}\mathbf{B}_t \ .}$$

- It is the **backward** dynamics associated with the **forward Ornstein-Ulhenbeck**

$$\boxed{\mathrm{d}\mathbf{X}_t = -\mathbf{X}_t + \sqrt{2}\mathrm{d}\mathbf{B}_t \ .}$$

- For any $t \geq 0$, $\mathcal{L}(\mathbf{X}_t)$ has the same distribution as $\mathcal{L}(\hat{\mathbf{X}}_t)$ where for any $t \geq 0$

$$\boxed{\mathrm{d}\hat{\mathbf{X}}_t = \{-\hat{\mathbf{X}}_t - \nabla \log p_t(\hat{\mathbf{X}}_t)\}\mathrm{d}t \ .}$$

- The **backward** of this **deterministic** dynamics is $(\hat{\mathbf{Y}}_t)_{t\in[0,T]}$ given by

$$\boxed{\mathrm{d}\hat{\mathbf{Y}}_t = \{\hat{\mathbf{Y}}_t + \nabla \log p_{T-t}(\hat{\mathbf{Y}}_t)\}\mathrm{d}t \ .}$$

- Comparing $(\mathbf{Y}_t)_{t\in[0,T]}$ and $(\hat{\mathbf{Y}}_t)_{t\in[0,T]}$
  - $(\mathbf{Y}_t)_{t\in[0,T]}$ is **stochastic** and $(\hat{\mathbf{Y}}_t)_{t\in[0,T]}$ is **deterministic**.
  - Both are given by the **time-reversal** of a **forward dynamics**.
  - They both approximate the **data distribution** at time $t = T$.
  - Notice the difference of a **factor two** in the drift!