

# Deep Generative Models

## Lecture 11

Roman Isachenko

Moscow Institute of Physics and Technology  
Yandex School of Data Analysis

2024, Autumn

# Recap of previous lecture

## Training of DDPM

1. Get the sample  $\mathbf{x}_0 \sim \pi(\mathbf{x})$ .
2. Sample timestamp  $t \sim U\{1, T\}$  and the noise  $\epsilon \sim \mathcal{N}(0, \mathbf{I})$ .
3. Get noisy image  $\mathbf{x}_t = \sqrt{\bar{\alpha}_t} \cdot \mathbf{x}_0 + \sqrt{1 - \bar{\alpha}_t} \cdot \epsilon$ .
4. Compute loss  $\mathcal{L}_{\text{simple}} = \|\epsilon - \epsilon_{\theta,t}(\mathbf{x}_t)\|^2$ .

## Sampling of DDPM

1. Sample  $\mathbf{x}_T \sim \mathcal{N}(0, \mathbf{I})$ .
2. Compute mean of  $p(\mathbf{x}_{t-1}|\mathbf{x}_t, \theta) = \mathcal{N}(\mu_{\theta,t}(\mathbf{x}_t), \sigma_t^2 \cdot \mathbf{I})$ :

$$\mu_{\theta,t}(\mathbf{x}_t) = \frac{1}{\sqrt{\alpha_t}} \cdot \mathbf{x}_t - \frac{1 - \alpha_t}{\sqrt{\alpha_t(1 - \bar{\alpha}_t)}} \cdot \epsilon_{\theta,t}(\mathbf{x}_t)$$

3. Get denoised image  $\mathbf{x}_{t-1} = \mu_{\theta,t}(\mathbf{x}_t) + \sigma_t \cdot \epsilon$ , where  $\epsilon \sim \mathcal{N}(0, \mathbf{I})$ .

# Recap of previous lecture

## DDPM objective

$$\mathbb{E}_{\pi(\mathbf{x}_0)} \mathbb{E}_{t \sim U\{1, T\}} \mathbb{E}_{q(\mathbf{x}_t | \mathbf{x}_0)} \left[ \frac{(1 - \alpha_t)^2}{2\tilde{\beta}_t \alpha_t} \left\| \mathbf{s}_{\theta, t}(\mathbf{x}_t) - \nabla_{\mathbf{x}_t} \log q(\mathbf{x}_t | \mathbf{x}_0) \right\|_2^2 \right]$$

In practice the coefficient is omitted.

## NCSN objective

$$\mathbb{E}_{\pi(\mathbf{x}_0)} \mathbb{E}_{t \sim U\{1, T\}} \mathbb{E}_{q(\mathbf{x}_t | \mathbf{x}_0)} \left\| \mathbf{s}_{\theta, \sigma_t}(\mathbf{x}_t) - \nabla_{\mathbf{x}_t} \log q(\mathbf{x}_t | \mathbf{x}_0) \right\|_2^2$$

**Note:** The objective of DDPM and NCSN is almost identical. But the difference in sampling scheme:

- ▶ NCSN uses annealed Langevin dynamics;
- ▶ DDPM uses ancestral sampling.

$$\mathbf{s}_{\theta, t}(\mathbf{x}_t) = -\frac{\epsilon_{\theta, t}(\mathbf{x}_t)}{\sqrt{1 - \bar{\alpha}_t}} = \nabla_{\mathbf{x}_t} \log p(\mathbf{x}_t | \theta)$$

# Recap of previous lecture

## Unconditional generation

$$\mathbf{x}_{t-1} = \frac{1}{\sqrt{\alpha_t}} \cdot \mathbf{x}_t + \frac{1 - \alpha_t}{\sqrt{\alpha_t}} \cdot \nabla_{\mathbf{x}_t} \log p(\mathbf{x}_t | \boldsymbol{\theta}) + \sigma_t \cdot \epsilon$$

## Conditional generation

$$\mathbf{x}_{t-1} = \frac{1}{\sqrt{\alpha_t}} \cdot \mathbf{x}_t + \frac{1 - \alpha_t}{\sqrt{\alpha_t}} \cdot \nabla_{\mathbf{x}_t} \log p(\mathbf{x}_t | \mathbf{y}, \boldsymbol{\theta}) + \sigma_t \cdot \epsilon$$

## Conditional distribution

$$\nabla_{\mathbf{x}_t} \log p(\mathbf{x}_t | \mathbf{y}, \boldsymbol{\theta}) = \nabla_{\mathbf{x}_t} \log p(\mathbf{y} | \mathbf{x}_t) - \frac{\epsilon_{\boldsymbol{\theta},t}(\mathbf{x}_t)}{\sqrt{1 - \bar{\alpha}_t}}$$

Here  $p(\mathbf{y} | \mathbf{x}_t)$  – classifier on noisy samples (we have to learn it separately).

## Classifier-corrected noise prediction

$$\epsilon_{\boldsymbol{\theta},t}(\mathbf{x}_t, \mathbf{y}) = \epsilon_{\boldsymbol{\theta},t}(\mathbf{x}_t) - \sqrt{1 - \bar{\alpha}_t} \cdot \nabla_{\mathbf{x}_t} \log p(\mathbf{y} | \mathbf{x}_t)$$

# Recap of previous lecture

## Guidance scale

$$\epsilon_{\theta,t}(\mathbf{x}_t, \mathbf{y}) = \epsilon_{\theta,t}(\mathbf{x}_t) - \gamma \cdot \sqrt{1 - \bar{\alpha}_t} \cdot \nabla_{\mathbf{x}_t} \log p(\mathbf{y}|\mathbf{x}_t)$$

$$\nabla_{\mathbf{x}_t}^{\gamma} \log p(\mathbf{x}_t|\mathbf{y}, \theta) = \nabla_{\mathbf{x}_t} \log \left( \frac{p(\mathbf{y}|\mathbf{x}_t)^{\gamma} p(\mathbf{x}_t|\theta)}{Z} \right)$$

**Note:** Guidance scale  $\gamma$  tries to sharpen the distribution  $p(\mathbf{y}|\mathbf{x}_t)$ .

## Guided sampling

$$\epsilon_{\theta,t}(\mathbf{x}_t, \mathbf{y}) = \epsilon_{\theta,t}(\mathbf{x}_t) - \gamma \cdot \sqrt{1 - \bar{\alpha}_t} \cdot \nabla_{\mathbf{x}_t} \log p(\mathbf{y}|\mathbf{x}_t)$$

$$\mu_{\theta,t}(\mathbf{x}_t, \mathbf{y}) = \frac{1}{\sqrt{\alpha_t}} \mathbf{x}_t - \frac{1 - \alpha_t}{\sqrt{\alpha_t(1 - \bar{\alpha}_t)}} \cdot \epsilon_{\theta,t}(\mathbf{x}_t, \mathbf{y})$$

$$\mathbf{x}_{t-1} = \mu_{\theta,t}(\mathbf{x}_t, \mathbf{y}) + \sigma_t \cdot \epsilon, \quad \epsilon \sim \mathcal{N}(0, \mathbf{I})$$

## Recap of previous lecture

- ▶ Previous method requires training the additional classifier model  $p(\mathbf{y}|\mathbf{x}_t)$  on the noisy data.
- ▶ Let try to avoid this requirement.

$$\nabla_{\mathbf{x}_t} \log p(\mathbf{y}|\mathbf{x}_t) = \nabla_{\mathbf{x}_t} \log p(\mathbf{x}_t|\mathbf{y}, \boldsymbol{\theta}) - \nabla_{\mathbf{x}_t} \log p(\mathbf{x}_t|\boldsymbol{\theta})$$

$$\begin{aligned}\nabla_{\mathbf{x}_t}^\gamma \log p(\mathbf{x}_t|\mathbf{y}, \boldsymbol{\theta}) &= \nabla_{\mathbf{x}_t} \log p(\mathbf{x}_t|\boldsymbol{\theta}) + \gamma \cdot \nabla_{\mathbf{x}_t} \log p(\mathbf{y}|\mathbf{x}_t) = \\ &= (1 - \gamma) \cdot \nabla_{\mathbf{x}_t} \log p(\mathbf{x}_t|\boldsymbol{\theta}) + \gamma \cdot \nabla_{\mathbf{x}_t} \log p(\mathbf{x}_t|\mathbf{y}, \boldsymbol{\theta})\end{aligned}$$

## Classifier-free-corrected noise prediction

$$\hat{\epsilon}_{\boldsymbol{\theta},t}(\mathbf{x}_t, \mathbf{y}) = \gamma \cdot \epsilon_{\boldsymbol{\theta},t}(\mathbf{x}_t, \mathbf{y}) + (1 - \gamma) \cdot \epsilon_{\boldsymbol{\theta},t}(\mathbf{x}_t)$$

- ▶ Train the single model  $\epsilon_{\boldsymbol{\theta},t}(\mathbf{x}_t, \mathbf{y})$  on **supervised** data alternating with real conditioning  $\mathbf{y}$  and empty conditioning  $\mathbf{y} = \emptyset$ .
- ▶ Apply the model twice during inference.

# Recap of previous lecture

## Continuous-in-time dynamic (neural ODE)

$$\frac{dz(t)}{dt} = \mathbf{f}_{\theta}(\mathbf{z}(t), t); \quad \text{with initial condition } \mathbf{z}(0) = \mathbf{z}_0.$$

$$\mathbf{z}(t_1) = \int_0^1 \mathbf{f}_{\theta}(\mathbf{z}(t), t) dt + \mathbf{z}_0 \approx \text{ODESolve}(\mathbf{z}(0), \mathbf{f}_{\theta}, t_0 = 0, t_1 = 1).$$

## Euler update step

$$\frac{\mathbf{z}(t + \Delta t) - \mathbf{z}(t)}{\Delta t} = \mathbf{f}_{\theta}(\mathbf{z}(t), t) \Rightarrow \mathbf{z}(t + \Delta t) = \mathbf{z}(t) + \Delta t \cdot \mathbf{f}_{\theta}(\mathbf{z}(t), t)$$

## Theorem (Picard)

If  $\mathbf{f}$  is uniformly Lipschitz continuous in  $\mathbf{z}$  and continuous in  $t$ , then the ODE has a **unique** solution.

$$\mathbf{x} = \mathbf{z}(1) = \mathbf{z}(0) + \int_0^1 \mathbf{f}_{\theta}(\mathbf{z}(t), t) dt$$

$$\mathbf{z} = \mathbf{z}(0) = \mathbf{z}(1) + \int_1^0 \mathbf{f}_{\theta}(\mathbf{z}(t), t) dt$$

# Outline

1. Kolmogorov-Fokker-Planck equation for NF log-likelihood
2. FFJORD (Hutchinson's trace estimator)
3. Adjoint method for continuous-in-time NF
4. SDE basics



# Outline

1. Kolmogorov-Fokker-Planck equation for NF log-likelihood
2. FFJORD (Hutchinson's trace estimator)
3. Adjoint method for continuous-in-time NF
4. SDE basics

# Continuous-in-time Normalizing Flows

What do we need?

- ▶ We need the way to compute  $p_t(\mathbf{z})$  at any moment  $t$ .
- ▶ We need the way to find the optimal parameters  $\theta$  of the dynamic  $\mathbf{f}_\theta$ .

Theorem (Kolmogorov-Fokker-Planck: special case)

If  $\mathbf{f}$  is uniformly Lipschitz continuous in  $\mathbf{z}$  and continuous in  $t$ , then

$$\frac{d \log p_t(\mathbf{z}(t))}{dt} = -\text{tr} \left( \frac{\partial \mathbf{f}_\theta(\mathbf{z}(t), t)}{\partial \mathbf{z}(t)} \right).$$

$$\log p_1(\mathbf{z}(1)) = \log p_0(\mathbf{z}(0)) - \int_0^1 \text{tr} \left( \frac{\partial \mathbf{f}_\theta(\mathbf{z}(t), t)}{\partial \mathbf{z}(t)} \right) dt.$$

It means that if we have the value  $\mathbf{z}_0 = \mathbf{z}(0)$  then the solution of the ODE will give us the density at the moment  $t = 1$ .

# Continuous-in-time Normalizing Flows

Forward transform + log-density

$$\mathbf{x} = \mathbf{z} + \int_0^1 \mathbf{f}_\theta(\mathbf{z}(t), t) dt$$
$$\log p(\mathbf{x}|\theta) = \log p(\mathbf{z}) - \int_0^1 \text{tr} \left( \frac{\partial \mathbf{f}_\theta(\mathbf{z}(t), t)}{\partial \mathbf{z}(t)} \right) dt$$

Here  $p(\mathbf{x}|\theta) = p_1(\mathbf{z})$ ,  $p(\mathbf{z}) = p_0(\mathbf{z})$ .

- ▶ **Discrete-in-time NF**: evaluation of determinant of the Jacobian costs  $O(m^3)$  (we need invertible  $\mathbf{f}$ ).
- ▶ **Continuous-in-time NF**: getting the trace of the Jacobian costs  $O(m^2)$  (we need smooth  $\mathbf{f}$ ).

Why  $O(m^2)$ ?

$\text{tr} \left( \frac{\partial \mathbf{f}_\theta(\mathbf{z}(t))}{\partial \mathbf{z}(t)} \right)$  costs  $O(m^2)$  ( $m$  evaluations of  $\mathbf{f}$ ), since we have to compute a derivative for each diagonal element. It is possible to reduce cost from  $O(m^2)$  to  $O(m)$ !

# Outline

1. Kolmogorov-Fokker-Planck equation for NF log-likelihood
2. FFJORD (Hutchinson's trace estimator)
3. Adjoint method for continuous-in-time NF
4. SDE basics

# Continuous-in-time Normalizing Flows

## Hutchinson's trace estimator

If  $\epsilon \in \mathbb{R}^m$  is a random variable with  $\mathbb{E}[\epsilon] = 0$  and  $\text{cov}(\epsilon) = \mathbf{I}$ , then

$$\begin{aligned}\text{tr}(\mathbf{A}) &= \text{tr}(\mathbf{A} \cdot \mathbf{I}) = \text{tr}\left(\mathbf{A} \cdot \mathbb{E}_{p(\epsilon)}\left[\epsilon\epsilon^T\right]\right) = \\ &= \mathbb{E}_{p(\epsilon)}\left[\text{tr}\left(\mathbf{A}\epsilon\epsilon^T\right)\right] = \mathbb{E}_{p(\epsilon)}\left[\epsilon^T \mathbf{A} \epsilon\right]\end{aligned}$$

Jacobian vector products  $\mathbf{v}^T \frac{\partial \mathbf{f}}{\partial \mathbf{z}}$  can be computed for approximately the same cost as evaluating  $\mathbf{f}$  (`torch.autograd.functional.jvp`).

## FFJORD density estimation

$$\begin{aligned}\log p_1(\mathbf{z}(1)) &= \log p_0(\mathbf{z}(0)) - \int_0^1 \text{tr}\left(\frac{\partial \mathbf{f}_\theta(\mathbf{z}(t), t)}{\partial \mathbf{z}(t)}\right) dt = \\ &= \log p_0(\mathbf{z}(0)) - \mathbb{E}_{p(\epsilon)} \int_0^1 \left[\epsilon^T \frac{\partial \mathbf{f}}{\partial \mathbf{z}} \epsilon\right] dt.\end{aligned}$$

# Outline

1. Kolmogorov-Fokker-Planck equation for NF log-likelihood
2. FFJORD (Hutchinson's trace estimator)
3. Adjoint method for continuous-in-time NF
4. SDE basics

# Neural ODE

## Continuous-in-time NF

$$\begin{aligned}\frac{d\mathbf{z}(t)}{dt} &= \mathbf{f}_{\theta}(\mathbf{z}(t), t) & \frac{d \log p_t(\mathbf{z}(t))}{dt} &= -\text{tr} \left( \frac{\partial \mathbf{f}_{\theta}(\mathbf{z}(t), t)}{\partial \mathbf{z}(t)} \right) \\ \mathbf{x} &= \mathbf{z} + \int_0^1 \mathbf{f}_{\theta}(\mathbf{z}(t), t) dt & \log p(\mathbf{x}|\theta) &= \log p(\mathbf{z}) - \int_0^1 \text{tr} \left( \frac{\partial \mathbf{f}_{\theta}(\mathbf{z}(t), t)}{\partial \mathbf{z}(t)} \right) dt\end{aligned}$$

How to get optimal parameters of  $\theta$ ?

For fitting parameters we need gradients. We need the analogue of the backpropagation.

## Forward pass (Loss function)

$$\mathbf{z} = \mathbf{x} + \int_1^0 \mathbf{f}_{\theta}(\mathbf{z}(t), t) dt, \quad L(\mathbf{z}) = -\log p(\mathbf{x}|\theta)$$

$$L(\mathbf{z}) = -\log p(\mathbf{z}) + \int_0^1 \text{tr} \left( \frac{\partial \mathbf{f}_{\theta}(\mathbf{z}(t), t)}{\partial \mathbf{z}(t)} \right) dt$$

# Neural ODE

## Adjoint functions

$$\mathbf{a}_z(t) = \frac{\partial L}{\partial \mathbf{z}(t)}; \quad \mathbf{a}_\theta(t) = \frac{\partial L}{\partial \theta(t)}.$$

These functions show how the gradient of the loss depends on the hidden state  $\mathbf{z}(t)$  and parameters  $\theta$ .

## Theorem (Pontryagin)

$$\frac{d\mathbf{a}_z(t)}{dt} = -\mathbf{a}_z(t)^T \cdot \frac{\partial \mathbf{f}_\theta(\mathbf{z}(t), t)}{\partial \mathbf{z}}; \quad \frac{d\mathbf{a}_\theta(t)}{dt} = -\mathbf{a}_z(t)^T \cdot \frac{\partial \mathbf{f}_\theta(\mathbf{z}(t), t)}{\partial \theta}.$$

## Solution for adjoint function

$$\begin{aligned} \frac{\partial L}{\partial \theta(1)} &= \mathbf{a}_\theta(1) = - \int_0^1 \mathbf{a}_z(t)^T \frac{\partial \mathbf{f}_\theta(\mathbf{z}(t), t)}{\partial \theta(t)} dt + 0 \\ \frac{\partial L}{\partial \mathbf{z}(1)} &= \mathbf{a}_z(1) = - \int_0^1 \mathbf{a}_z(t)^T \frac{\partial \mathbf{f}_\theta(\mathbf{z}(t), t)}{\partial \mathbf{z}(t)} dt + \frac{\partial L}{\partial \mathbf{z}(0)} \end{aligned}$$

**Note:** These equations are solved in reverse time direction.



# Adjoint method

## Forward pass

$$\mathbf{z} = \mathbf{z}(0) = \int_0^1 \mathbf{f}_{\theta}(\mathbf{z}(t), t) dt + \mathbf{x} \quad \Rightarrow \quad \text{ODE Solver}$$

## Backward pass

$$\left. \begin{aligned} \frac{\partial L}{\partial \theta(1)} &= \mathbf{a}_{\theta}(1) = - \int_0^1 \mathbf{a}_{\mathbf{z}}(t)^T \frac{\partial \mathbf{f}_{\theta}(\mathbf{z}(t), t)}{\partial \theta(t)} dt + 0 \\ \frac{\partial L}{\partial \mathbf{z}(1)} &= \mathbf{a}_{\mathbf{z}}(1) = - \int_0^1 \mathbf{a}_{\mathbf{z}}(t)^T \frac{\partial \mathbf{f}_{\theta}(\mathbf{z}(t), t)}{\partial \mathbf{z}(t)} dt + \frac{\partial L}{\partial \mathbf{z}(0)} \\ \mathbf{z}(1) &= - \int_1^0 \mathbf{f}_{\theta}(\mathbf{z}(t), t) dt + \mathbf{z}_0. \end{aligned} \right\} \Rightarrow \text{ODE Solver}$$

**Note:** These scary formulas are the standard backprop in the discrete case.

# Outline

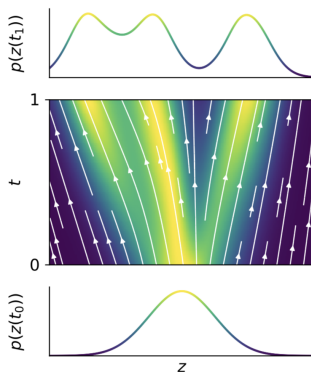
1. Kolmogorov-Fokker-Planck equation for NF log-likelihood
2. FFJORD (Hutchinson's trace estimator)
3. Adjoint method for continuous-in-time NF
4. SDE basics

# Ordinary differential equation (ODE)

## Neural ODE

$$\frac{dz(t)}{dt} = \mathbf{f}_\theta(\mathbf{z}(t), t); \quad \text{with initial condition } \mathbf{z}(t_0) = \mathbf{z}_0$$

- ▶  $\mathbf{z}(t_0)$  is a random variable with the density function  $p(\mathbf{z}(t_0))$ .
- ▶  $\mathbf{z}(t_1)$  is a random variable with the density function  $p(\mathbf{z}(t_1))$ .
- ▶  $p_t(\mathbf{z}) = p(\mathbf{z}, t)$  is the joint density function (probability path).  
What is the difference between  $p_t(\mathbf{z}(t))$  and  $p_t(\mathbf{z})$ ?
- ▶ Let consider time interval  $[t_0, t_1] = [0, 1]$  without loss of generality.



# Ordinary differential equation (ODE)

$$d\mathbf{z} = \mathbf{f}_{\theta}(\mathbf{z}, t) \cdot dt$$

## Discretization of ODE (Euler method)

$$\mathbf{z}(t + dt) = \mathbf{z}(t) + \mathbf{f}_{\theta}(\mathbf{z}(t), t) \cdot dt$$

## Theorem (Kolmogorov-Fokker-Planck: special case)

If  $\mathbf{f}$  is uniformly Lipschitz continuous in  $\mathbf{z}$  and continuous in  $t$ , then

$$\frac{d \log p(\mathbf{z}(t), t)}{dt} = -\text{tr} \left( \frac{\partial \mathbf{f}_{\theta}(\mathbf{z}(t), t)}{\partial \mathbf{z}(t)} \right).$$

It means that if we have the value  $\mathbf{z}_0 = \mathbf{z}(0)$  then the solution of the ODE will give us the density at the moment  $t = 1$ .

# Stochastic differential equation (SDE)

Let define stochastic process  $\mathbf{x}(t)$  with initial condition  $\mathbf{x}(0) \sim p_0(\mathbf{x}) = \pi(\mathbf{x})$ :

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

- ▶  $\mathbf{f}(\mathbf{x}, t) : \mathbb{R}^m \times [0, 1] \rightarrow \mathbb{R}^m$  is the **drift** function of  $\mathbf{x}(t)$ .
- ▶  $g(t) : \mathbb{R} \rightarrow \mathbb{R}$  is the **diffusion** function of  $\mathbf{x}(t)$ .
- ▶  $\mathbf{w}(t)$  is the standard Wiener process (Brownian motion):
  1.  $\mathbf{w}(0) = 0$  (almost surely);
  2.  $\mathbf{w}(t)$  has independent increments;
  3.  $\mathbf{w}(t) - \mathbf{w}(s) \sim \mathcal{N}(0, (t - s)\mathbf{I})$ , for  $t > s$ .
- ▶  $d\mathbf{w} = \mathbf{w}(t + dt) - \mathbf{w}(t) = \mathcal{N}(0, \mathbf{I} \cdot dt) = \epsilon \cdot \sqrt{dt}$ , where  $\epsilon \sim \mathcal{N}(0, \mathbf{I})$ .
- ▶ If  $g(t) = 0$  we get standard ODE.

# Stochastic differential equation (SDE)

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

- ▶ In contrast to ODE, initial condition  $\mathbf{x}(0)$  does not uniquely determine the process trajectory.
- ▶ We have two sources of randomness: initial distribution  $p_0(\mathbf{x})$  and Wiener process  $\mathbf{w}(t)$ .

## Discretization of SDE (Euler method)

$$\mathbf{x}(t + dt) = \mathbf{x}(t) + \mathbf{f}(\mathbf{x}(t), t) \cdot dt + g(t) \cdot \epsilon \cdot \sqrt{dt}$$

If  $dt = 1$ , then

$$\mathbf{x}_{t+1} = \mathbf{x}_t + \mathbf{f}(\mathbf{x}_t, t) + g(t) \cdot \epsilon$$

- ▶ At each moment  $t$  we have the density  $p_t(\mathbf{x}) = p(\mathbf{x}, t)$ .
- ▶  $p : \mathbb{R}^m \times [0, 1] \rightarrow \mathbb{R}_+$  is a **probability path** between  $p_0(\mathbf{x})$  and  $p_1(\mathbf{x})$ .
- ▶ How to get the distribution path  $p_t(\mathbf{x})$  for  $\mathbf{x}(t)$ ?

# Stochastic differential equation (SDE)

$$d\mathbf{x} = \mathbf{f}(\mathbf{x}, t)dt + g(t)d\mathbf{w}, \quad d\mathbf{w} = \boldsymbol{\epsilon} \cdot \sqrt{dt}, \quad \boldsymbol{\epsilon} \sim \mathcal{N}(0, \mathbf{I}).$$

## Theorem (Kolmogorov-Fokker-Planck)

Evolution of the distribution  $p_t(\mathbf{x})$  is given by the following equation:

$$\frac{\partial p_t(\mathbf{x})}{\partial t} = -\text{div}(\mathbf{f}(\mathbf{x}, t)p_t(\mathbf{x})) + \frac{1}{2}g^2(t)\Delta_{\mathbf{x}}p_t(\mathbf{x})$$

Here

$$\text{div}(\mathbf{v}) = \sum_{i=1}^m \frac{\partial v_i(\mathbf{x})}{\partial x_i} = \text{tr} \left( \frac{\partial \mathbf{v}(\mathbf{x})}{\partial \mathbf{x}} \right)$$

$$\Delta_{\mathbf{x}}p_t(\mathbf{x}) = \sum_{i=1}^m \frac{\partial^2 p_t(\mathbf{x})}{\partial x_i^2} = \text{tr} \left( \frac{\partial^2 p_t(\mathbf{x})}{\partial \mathbf{x}^2} \right)$$

$$\frac{\partial p_t(\mathbf{x})}{\partial t} = \text{tr} \left( -\frac{\partial}{\partial \mathbf{x}} [\mathbf{f}(\mathbf{x}, t)p_t(\mathbf{x})] + \frac{1}{2}g^2(t)\frac{\partial^2 p_t(\mathbf{x})}{\partial \mathbf{x}^2} \right)$$

# Stochastic differential equation (SDE)

## Theorem (Kolmogorov-Fokker-Planck)

$$\frac{\partial p_t(\mathbf{x})}{\partial t} = \text{tr} \left( -\frac{\partial}{\partial \mathbf{x}} [\mathbf{f}(\mathbf{x}, t) p_t(\mathbf{x})] + \frac{1}{2} g^2(t) \frac{\partial^2 p_t(\mathbf{x})}{\partial \mathbf{x}^2} \right)$$

- ▶ KFP theorem does not define the SDE uniquely in general case.
- ▶ This is the generalization of KFP theorem that we used in continuous-in-time NF:

$$\frac{d \log p_t(\mathbf{x}(t))}{dt} = -\text{tr} \left( \frac{\partial \mathbf{f}(\mathbf{x}, t)}{\partial \mathbf{x}} \right).$$

## Langevin SDE (special case)

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

$$d\mathbf{x} = \frac{1}{2} \frac{\partial}{\partial \mathbf{x}} \log p_t(\mathbf{x})dt + \mathbf{1} \cdot d\mathbf{w}$$

Let apply KFP theorem to this SDE.



## Langevin SDE (special case)

$$d\mathbf{x} = \frac{1}{2} \frac{\partial}{\partial \mathbf{x}} \log p_t(\mathbf{x}) dt + 1 \cdot d\mathbf{w}$$

$$\begin{aligned} \frac{\partial p_t(\mathbf{x})}{\partial t} &= \text{tr} \left( -\frac{\partial}{\partial \mathbf{x}} \left[ p_t(\mathbf{x}) \frac{1}{2} \frac{\partial}{\partial \mathbf{x}} \log p_t(\mathbf{x}) \right] + \frac{1}{2} \frac{\partial^2 p_t(\mathbf{x})}{\partial \mathbf{x}^2} \right) = \\ &= \text{tr} \left( -\frac{\partial}{\partial \mathbf{x}} \left[ \frac{1}{2} \frac{\partial}{\partial \mathbf{x}} p_t(\mathbf{x}) \right] + \frac{1}{2} \frac{\partial^2 p_t(\mathbf{x})}{\partial \mathbf{x}^2} \right) = 0 \end{aligned}$$

The density  $p_t(\mathbf{x}) = \text{const}(t)$ !

If  $\mathbf{x}(0) \sim p_0(\mathbf{x})$ , then  $\mathbf{x}(t) \sim p_0(\mathbf{x})$ .

## Discretized Langevin SDE

$$\mathbf{x}_{t+1} - \mathbf{x}_t = \frac{\eta}{2} \cdot \frac{\partial}{\partial \mathbf{x}} \log p_t(\mathbf{x}) + \sqrt{\eta} \cdot \epsilon, \quad \eta \approx dt.$$

## Langevin dynamic

$$\mathbf{x}_{t+1} = \mathbf{x}_t + \frac{\eta}{2} \cdot \nabla_{\mathbf{x}} \log p(\mathbf{x}|\theta) + \sqrt{\eta} \cdot \epsilon, \quad \eta \approx dt.$$

# Summary

- ▶ Kolmogorov-Fokker-Planck theorem allows to calculate  $\log p(\mathbf{z}, t)$  at arbitrary moment  $t$ .
- ▶ FFJORD model makes such kind of NF scalable.
- ▶ SDE defines stochastic process with drift and diffusion terms. ODEs are the special case of SDEs.
- ▶ KFP equation defines the dynamic of the probability function for the SDE.
- ▶ Langevin SDE has constant probability path.