

Deep Generative Models

Lecture 13

Roman Isachenko

 Ozon Masters

Spring, 2022

Outline

1. Neural ODE: finish
2. Continuous-in-time normalizing flows
3. Stochastic differential equations basics (SDE)

Outline

1. Neural ODE: finish
2. Continuous-in-time normalizing flows
3. Stochastic differential equations basics (SDE)

Neural ODE

Adjoint functions

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

Theorem (Pontryagin)

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

Do we know any initial condition?

Solution for adjoint function

$$\frac{\partial L}{\partial \theta(t_0)} = \mathbf{a}_\theta(t_0) = - \int_{t_1}^{t_0} \mathbf{a}_z(t)^T \frac{\partial f(\mathbf{z}(t), \theta)}{\partial \theta(t)} dt + 0$$

$$\frac{\partial L}{\partial \mathbf{z}(t_0)} = \mathbf{a}_z(t_0) = - \int_{t_1}^{t_0} \mathbf{a}_z(t)^T \frac{\partial f(\mathbf{z}(t), \theta)}{\partial \mathbf{z}(t)} dt + \frac{\partial L}{\partial \mathbf{z}(t_1)}$$

Note: These equations are solved back in time.

Neural ODE

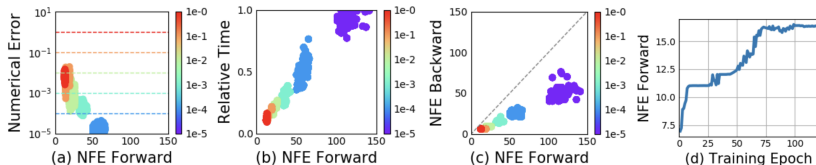
Forward pass

$$\mathbf{z}(t_1) = \int_{t_0}^{t_1} f(\mathbf{z}(t), \boldsymbol{\theta}) dt + \mathbf{z}_0 \Rightarrow \text{ODE Solver}$$

Backward pass

$$\left. \begin{aligned} \frac{\partial L}{\partial \boldsymbol{\theta}(t_0)} &= \mathbf{a}_{\boldsymbol{\theta}}(t_0) = - \int_{t_1}^{t_0} \mathbf{a}_{\mathbf{z}}(t)^T \frac{\partial f(\mathbf{z}(t), \boldsymbol{\theta})}{\partial \boldsymbol{\theta}(t)} dt + 0 \\ \frac{\partial L}{\partial \mathbf{z}(t_0)} &= \mathbf{a}_{\mathbf{z}}(t_0) = - \int_{t_1}^{t_0} \mathbf{a}_{\mathbf{z}}(t)^T \frac{\partial f(\mathbf{z}(t), \boldsymbol{\theta})}{\partial \mathbf{z}(t)} dt + \frac{\partial L}{\partial \mathbf{z}(t_1)} \\ \mathbf{z}(t_0) &= - \int_{t_1}^{t_0} f(\mathbf{z}(t), \boldsymbol{\theta}) dt + \mathbf{z}_1. \end{aligned} \right\} \Rightarrow \text{ODE Solver}$$

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



Outline

1. Neural ODE: finish
2. Continuous-in-time normalizing flows
3. Stochastic differential equations basics (SDE)

Continuous Normalizing Flows

Discrete Normalizing Flows

$$\mathbf{z}_{t+1} = f(\mathbf{z}_t, \boldsymbol{\theta}); \quad \log p(\mathbf{z}_{t+1}) = \log p(\mathbf{z}_t) - \log \left| \det \frac{\partial f(\mathbf{z}_t, \boldsymbol{\theta})}{\partial \mathbf{z}_t} \right|.$$

Continuous-in-time dynamic transformation

$$\frac{d\mathbf{z}(t)}{dt} = f(\mathbf{z}(t), \boldsymbol{\theta}).$$

Assume that function f is uniformly Lipschitz continuous in \mathbf{z} and continuous in t . From Picard's existence theorem, it follows that the above ODE has a **unique solution**.

Forward and inverse transforms

$$\begin{aligned}\mathbf{x} = \mathbf{z}(t_1) &= \mathbf{z}(t_0) + \int_{t_0}^{t_1} f(\mathbf{z}(t), \boldsymbol{\theta}) dt \\ \mathbf{z} = \mathbf{z}(t_0) &= \mathbf{z}(t_1) + \int_{t_1}^{t_0} f(\mathbf{z}(t), \boldsymbol{\theta}) dt\end{aligned}$$

Continuous Normalizing Flows

To train this flow we have to get the way to calculate the density $p(\mathbf{z}(t))$.

Theorem (special case of Kolmogorov-Fokker-Planck)

if function f is uniformly Lipschitz continuous in \mathbf{z} and continuous in t , then

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

Note: Unlike discrete-in-time flows, the function f does not need to be bijective, because uniqueness guarantees that the entire transformation is automatically bijective.

Density evaluation

$$\log p(\mathbf{x}|\boldsymbol{\theta}) = \log p(\mathbf{z}) - \int_{t_0}^{t_1} \text{trace} \left(\frac{\partial f(\mathbf{z}(t), \boldsymbol{\theta})}{\partial \mathbf{z}(t)} \right) dt.$$

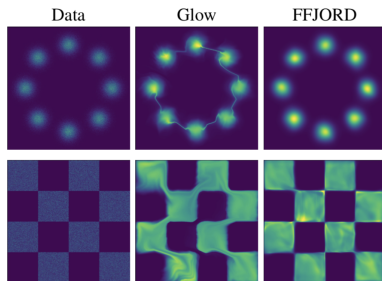
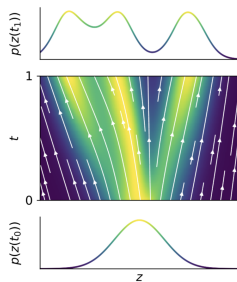
Adjoint method is used to integral evaluation.

Continuous Normalizing Flows

Forward transform + log-density

$$\begin{bmatrix} \mathbf{x} \\ \log p(\mathbf{x}|\boldsymbol{\theta}) \end{bmatrix} = \begin{bmatrix} \mathbf{z} \\ \log p(\mathbf{z}) \end{bmatrix} + \int_{t_0}^{t_1} \begin{bmatrix} f(\mathbf{z}(t), \boldsymbol{\theta}) \\ -\text{trace} \left(\frac{\partial f(\mathbf{z}(t), \boldsymbol{\theta})}{\partial \mathbf{z}(t)} \right) \end{bmatrix} dt.$$

- ▶ Discrete-in-time normalizing flows need invertible f . It costs $O(d^3)$ to get determinant of Jacobian.
- ▶ Continuous-in-time flows require only smoothness of f . It costs $O(d^2)$ to get trace of Jacobian.



Outline

1. Neural ODE: finish
2. Continuous-in-time normalizing flows
3. Stochastic differential equations basics (SDE)

Stochastic differential equation (SDE)

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

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

- ▶ $\mathbf{w}(t)$ is the standard Wiener process (Brownian motion)

$$\mathbf{w}(t) - \mathbf{w}(s) \sim \mathcal{N}(0, t-s), \quad d\mathbf{w} = \epsilon \cdot \sqrt{dt}, \text{ where } \epsilon \sim \mathcal{N}(0, 1).$$

- ▶ $\mathbf{f}(\mathbf{x}, t)$ is the **drift** function of $\mathbf{x}(t)$.
- ▶ $g(t)$ is the **diffusion** coefficient of $\mathbf{x}(t)$.
- ▶ If $g(t) = 0$ we get standard ODE.

Stochastic differential equation (SDE)

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

How to get distribution $p(\mathbf{x}|t)$ for $\mathbf{x}(t)$?

Theorem (Kolmogorov-Fokker-Planck)

Evolution of the distribution $p(\mathbf{x}|t)$ is given by the following ODE:

$$\frac{\partial p(\mathbf{x}|t)}{\partial t} = -\frac{\partial}{\partial \mathbf{x}}(\mathbf{f}(\mathbf{x}, t)p(\mathbf{x})) + \frac{1}{2}g^2(t)\frac{\partial^2 p(\mathbf{x}|t)}{\partial \mathbf{x}^2}$$

Stochastic differential equation (SDE)

Langevin SDE

Let consider special case of SDE with $g(t) = 1$ and $\mathbf{f}(\mathbf{x}, t) = \frac{1}{2} \frac{\partial}{\partial \mathbf{x}} \log p(\mathbf{x}|t)$.

$$d\mathbf{x} = \frac{1}{2} \frac{\partial}{\partial \mathbf{x}} \log p(\mathbf{x}|t) dt + d\mathbf{w}$$

Let apply KFP theorem.

$$\begin{aligned} \frac{\partial p(\mathbf{x}|t)}{\partial t} &= -\frac{\partial}{\partial \mathbf{x}} \left(p(\mathbf{x}|t) \frac{1}{2} \frac{\partial}{\partial \mathbf{x}} \log p(\mathbf{x}|t) \right) + \frac{1}{2} \frac{\partial^2 p(\mathbf{x}|t)}{\partial \mathbf{x}^2} = \\ &= -\frac{\partial}{\partial \mathbf{x}} \left(\frac{1}{2} \frac{\partial}{\partial \mathbf{x}} p(\mathbf{x}|t) \right) + \frac{1}{2} \frac{\partial^2 p(\mathbf{x}|t)}{\partial \mathbf{x}^2} = 0 \end{aligned}$$

The density is $p(\mathbf{x}|t) = \text{const.}$

Langevin dynamic

Let discretize the Langevin SDE

Stochastic differential equation (SDE)

Langevin dynamic

Let discretize the Langevin SDE

$$\mathbf{x}_{t+1} = \mathbf{x}_t + \eta \frac{1}{2} \frac{\partial}{\partial \mathbf{x}} \log p(\mathbf{x}|t) + \sqrt{\eta} \cdot \epsilon, \quad \epsilon \sim \mathcal{N}(0, 1).$$

Summary

- ▶ Adjoint method generalizes backpropagation procedure and allows to train Neural ODE solving ODE for adjoint function back in time.
- ▶ Fokker-Planck theorem allows to construct continuous-in-time normalizing flow with less functional restrictions.
- ▶ FFJORD model makes such kind of flows scalable.