How to Compute Likelihood with Diffusion Models

Jiachun Jin

ShanghaiTech University

Abstract

For some time now I did not know exactly why we can compute the likelihood of a sample with a diffusion model. In this note, I discuss how the ODE nature of a diffusion model makes exact likelihood evaluation possible.

1 The Probability Flow ODE

In diffusion models, we first have a forward diffusion process that perturbs the data distribution $p_0 = p_{\text{data}}$ to the prior distribution $p_1 = \mathcal{N}(0, \mathbf{I})$

$$\mathsf{d}\mathbf{x} = f_t(\mathbf{x})\mathsf{d}t + g_t\mathsf{d}\mathbf{w}, \tag{1}$$

we are always interested in the marginal distribution $p_t(\mathbf{x}), \forall t \in [0, 1]$, and its instantaneous change can be described by the Fokker-Planck Equation:

$$\frac{\partial}{\partial t} p_t(\mathbf{x}) = -\nabla_{\mathbf{x}} \cdot [f_t(\mathbf{x})p_t(\mathbf{x})] + \frac{1}{2} g_t^2 \nabla_{\mathbf{x}} \cdot \nabla_{\mathbf{x}} p_t(\mathbf{x}), \tag{2}$$

note that for the RHS, $\forall \sigma_t^2 < g_t^2$, we always have the following equivalence (refer to the blog post by Jianlin Su)

$$\begin{aligned}
& -\nabla_{\mathbf{x}} \cdot [f_t(\mathbf{x})p_t(\mathbf{x})] + \frac{1}{2} g_t^2 \nabla_{\mathbf{x}} \cdot \nabla_{\mathbf{x}} p_t(\mathbf{x}) \\
&= -\nabla_{\mathbf{x}} \cdot \left[f_t(\mathbf{x})p_t(\mathbf{x}) - \frac{1}{2}\left(g_t^2 - \sigma_t^2\right) \nabla_{\mathbf{x}} p_t(\mathbf{x}) \right] + \frac{1}{2} \sigma_t^2 \nabla_{\mathbf{x}} \cdot \nabla_{\mathbf{x}} p_t(\mathbf{x}) \\
&= -\nabla_{\mathbf{x}} \cdot \left[\underbrace{\left(f_t(\mathbf{x}) - \frac{1}{2}\left(g_t^2 - \sigma_t^2\right) \nabla_{\mathbf{x}} \log p_t(\mathbf{x}) \right)}_{\tilde{f}_t} p_t(\mathbf{x}) \right] + \frac{1}{2} \underbrace{\sigma_t^2}_{\tilde{g}_t^2} \nabla_{\mathbf{x}} \cdot \nabla_{\mathbf{x}} p_t(\mathbf{x}),
\end{aligned} \tag{3}$$

this tell us that all the following diffusion processes have the same marginal distribution as equation 1

$$\begin{aligned}
\mathsf{d}\mathbf{x} &= \tilde{f}_t(\mathbf{x})\mathsf{d}t + \tilde{g}_t \mathsf{d}\mathbf{w} \\
&= \left(f_t(\mathbf{x}) - \frac{1}{2}\left(g_t^2 - \sigma_t^2\right) \nabla_{\mathbf{x}} \log p_t(\mathbf{x}) \right) \mathsf{d}t + \sigma_t \mathsf{d}\mathbf{w},
\end{aligned} \tag{4}$$

and one special case can be the one given by setting $\sigma_t = 0$, in this case all the stochacity is removed, and the marginal distributions deduced from the ordianary differential equation

$$\mathsf{d}\mathbf{x} = f_t(\mathbf{x})\mathsf{d}t - \frac{1}{2} g_t^2 \nabla_{\mathbf{x}} \log p_t(\mathbf{x})\mathsf{d}t, \tag{5}$$

is still equivalent to the ones deduced by equation 1. This is the probability flow ODE (PF ODE), and has the same form whenever in the forward or the reverse direction.

2 The Instantaneous Change of Variables Formula

In practice, we always use a neural network $s_t^\theta(\mathbf{x})$ to approximate the exact score $\nabla_\mathbf{x} \log p_t(\mathbf{x})$, and in this case, the continuous dynamics of $\mathbf{x}_t$ is specified by the following Neural ODE [Chen et al., 2018]

$$\frac{\mathrm{d}\mathbf{x}}{\mathrm{d}t} = f_t(\mathbf{x}) - \frac{1}{2} g_t^2 s_t^\theta(\mathbf{x}) = \tau_t^\theta(\mathbf{x}), \tag{6}$$

and we can easily get the instantaneous change of $p_t(\mathbf{x})$ from equation 2 by plugging $f_t(\mathbf{x}) = \tau_\theta(\mathbf{x}, t)$ and $g_t = 0$

$$\frac{\partial}{\partial t} p_t(\mathbf{x}) = -\nabla_\mathbf{x} \cdot \left[\tau_t^\theta(\mathbf{x}) p_t(\mathbf{x})\right] + 0, \tag{7}$$

equation 7 is sometimes called as the Continuity Equation, that the instantaneous change of $p_t(\mathbf{x})$ is determined by the trace of the Jacobian of $\tau_t^\theta(\mathbf{x}) p_t(\mathbf{x})$. Further by the log-derivative trick, we finally reach to

$$\frac{\partial}{\partial t} \log p_t(\mathbf{x}) = \frac{1}{p_t(\mathbf{x})} \frac{\partial}{\partial t} p_t(\mathbf{x}) = -\nabla_\mathbf{x} \cdot \left[\tau_t^\theta(\mathbf{x})\right] = -\mathrm{trace}\left(\frac{\partial}{\partial \mathbf{x}} \tau_t^\theta\right). \tag{8}$$

Remark 1. *In the above, I omitted the dependence to t of $\mathbf{x}$, actually $\mathbf{x}$ itself is a random variable relied on the time index t, which is $\mathbf{x}(t)$.*

With the foundamental theorem of calculus, we have

$$\log p_1(\mathbf{x}(1)) - \log p_0(\mathbf{x}(0)) = \int_0^1 -\nabla_\mathbf{x} \cdot \left[\tau_t^\theta(\mathbf{x})\right] \mathrm{d}t, \tag{9}$$

thus the log density of a generated sample $\mathbf{x}(0)$ from $\mathbf{x}(1)$ from a diffusion model can be computed as

$$\log p_0(\mathbf{x}(0)) = \log p_1(\mathbf{x}(1)) + \int_0^1 \nabla_\mathbf{x} \cdot \left[\tau_t^\theta(\mathbf{x})\right] \mathrm{d}t. \tag{10}$$

3 Computing the Likelihood

In the following, I am going to do some notation change. I will use $\mathbf{z}$ to denote $\mathbf{x}(1)$ and $\mathbf{x}$ to denote $\mathbf{x}(0)$, and $\mathbf{z}(t)$ to denote the intermediate latent variables. With the new notation, the neural ODE becomes to

$$\frac{\mathrm{d}\mathbf{z}(t)}{\mathrm{d}t} = \tau_t^\theta(\mathbf{z}(t)), \tag{11}$$

and the log density of $\mathbf{x}$ in equation 10 is given by

$$\log p(\mathbf{x}) = \log p(\mathbf{z}) + \int_0^1 \mathrm{trace}\left(\frac{\partial}{\partial \mathbf{z}(t)} \tau_t^\theta\right) \mathrm{d}t. \tag{12}$$

Given a new data point $\mathbf{x}$, to compute $\log p(\mathbf{x})$, we first need to integrate equation 11 to get the latent variable $\mathbf{z}$, and then integrate equation 12 to get the final result. Actually, we can do this in one pass by integrating the LHS of the below equation

$$\begin{bmatrix}\mathbf{x} \\ 0\end{bmatrix} + \int_0^1 \begin{bmatrix}\tau_t^\theta(\mathbf{z}(t)) \\ \mathrm{trace}\left(\frac{\partial}{\partial \mathbf{z}(t)} \tau_t^\theta\right)\end{bmatrix} \mathrm{d}t = \begin{bmatrix}\mathbf{z} \\ \log p(\mathbf{x}) - \log p(\mathbf{z})\end{bmatrix}. \tag{13}$$

And the complexity is $\mathcal{O}\left(D^2 T\right)$.

References

R. T. Chen, Y. Rubanova, J. Bettencourt, and D. K. Duvenaud. Neural ordinary differential equations. Advances in neural information processing systems, 31, 2018.