Karras et al.'s DDPM Improvements

Pramook Khungurn

January 17, 2023

This note is written as I read "Elucidating the Design Space of Diffusion-Based Generative Models" by Karras et al. [KAAL22]. I have to admit that it is very hard for me to read the paper because it requires so much background to understand, and all the derivations are folded into the appendix.

1 Introduction

- The paper casts a number of previous works on DDPM and score-based generative models into a common framework and suggests improvements to multiple parts of generative models in the framework.
- The previous works include:
  - The stochastic differential equation paper by Song et al. [SSDK+21].
  - The improved DDPM paper by Nichol and Dharival [ND21].
  - The DDIM paper by Song et al. [SME20].
  - The adaptive step-size SDE solver paper by Jolicoeur-Martineau et al. [JMLPT+21].
- The end results are score improvements on several datasets.
  - SOTA FID score on CIFAR-10.
  - Near-SOTA FID score on ImageNet-64 without retraining.
  - SOTA FID score on ImageNet-64 with retraining.
- I also believe that the Heun's 2nd order solver introduced in this paper has been include as a sampler in Stable Diffusion.
  - Actually, this is why I try to read the paper.

2 A Common Framework for Diffusion Models

2.1 Preliminary

- Let each data item be a vector in $\mathbb{R}^d$. We typically denote one with the letter $\mathbf{x}$.
- Let $p_{\mathrm{data}}(\mathbf{x})$ denote the data distribution. Let σ_{data} denotes the standard deviation of the ata.
- All diffusion model works with the distribution $p(\mathbf{x};\sigma)$ obtained by adding i.i.d. Gaussian noise of standard deviation σ to the data sampled from p_{data}.
  - $p(\mathbf{x};0) = p_{\mathrm{data}}(\mathbf{x})$.
  - If $\sigma_{\max} \gg \sigma_{\mathrm{data}}$, then $p(\mathbf{x};\sigma_{\max})$ would be very close to $\mathcal{N}(\mathbf{0}, \sigma_{\max}^2 I)$.
- In general, here's how a diffusion model generates a sample.

1. We specify a number of noise levels $0 = \sigma_1 < \sigma_2 < \cdots < \sigma_T = \sigma_{\max}$.
2. We first sample a point $\mathbf{x}_T$ from $\mathcal{N}(\mathbf{0}, \sigma_{\max}^2 I)$ and simply assume that it comes from $p(\mathbf{x}; \sigma_{\max})$.
3. Given a point $\mathbf{x}_t$ that comes from $p(\mathbf{x}; \sigma_t)$, we revert the noising process inherent in $p(\mathbf{x}; \sigma_t)$ to produce a point $\mathbf{x}_{t-1}$ that should come from $p(\mathbf{x}; \sigma_{t-1})$.
4. Starting from $\mathbf{x}_T$, we repeat Step 3 until we reach $\mathbf{x}_0$, which is returned as the output of the sampling process.

2.2 A General SDE and Its Probability Flow ODE

- In [SSDK+21], Song et al. suggests that the noising process (i.e., $p(\mathbf{x}, \sigma)$) can be modeled by stochastic differential equations.
- The sequence σ_1, σ_2, $\dots$, σ_T becomes a continuous function $\sigma(t)$ of time where $t \in [0, T]$.
- Here, $\sigma(0) = 0$, and $\sigma(T) = \sigma_{\max}$.
- For each tiem t, we view $\mathbf{x}(t)$ which should be distributed according to $p(\mathbf{x}; \sigma(t))$ as a random variable. This means that $\{\mathbf{x}(t) : t \in [0, T]\}$ is a stochastic process.
- The evaluation of the above stochastic process is governed by the stochastic differential equation:

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

  where $\mathbf{f} : \mathbb{R}^d \times \mathbb{R} \to \mathbb{R}^d$ is called the **drift coefficient**, $g(t) : \mathbb{R} \to \mathbb{R}$ is called the **diffusion coefficient**, and $\mathbf{W}(t)$ is the standard d-dimensional Brownian motion (aka the Wiener process).
- The initial condition is $\mathbf{x}(0) \sim p_{\text{data}}$.
  - To make it simpler, we say that $\mathbf{x}(0)$ is fixed and has no variance when deriving a solution to SDEs.
- The Song et al. paper gives two SDEs. One is called the *variance-exploding (VE)* SDE, and another the *variance-preserving (VP)* SDE. Both SDEs are of the form

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

  where $f(\mathbf{x}, t)$ becomes $f(t)\mathbf{x}$, and f is reduced to a function of signature $\mathbb{R} \to \mathbb{R}$.
- The above SDE has an explicit, unique solution. We appeal from the following theorem (Equation 6.2) from [SS19].

  Theorem 1. *Consider a* **linear SDE** *of the form*

$$\mathrm{d}\mathbf{x} = \big(F(t)\mathbf{x} + \mathbf{u}(t)\big)\,\mathrm{d}t + L(t)\,\mathrm{d}\mathbf{W}$$

  where $F : \mathbb{R} \to \mathbb{R}^{d\times d}$, $\mathbf{u} : \mathbb{R} \to \mathbb{R}^d$, and $L : \mathbb{R} \to \mathbb{R}^{d\times d}$. We have that the solution of this equation is a Gaussian process whose mean and covariance matrix,

$$\begin{aligned} \boldsymbol{\mu}(t) &= E[\mathbf{x}(t)], \\ \Sigma(t) &= E[(\mathbf{x}(t) - \boldsymbol{\mu}(t))(\mathbf{x}(t) - \boldsymbol{\mu}(t))^T], \end{aligned}$$

  satisfy the ODEs

$$\begin{aligned} \frac{\mathrm{d}\boldsymbol{\mu}(t)}{\mathrm{d}t} &= F(t)\boldsymbol{\mu}(t), \\ \frac{\mathrm{d}\Sigma(t)}{\mathrm{d}t} &= F(t)\Sigma(t) + \Sigma(t)F(t)^T + L(t)L(t)^T. \end{aligned}$$

- Setting $F(t) = f(t)I$, $\mathbf{u}(t) = \mathbf{0}$, and $L(t) = g(t)I$, we have that

$$\begin{aligned}\frac{\mathrm{d}\boldsymbol{\mu}(t)}{\mathrm{d}t} &= f(t)\boldsymbol{\mu}(t),\\ \frac{\mathrm{d}\Sigma(t)}{\mathrm{d}t} &= 2f(t)\Sigma(t) + g(t)^2 I.\end{aligned}$$

  Assuming that the sample $\mathbf{x}(0)$ has already been sampled and fixed. The initial condition is $\boldsymbol{\mu}(0) = \mathbf{x}(0)$, and $\Sigma(0) = 0$. Solving the first ODE, we have that

$$\boldsymbol{\mu}(0) = \mathbf{x}(0)\exp\bigg(\int_0^t f(u)\,\mathrm{d}u\bigg).$$

  The paper defines

$$s(t) := \exp\bigg(\int_0^t f(u)\,\mathrm{d}u\bigg),$$

  and so

$$\boldsymbol{\mu}(0) = s(t)\mathbf{x}(0).$$

  For the second ODE, notice that $\Sigma(t)$ is a diagonal matrix whose entries are all the same. Let the entries have value $v(t)$. We have that

$$\frac{\mathrm{d}v(t)}{\mathrm{d}t} = 2f(t)v(t) + g(t)^2.$$

  Solving the equation (see Proposition 2), we have that

$$v(t) = s(t)^2 \int_0^t \frac{g(u)^2}{s(u)^2}\,\mathrm{d}u,$$

  assumming that $v(0) = 0$ (because we are under the setting where the sample $\mathbf{x}(0)$ has alrady been fixed). The paper defines

$$\sigma(t) := \sqrt{\int_0^t \frac{g(u)^2}{s(u)^2}\,\mathrm{d}u},$$

  and so the conditional density of $\mathbf{x}(t)$ given $\mathbf{x}(0)$ is given by

$$p_{t|0}(\mathbf{x}|\mathbf{x}_0) = \mathcal{N}(\mathbf{x}; s(t)\mathbf{x}_0, s(t)^2\sigma(t)^2 I).$$

  The marginal density of $\mathbf{x}(t)$ is given by

$$p_t(\mathbf{x}) = \int_{\mathbb{R}^d} p_{t|0}(\mathbf{x}|\mathbf{x}_0)p_{\mathrm{data}}(\mathbf{x}_0)\,\mathrm{d}\mathbf{x}_0.$$

- According to [SSDK+21], the stochastic process $\mathbf{x}(t)$, when interpreted as a function of signature $\mathbb{R} \to \mathbb{R}^d$, and the marginal probability distribution $p_t(\mathbf{x})$ also satisfies the ODE

$$\frac{\mathrm{d}\mathbf{x}}{\mathrm{d}t} = f(t)\mathbf{x} - \frac{1}{2}g(t)^2\nabla_{\mathbf{x}}\log p_t(\mathbf{x})$$

  called the **probability flow ODE**. (Deriving this equation can be done my manipulating the Fokker–Planck equation that $p_t(\mathbf{x})$ is supposed to satisfy. See my notes on score-based generative models for more information [Khu22].)

2.3 Making $s(t)$ and $\sigma(t)$ First Class Citizens

- Note that the conditional distribution $p_{t|0}$ is given in terms of $s(t)$ and $\sigma(t)$, which are derived from $f(t)$ and $g(t)$. However, we are more interested in $s(t)$ and $\sigma(t)$ because they tells use directly what the means and the standard deviations are. So, the paper rewrites the probability ODE in terms of $s(t)$ and $\sigma(t)$ instead of $f(t)$ and $g(t)$.

- Consider the expression for the marginal distribution of $\mathbf{x}(t)$.

$$\begin{aligned} p_t(\mathbf{x}) &= \int_{\mathbb{R}^d} p_{t|0}(\mathbf{x}|\mathbf{x}_0) p_{\mathrm{data}}(\mathbf{x}_0)\, \mathrm{d}\mathbf{x}_0 \\ &= \int_{\mathbb{R}^d} \mathcal{N}(\mathbf{x}; s(t)\mathbf{x}_0, s^2(t)\sigma^2(t)I) p_{\mathrm{data}}(\mathbf{x}_0)\, \mathrm{d}\mathbf{x}_0 \\ &= \int_{\mathbb{R}^d} s(t)^{-d} \mathcal{N}(\mathbf{x}/s(t); \mathbf{x}_0, \sigma^2(t)I) p_{\mathrm{data}}(\mathbf{x}_0)\, \mathrm{d}\mathbf{x}_0 \\ &= s(t)^{-d} \int_{\mathbb{R}^d} \mathcal{N}(\mathbf{x}/s(t); \mathbf{x}_0, \sigma^2(t)I) p_{\mathrm{data}}(\mathbf{x}_0)\, \mathrm{d}\mathbf{x}_0 \\ &= s(t)^{-d} \Big[p_{\mathrm{data}} * \mathcal{N}(\mathbf{0}, \sigma^2(t)I) \Big] \big(\mathbf{x}/s(t) \big) \end{aligned}$$

  where $*$ is the convolution operation.

- $p_{\mathrm{data}} * \mathcal{N}(\mathbf{0}, \sigma^2(t)I)$ is the corrupted version of p_{data} due to the noise added by the diffusion process. The paper defines

$$p(\mathbf{x}; \sigma) := p_{\mathrm{data}} * \mathcal{N}(\mathbf{0}, \sigma^2 I).$$

  So,

$$p_t(\mathbf{x}) = \frac{p(\mathbf{x}/s(t); \sigma(t))}{s(t)^d}.$$

- Plugging the expression for $p_t(\mathbf{x})$ into the probability flow ODE, we have that

$$\begin{aligned} \frac{\mathrm{d}\mathbf{x}}{\mathrm{d}t} &= f(t)\mathbf{x} - \frac{1}{2} g(t)^2 \nabla_{\mathbf{x}} \log \frac{p(\mathbf{x}/s(t); \sigma(t))}{s(t)^d} \\ \frac{\mathrm{d}\mathbf{x}}{\mathrm{d}t} &= f(t)\mathbf{x} - \frac{1}{2} g(t)^2 \nabla_{\mathbf{x}} \log p(\mathbf{x}/s(t); \sigma(t)). \end{aligned}$$

- We now write $f(t)$ and $g(t)$ in terms of $s(t)$ and $\sigma(t)$. For $f(t)$, we have that

$$\begin{aligned} \dot{s}(t) &= f(t)s(t) \\ f(t) &= \frac{\dot{s}(t)}{s(t)}. \end{aligned}$$

  For $g(t)$, we have that

$$\begin{aligned} \{\sigma^2(t)\}' &= \frac{g(t)^2}{s(t)^2} \\ 2\sigma(t)^2\dot{\sigma}(t) &= \frac{g(t)^2}{s(t)^2} \\ g(t) &= s(t)\sqrt{2\sigma(t)\dot{\sigma}(t)}. \end{aligned}$$

- Plugging in expressions for $f(t)$ and $g(t)$, we have that

$$\frac{\mathrm{d}\mathbf{x}}{\mathrm{d}t} = \frac{\dot{s}(t)}{s(t)}\mathbf{x} - s(t)^2\sigma(t)\dot{\sigma}(t)\nabla_{\mathbf{x}} \log p(\mathbf{x}/s(t); \sigma(t)). \tag{1}$$

  When $s(t) = 1$ for all t (which is the case for all the variations the paper deals with except the VP SDE formulation), the equation becomes,

$$\mathrm{d}\mathbf{x} = -\sigma(t)\dot{\sigma}(t)\nabla_{\mathbf{x}} \log p(\mathbf{x}; \sigma(t))\,\mathrm{d}t.$$

2.4 Deterministic Sampling

- A way to sample from p_{data} is to (1) sample an element from $p_T(\mathbf{x})$ and (2) simulate Equation (1) numerically backward in time from $t = T$ to $t = 0$. The ability to sample from $p_T(\mathbf{x})$ rests on the fact that $p_{T|0} \sim \mathcal{N}(s(T)\mathbf{x}(0), s(T)^2\sigma(T)^2 I)$. This can be easy in the following situations.

  1. $s(T) = 1$, and $\sigma(T) \gg \sigma_{\text{data}}$.
     In this case, $p_T(\mathbf{x}) \approx \mathcal{N}(\mathbf{0}, \sigma(T)^2 I)$, and we can sample $\mathbf{x}(T)$ from $\mathcal{N}(\mathbf{0}, \sigma(T)^2 I)$.
  2. $s(T) \approx 0$, and $s(T)^2\sigma(T)^2 = 1$.
     In this case, $p_T(\mathbf{x}) \approx \mathcal{N}(\mathbf{0}, I)$, and we can sample $\mathbf{x}(T)$ from $\mathcal{N}(\mathbf{0}, I)$.

  In [SSDK+21], the VE formulation corresponds to the first situation, and the VP formulation corresponds to the second situation.

- Note that the sampling algorithm above is not "deterministic" per se. There is randomness in the sampling of $\mathbf{x}(T)$. However, this is the only randomness that is present in the algorithm. We will discuss "stochastic" sampling algorithms later. These algorithms use random numbers in each of its iterations.

- In order to simulate the ODE of Equation (1) numerically, we need to be able to compute the **score** $\nabla_{\mathbf{x}} \log p(\mathbf{x}/s(t); \sigma^2(t))$. To make the problem more general, we drop the $s(t)^{-1}$ scaling factor and see if there is a way to compute $\nabla_{\mathbf{x}} \log p(\mathbf{x}; \sigma)$. A common way to do this is to train a neural network $D_{\boldsymbol{\theta}}(\widetilde{\mathbf{x}}, \sigma)$ such that it outputs $\mathbf{x}$ given a sample $\widetilde{\mathbf{x}} \sim p(\mathbf{x}; \sigma)$. This can be done by minimizing the loss

$$\mathcal{L}(\boldsymbol{\theta}, \sigma) := E_{\mathbf{x}_0 \sim p_{\text{data}}} E_{\boldsymbol{\xi} \sim \mathcal{N}(0,I)} \left[\left\| D_{\boldsymbol{\theta}}(\mathbf{x}_0 + \sigma\boldsymbol{\xi}, \sigma) - \mathbf{x}_0 \right\|^2 \right]$$

  Then, for the optimal $\boldsymbol{\theta}^*$, we would have that

$$\nabla_{\mathbf{x}} \log p(\mathbf{x}, \sigma) \approx \frac{D_{\boldsymbol{\theta}^*}(\mathbf{x}, \sigma) - \mathbf{x}}{\sigma^2}.$$

  The implementation of $D_{\boldsymbol{\theta}}(\mathbf{x}, \sigma)$ might include scaling $\mathbf{x}$ by the appropriate factor to make the inference easier. It is also common to use a single neural network for all σ values, and so we need to think of a way to sample σ and a way to assign weight to each $\mathcal{L}(\boldsymbol{\theta}, \sigma)$.

- We will come back to how to train $D(\mathbf{x}, \sigma)$ later, but let us formulate the sampling algorithm right away that involves the probability flow ODE right away.

- We distinguished between samples from the distribution $p(\mathbf{x}; \sigma(t)) = \mathcal{N}(\mathbf{x}; \mathbf{x}(0), \sigma(t)^2 I)$, which is not scaled by $s(t)$, and the distribution $p_t(\mathbf{x}) = \mathcal{N}(\mathbf{x}; s(t)\mathbf{x}(0), s(t)^2\sigma(t)^2 I)$, which is scaled by $s(t)$.

- In particular, we let $\mathbf{x}$ denote a sample from $p_t(\mathbf{x})$, and $\hat{\mathbf{x}}$ denote a sample from $p(\hat{\mathbf{x}}, \sigma(t))$. It follows that

$$\mathbf{x} = s(t)\hat{\mathbf{x}}.$$

  The above equation is supposed to be read in the following way: one can sample $\mathbf{x} \sim p_T$ by sampling $\hat{\mathbf{x}} \sim \mathcal{N}(\mathbf{x}(0), \sigma^2(t)I)$ and them computing $\mathbf{x} := s(t)\hat{\mathbf{x}}$.

Algorithm 1 Euler method for sampling from p_{data} by simulating the probability flow ODE (2).

```
EULER-SAMPLER($D_{\boldsymbol{\theta}}, \sigma, s, \{t_0, t_1, \ldots, t_N\}$)
1  Sample $\mathbf{x}_0 \sim \mathcal{N}(\mathbf{0}, s(t_0)^2\sigma(t_0)^2 I)$.
2  for $i \leftarrow 0, 1, \ldots, N-1$
3      $\mathbf{d}_i \leftarrow \Big(\frac{\dot\sigma(t_i)}{\sigma(t_i)} + \frac{\dot s(t_i)}{s(t_i)}\Big)\mathbf{x}_i - s(t_i)\frac{\dot\sigma(t_i)}{\sigma(t_i)} D_{\boldsymbol{\theta}}\Big(\frac{\mathbf{x}_i}{s(t_i)}, \sigma(t_i)\Big)$
4      $\mathbf{x}_{i+1} \leftarrow \mathbf{x}_i + (t_{i+1} - t_i)\mathbf{d}_i$
5  return $\mathbf{x}_N$.
```

Algorithm 2 Heun's method for sampling from p_{data} by simulating the probability flow ODE (2).

```
HEUN-SAMPLER($D_{\boldsymbol{\theta}}, \sigma, s, \{t_0, t_1, \ldots, t_N\}$)
1  Sample $\mathbf{x}_0 \sim \mathcal{N}(\mathbf{0}, s(t_0)^2\sigma(t_0)^2 I)$.
2  for $i \leftarrow 0, 1, \ldots, N-1$
3      $\mathbf{d}_i \leftarrow \Big(\frac{\dot\sigma(t_i)}{\sigma(t_i)} + \frac{\dot s(t_i)}{s(t_i)}\Big)\mathbf{x}_i - s(t_i)\frac{\dot\sigma(t_i)}{\sigma(t_i)} D_{\boldsymbol{\theta}}\Big(\frac{\mathbf{x}_i}{s(t_i)}, \sigma(t_i)\Big)$
4      $\mathbf{x}_{i+1} \leftarrow \mathbf{x}_i + (t_{i+1} - t_i)\mathbf{d}_i$
5      if $\sigma(t_{i+1}) \neq 0$
6          $\mathbf{d}'_i \leftarrow \Big(\frac{\dot\sigma(t_{i+1})}{\sigma(t_{i+1})} + \frac{\dot s(t_{i+1})}{s(t_{i+1})}\Big)\mathbf{x}_{i+1} - s(t_{i+1})\frac{\dot\sigma(t_{i+1})}{\sigma(t_{i+1})} D_{\boldsymbol{\theta}}\Big(\frac{\mathbf{x}_{i+1}}{s(t_{i+1})}, \sigma(t_{i+1})\Big)$
7          $\mathbf{x}_{i+1} \leftarrow \mathbf{x}_i + \frac{1}{2}(t_{i+1} - t_i)(\mathbf{d}_i + \mathbf{d}'_i)$
8  return $\mathbf{x}_N$.
```

- We have that

$$\begin{aligned}\nabla_{\mathbf{x}} \log p(\mathbf{x}/s(t); \sigma(t)) &= \nabla_{s(t)\hat{\mathbf{x}}} \log p(\hat{\mathbf{x}}; \sigma(t)) = \frac{1}{s(t)} \nabla_{\hat{\mathbf{x}}} \log p(\hat{\mathbf{x}}; \sigma(t)) \\ &\approx \frac{D_{\boldsymbol{\theta}}(\hat{\mathbf{x}}, \sigma(t)) - \hat{\mathbf{x}}}{s(t)\sigma^2(t)} \\ &= \frac{D_{\boldsymbol{\theta}}(\mathbf{x}/s(t), \sigma(t)) - \mathbf{x}/s(t)}{s(t)\sigma^2(t)}.\end{aligned}$$

- Subtituting the above expression for the score into Equation 1, we have that

$$\begin{aligned}\frac{\mathrm{d}\mathbf{x}}{\mathrm{d}t} &= \frac{\dot s(t)}{s(t)}\mathbf{x} - s(t)^2\sigma(t)\dot\sigma(t)\frac{D_{\boldsymbol{\theta}}(\mathbf{x}/s(t), \sigma(t)) - \mathbf{x}/s(t)}{s(t)\sigma^2(t)} \\ \frac{\mathrm{d}\mathbf{x}}{\mathrm{d}t} &= \frac{\dot s(t)}{s(t)}\mathbf{x} - s(t)\frac{\dot\sigma(t)}{\sigma(t)}\bigg(D_{\boldsymbol{\theta}}\Big(\frac{\mathbf{x}}{s(t)}, \sigma(t)\Big) - \frac{\mathbf{x}}{s(t)}\bigg) \\ \frac{\mathrm{d}\mathbf{x}}{\mathrm{d}t} &= \bigg(\frac{\dot s(t)}{s(t)} + \frac{\dot\sigma(t)}{\sigma(t)}\bigg)\mathbf{x} - s(t)\frac{\dot\sigma(t)}{\sigma(t)}D_{\boldsymbol{\theta}}\bigg(\frac{\mathbf{x}}{s(t)}, \sigma(t)\bigg). \qquad (2)\end{aligned}$$

- There are many algorithms for numerically solving an ODE such as Euler method and Runge–Kutta methods. The pseudocode for the Euler method that solves Equation (2) backward in time is given in Algorithm 1. The paper advocates the use of Heun's method, which is a second order integrator. Its pseudocode is given in Algorithm 2.

- Note that, to use the above algoriths, we must provide a sequence of times $\{t_0, t_1, \ldots, t_N\}$ where

$$T = t_0 > t_1 > t_2 > \cdots > t_{N-1} > t_N = 0.$$

2.5 Stochastic Sampling

- Most previous works on diffusion models insert noise in each step of the sampling process. Let us refer to this type of sampling as "stochastic sampling" as opposed to "deterministic sampling" above.
- The paper goes to a very long derivation (Appendix B.5) that uses heat equation to derive the reverse-time SDE [And82] so that it can derive the stochastic sampling algorithm. Moreover, the derived equation only works for the non-scaled version (i.e., $s(t) = 1$ and $f(t) = 0$) of the forward SDE. The stochastic sampling algorithm is also only for the non-scaled case. The paper claims that the derivation for the scaled case is similar and omits the derivation.
- I found the above treatment annoyhing as it is not complete. I also struggled with the derivation because it uses the heat equation and knowledge about partial differential equations, which I have not studied. So, in this note, I will summarize the results and provide an alternative, simpler derivation in the Appendix.
- The paper states that, for any $\beta(t) : \mathbb{R} \to [0,\infty)$, the SDE
$$\mathrm{d}\mathbf{x} = \underbrace{\bigg(\mathbf{f}(\mathbf{x},t) - \frac{1}{2}g(t)^2\nabla_{\mathbf{x}}\log p_t(\mathbf{x})\bigg)\,\mathrm{d}t}_{\text{probability flow ODE}} + \bigg(\underbrace{\frac{1}{2}\beta(t)g(t)^2\nabla_{\mathbf{x}}\log p_t(\mathbf{x})\,\mathrm{d}t + \sqrt{\beta(t)}g(t)\,\mathrm{d}\mathbf{W}}_{\text{Langevin diffusion SDE}}\bigg). \tag{3}$$
yields the same marginal probability distribution $p_t(\mathbf{x})$ as Equation 7. This is true, and you can find the proof in the Appendix (Theorem 4). The equation gives us multiple ways to simulate Equation (7) forward in time.
- The paper also states that, for any $\beta(t) : \mathbb{R} \to [0,\infty)$, the SDE
$$\mathrm{d}\mathbf{x} = \underbrace{\bigg(\mathbf{f}(\mathbf{x},t) - \frac{1}{2}g(t)^2\nabla_{\mathbf{x}}\log p_t(\mathbf{x})\bigg)\,\mathrm{d}t}_{\text{probability flow ODE}} + \bigg(\underbrace{-\frac{1}{2}\beta(t)g(t)^2\nabla_{\mathbf{x}}\log p_t(\mathbf{x})\,\mathrm{d}t + \sqrt{\beta(t)}g(t)\,\mathrm{d}\overline{\mathbf{W}}}_{\text{Langevin diffusion SDE}}\bigg) \tag{4}$$
where $\overline{\mathbf{W}}$ is the standard d-dimensional Brownian motion that goes backward in time, yields the same marginal $p_t(\mathbf{x})$ as the distribution yielded by Equation (7). This time, however, the evolution of $p_t(\mathbf{x})$ is backward in time.
- Specializing to our particular SDE, we have that $\mathbf{f}(\mathbf{x},t) = f(t)\mathbf{x}$. So,
$$\mathrm{d}\mathbf{x} = \underbrace{\Big(f(t)\mathbf{x} - \frac{1}{2}g(t)^2\nabla_{\mathbf{x}}\log p_t(\mathbf{x})\Big)\,\mathrm{d}t}_{\text{probability flow ODE}} \underbrace{- \frac{1}{2}\beta(t)g(t)^2\nabla_{\mathbf{x}}\log p_t(\mathbf{x})\,\mathrm{d}t}_{\text{deterministic noise decay}} + \underbrace{\sqrt{\beta(t)}g(t)\,\mathrm{d}\overline{\mathbf{W}}}_{\text{noise injection}}$$
Substituting $f(t) = \dot{s}(t)/s(t)$ and $g(t) = s(t)\sqrt{2\sigma(t)\dot{\sigma}(t)}$ and $p_t(\mathbf{x}) = p(\mathbf{x}/s(t);\sigma(t))s(t)^{-d}$, we have that
$$\begin{aligned}
\mathrm{d}\mathbf{x} &= \Big(f(t)\mathbf{x} - \frac{1}{2}g(t)^2\nabla_{\mathbf{x}}\log p_t(\mathbf{x})\Big)\,\mathrm{d}t - \frac{1}{2}\beta(t)g(t)^2\nabla_{\mathbf{x}}\log p_t(\mathbf{x})\,\mathrm{d}t + \sqrt{\beta(t)}g(t)\,\mathrm{d}\overline{\mathbf{W}} \\
\mathrm{d}\mathbf{x} &= \bigg(\frac{\dot{s}(t)}{s(t)}\mathbf{x} - \big(1+\beta(t)\big)s(t)^2\sigma(t)\dot{\sigma}(t)\nabla_{\mathbf{x}}\log p(\mathbf{x}/s(t);\sigma(t))\bigg)\mathrm{d}t \\
&\quad + s(t)\sqrt{2\beta(t)\sigma(t)\dot{\sigma}(t)}\mathrm{d}\overline{\mathbf{W}} \\
\mathrm{d}\mathbf{x} &= \bigg(\frac{\dot{s}(t)}{s(t)}\mathbf{x} - \big(1+\beta(t)\big)s(t)^2\sigma(t)\dot{\sigma}(t)\frac{D_{\boldsymbol{\theta}}(\mathbf{x}/s(t),\sigma(t)) - \mathbf{x}/s(t)}{s(t)\sigma^2(t)}\bigg)\mathrm{d}t \\
&\quad + s(t)\sqrt{2\beta(t)\sigma(t)\dot{\sigma}(t)}\mathrm{d}\overline{\mathbf{W}} \\
&= \bigg[\bigg(\frac{\dot{s}(t)}{s(t)} + \big(1+\beta(t)\big)\frac{\dot{\sigma}(t)}{\sigma(t)}\bigg)\mathbf{x} - \big(1+\beta(t)\big)s(t)\frac{\dot{\sigma}(t)}{\sigma(t)}D_{\boldsymbol{\theta}}\bigg(\frac{\mathbf{x}}{s(t)},\sigma(t)\bigg)\bigg]\mathrm{d}t + s(t)\sqrt{2\beta(t)\sigma(t)\dot{\sigma}(t)}\mathrm{d}\overline{\mathbf{W}}.
\end{aligned} \tag{5}$$

Algorithm 3 Euler–Maruyama method for sampling from p_{data} by simulating the SDE (5).

```
EULER-MARUYAMA-SAMPLER($D_{\boldsymbol{\theta}}, \sigma, s, \{t_0, t_1, \dots, t_N\}$)
1  Sample $\mathbf{x}_0 \sim \mathcal{N}(\mathbf{0}, s(t_0)^2\sigma(t_0)^2 I)$.
2  for $i \leftarrow 0, 1, \dots, N-1$
3      $\mathbf{s}_i \leftarrow (1+\beta(t_i))s(t_i)\frac{\dot\sigma(t_i)}{\sigma(t_i)} D_{\boldsymbol{\theta}}\Big(\frac{\mathbf{x}_i}{s(t_i)}, \sigma(t_i)\Big)$
4      $\mathbf{d}_i \leftarrow \Big(\frac{\dot\sigma(t_i)}{\sigma(t_i)} + (1+\beta(t_i))\frac{\dot s(t_i)}{s(t_i)}\Big)\mathbf{x}_i - \mathbf{s}_i$
5      $\boldsymbol{\xi}_i \sim \mathcal{N}(\mathbf{0}, 2\beta(t)\sigma(t)\dot\sigma(t) I)$.
6      $\mathbf{x}_{i+1} \leftarrow \mathbf{x}_i + (t_{i+1} - t_i)\mathbf{d}_i + \sqrt{|t_{i+1} - t_i|}\boldsymbol{\xi}_i$
7  return $\mathbf{x}_N$.
```

- The paper does not derive Equation (5). It opts for a simple equation where $s(t) = 1$, which we have that

$$\begin{aligned}
\mathrm{d}\mathbf{x} &= \bigg(\frac{\dot s(t)}{s(t)}\mathbf{x} - \big(1+\beta(t)\big)s(t)^2\sigma(t)\dot\sigma(t)\nabla_{\mathbf{x}}\log p(\mathbf{x}/s(t); \sigma(t))\bigg)\mathrm{d}t \\
&\quad + s(t)\sqrt{2\beta(t)\sigma(t)\dot\sigma(t)}\mathrm{d}\overline{\mathbf{W}} \\
\mathrm{d}\mathbf{x} &= -\big(1+\beta(t)\big)\sigma(t)\dot\sigma(t)\nabla_{\mathbf{x}}\log p(\mathbf{x}; \sigma(t))\,\mathrm{d}t + \sqrt{2\beta(t)\sigma(t)\dot\sigma(t)}\mathrm{d}\overline{\mathbf{W}}.
\end{aligned}$$

  Take $\beta(t) := \gamma(t)\frac{\sigma(t)}{\dot\sigma(t)}$. The equation becomes

$$\mathrm{d}\mathbf{x} = -\sigma(t)\dot\sigma(t)\nabla_{\mathbf{x}}\log p(\mathbf{x}; \sigma(t))\,\mathrm{d}t - \gamma(t)\sigma^2(t)\nabla_{\mathbf{x}}\log p(\mathbf{x}; \sigma(t))\,\mathrm{d}t + \sqrt{2\gamma(t)}\sigma(t)\mathrm{d}\overline{\mathbf{W}}. \tag{6}$$

  This matches Equation (6) in the paper if we change the name of γ to β.

- Equation (5) leads to an sampling algorithm based on the Euler–Maruyama method (Algorithm 3).

- The paper also proposes a stochastic sampler based on Heun's method. The implmenetation is little different from that to the Euler–Maruayama method above. The paper adds the noise to the current data item first before performing the steps of Heun's method. See Algorithm 4.

3 Improvements on Deterministic Sampling

- The paper argues that the noise/scaling/time schedule ($\sigma(t)$, $s(t)$, and $\{t_i\}$) should be doupled from the specification of the denoising model $D_{\boldsymbol{\theta}}$.

- It does so by performing experiments on pre-trained models from previous works, only changing the noise/scaling/time schedule. It was able to achieve imprvements.

- The used models include:
  - DDPM++ cont. trained on CIFAR-10 at 32×32 [SSDK$^+$21].
    It is associated with the VP SDE formulation.
  - NCSN++ cont. trained on CIFAR-10 at 32×32 [SSDK$^+$21].
    It is associated with the VE SDE formulation.
  - ADM (dropout) trained on class-conditional ImageNet at 64×64 [DN21].
    It is associated with the "improved DDPM paper" [DN21] and the DDIM paper [SME20].

- The paper advocates for the use of Heun's 2nd order method (Algorithm 2).

Algorithm 4 Huen's method for stochastically sampling from p_{data} by simulating the SDE (5).

```
HEUN-STOCHASTIC-SAMPLER($D_{\boldsymbol{\theta}}, \sigma, s, \{t_0, t_1, \dots, t_N\}$)
1  Sample $\mathbf{x}_0 \sim \mathcal{N}(\mathbf{0}, s(t_0)^2\sigma(t_0)^2 I)$.
2  for $i \leftarrow 0, 1, \dots, N-1$
3      $\boldsymbol{\xi}_i \sim \mathcal{N}(\mathbf{0}, 2\beta(t)\sigma(t)\dot{\sigma}(t)I)$.
4      $\widetilde{\mathbf{x}}_i \leftarrow \mathbf{x}_i + \sqrt{|t_{i+1} - t_i|}\boldsymbol{\xi}_i$
5      $\mathbf{s}_i \leftarrow (1 + \beta(t_i))s(t_i)\frac{\dot{\sigma}(t_i)}{\sigma(t_i)}D_{\boldsymbol{\theta}}\Big(\frac{\widetilde{\mathbf{x}}_i}{s(t_i)}, \sigma(t_i)\Big)$
6      $\mathbf{d}_i \leftarrow \Big(\frac{\dot{\sigma}(t_i)}{\sigma(t_i)} + (1 + \beta(t_i))\frac{\dot{s}(t_i)}{s(t_i)}\Big)\widetilde{\mathbf{x}}_i - \mathbf{s}_i$
7      $\mathbf{x}_{i+1} \leftarrow \widetilde{\mathbf{x}}_i + (t_{i+1} - t_i)\mathbf{d}_i$
8      if $\sigma(t_{i+1}) \neq 0$
9          $\mathbf{d}'_i \leftarrow \Big(\frac{\dot{\sigma}(t_{i+1})}{\sigma(t_{i+1})} + \frac{\dot{s}(t_{i+1})}{s(t_{i+1})}\Big)\mathbf{x}_{i+1} - s(t_{i+1})\frac{\dot{\sigma}(t_{i+1})}{\sigma(t_{i+1})}D_{\boldsymbol{\theta}}\Big(\frac{\mathbf{x}_{i+1}}{s(t_{i+1})}, \sigma(t_{i+1})\Big)$
10         $\mathbf{x}_{i+1} \leftarrow \widetilde{\mathbf{x}}_i + \frac{1}{2}(t_{i+1} - t_i)(\mathbf{d}_i + \mathbf{d}'_i)$
11 return $\mathbf{x}_N$.
```

- The authors argue that it provides the right balance betweeen sample quality and the number of neural function evaluations (NFE).
- From experiments (Figure 2 in the paper), using Heun's methods led to improvment in all casts. FID score dropped faster as a function of the number of NFEs.

• Comments/improvements on the time steps $\{t_i\}$.

- The step size $|t_{i+1} - t_i|$ should decrease monotonically with decreasing σ. (In other words, the steps should become shorter as we more nearer to a finished product.)
- When defining the time steps, the paper first decide on the noise schedule $\sigma_0, \sigma_1, \dots, \sigma_N$ first. Then, it computes $t_i = \sigma^{-1}(\sigma_i)$ for each i where σ is the noise schedule specific to each model (Table 1 in the paper) For the noise schedule, the paper uses

$$\sigma_i = \begin{cases} \big(\sigma_{\text{max}}^{1/\rho} + \frac{i}{N-1}(\sigma_{\text{min}}^{1/\rho} - \sigma_{\text{max}}^{1/\rho})\big)^\rho & 0 \le i < N, \\ 0 & i = N \end{cases}$$

  where ρ is a positive constant.
- ρ controls how much the steps near σ_{min} are shortened at the expense of the longer steps near σ_{max}.
  * $\rho = 3$ nearly equalizes the truncation error at each step.
  * $\rho \in [5, 10]$ performs much better in terms of image quality.
  * The paper uses $\rho = 7$.

• Comments on $\sigma(t)$ and $s(t)$.

- The authors argue that the best choice for these functions is $\sigma(t) = t$ and $s(t) = 1$.
- σ and t becomes interchangeable.
- The probability flow ODE (2) becomes

$$\frac{\mathrm{d}\mathbf{x}}{\mathrm{d}t} = \frac{\mathbf{x} - D_{\boldsymbol{\theta}}(\mathbf{x}, t)}{t}.$$

- A single Euler step to $t = 0$ yields the denoised image $D_{\boldsymbol{\theta}}(\mathbf{x}, t)$.

  - The tangent of the solution trajectory always points toward the denoiser output, so the tangent (i.e., the score) would change slowly with the noise level. The solution trajectory would be largely linear.
  - The paper says that the DDIM already employs this schedule [SME20]. Moreover, when switching the VP and VE models to this noise/scaling schedule, the authors got improved results.

4 Stochastic Sampling

- The authors observe that deterministic sampling tends to lead to poor sample quality than stochastic sampling.
- Setting $s(t) = 1$, the SDE we have to simulate is Equation (6):

$$\mathrm{d}\mathbf{x} = \underbrace{-\sigma(t)\dot{\sigma}(t)\nabla_{\mathbf{x}} \log p(\mathbf{x}; \sigma(t))\, \mathrm{d}t}_{\text{probability flow ODE}} - \underbrace{\underbrace{\beta(t)\sigma^2(t)\nabla_{\mathbf{x}} \log p(\mathbf{x}; \sigma(t))\, \mathrm{d}t}_{\text{detemrinistic noise decay}} + \underbrace{\sqrt{2\beta(t)}\sigma(t)\mathrm{d}\overline{\mathbf{W}}}_{\text{noise injection}}}_{\text{Langevin diffusion SDE}}.$$

  where $\beta : \mathbb{R} \to [0, \infty)$ is a non-negative function. The paper says that $\beta(t)$ is effectively the rate at which existing noise is replaced with new noise.
- The paper says that the Langevin diffusion SDE drives the sample towards the desired distribution, correcting for errors made in earlier sampling steps. This is probabilty the reason why stochastic sampling is useful.
- However, $\beta(t)$ should not be set too large because the Langevin steps would reduce introduce as well. So, the optimal $\beta(t)$ should be determined empirically.
- The paper then introduces a stochastic sampler (Algorithm 2 in the paper) that I had a tough time recovering the SDE it tries to simulate. I guess the closest one would be

$$\mathrm{d}\mathbf{x} = (1 + \beta(t))\frac{\mathbf{x} - D_{\boldsymbol{\theta}(\mathbf{x},t)}}{t} + \sqrt{\beta(t)t}\mathrm{d}\overline{W}$$

  which is Equation (5) with $s(t) = 1$ and $\sigma(t) = t$. The reason why it is not correctly simulating the above equation is because:
  - The noise has to be scaled with something porportional to $\sqrt{t}$. However, the scale of the noise is proportional to $\sqrt{\hat{t}_i^2 - t_i} = \sqrt{(1+\gamma_i)t_i^2 - t_i^2} = \sqrt{\gamma}t_i$
  - The score, multiplied by noise factor $\beta(t) = \gamma_i$, must be **added**, not subtracted from the update. In the paper, the added noise seems to shorten the distance travelled by the deterministic step. It should be the other way around.
- The paper comments on adding noise in general.
  - Excessive noise addition causes loss of details in the generated images.
  - Excessive noise addition also causes a drift towards oversaturated colors at very low and high noise levels. This is not observed in deterministic sampling.
    * The paper solves this by only performing noise addition when $t_i \in [S_{\min}, S_{\max}]$.
    * They also limit the total noise by the S_{churn} parameter.
    * They clamp γ_i variable so that the noise addition never introduces more noise than there is in the image.
    * They set the noise level S_{noise} to be slightly above 1 to avoid loss of details.

- The paper's stochastic sampler outperformed other works, getting good FID score while keeping the NFEs low.
- However, the algorithm also has a number of parameters S_{min}, S_{max}, S_{churn}, and S_{noise}. The authors had to fine-tune these parameters to get close to SOTA FID on ImageNet-64.

5 Precondition and Training

- Especially with the variance exploding setting, the input to $D_{\boldsymbol{\theta}}$ has very large dynamic range. Nevertheless, it is advisable to keep the input and output signal magnitudes fixed to avoid large variation in gradient magnitudes.
- In order to do so, the paper does not represent $D_{\boldsymbol{\theta}}$ as a neural network directly. It is instead a wrapper around another network $F_{\boldsymbol{\theta}}$. In particular, it uses

$$D_{\boldsymbol{\theta}}(\mathbf{x},\sigma) = c_{\text{skip}}(\sigma)\mathbf{x} + c_{\text{out}}(\sigma)F_{\boldsymbol{\theta}}(c_{\text{in}}(\sigma)\mathbf{x}, c_{\text{noise}}(\sigma)).$$

  The formulation incorporates four scaling operations that can be used to make sure that magnitudes remain constant between noise levels.
- When training, a noise level is sampled according to the probability $p_{\text{train}}(\sigma)$. The loss for the noise level σ is multipled by the weight $\lambda(\sigma)$. All in all, the loss becomes

$$E_{\substack{\sigma\sim p_{\text{train}},\\ \mathbf{y}\sim p_{\text{data}},\\ \boldsymbol{\xi}\sim\mathcal{N}(\mathbf{0},\sigma^2 I)}} \bigg[\lambda(\sigma)c_{\text{out}}(\sigma)^2 \bigg\| F_{\boldsymbol{\theta}}(c_{\text{in}}(\sigma)(\mathbf{y}+\boldsymbol{\xi}), c_{\text{noise}(\sigma)}) - \frac{\mathbf{y} - c_{\text{skip}}(\sigma)(\mathbf{y}+\boldsymbol{\xi})}{c_{\text{out}(\sigma)}}\bigg\|^2\bigg].$$

- To facilitate further discussion, define

$$\begin{aligned} w(\sigma) &:= \lambda(\sigma)c_{\text{out}}(\sigma)^2, \\ F_{\text{target}}(\mathbf{y},\boldsymbol{\xi};\sigma) &:= \frac{\mathbf{y} - c_{\text{skip}}(\mathbf{y}+\boldsymbol{\xi})}{c_{\text{out}}(\sigma)} \end{aligned}$$

  So, the loss function becomes

$$E_{\substack{\sigma\sim p_{\text{train}},\\ \mathbf{y}\sim p_{\text{data}},\\ \boldsymbol{\xi}\sim\mathcal{N}(\mathbf{0},\sigma^2 I)}} \Big[w(\sigma)\Big\|F_{\boldsymbol{\theta}}(c_{\text{in}}(\sigma)(\mathbf{y}+\boldsymbol{\xi}), c_{\text{noise}(\sigma)}) - F_{\text{target}}(\mathbf{y},\boldsymbol{\xi};\sigma)\Big\|^2\Big].$$

- The paper picks c_{in}, c_{out}, c_{skip}, and λ as follows.
  - The training inputs to $F_{\boldsymbol{\theta}}$ should have unit variance. So,

$$c_{\text{in}}(\sigma) = 1/\sqrt{\sigma^2 + \sigma_{data}^2}.$$

  - The training target F_{target} should also have unit variance. So,

$$c_{\text{out}}(\sigma)^2 = (1 - c_{\text{skip}}(\sigma))^2\sigma_{\text{data}}^2 + c_{\text{skip}}(\sigma)^2\sigma^2.$$

  - c_{skip} is selected to minimize c_{out} so that the errors of $F_{\boldsymbol{\theta}}$ are amplified as little as possible. This gives

$$\begin{aligned} c_{\text{skip}}(\sigma) &= \operatorname*{arg\,min}_{c_{\text{skip}}(\sigma)} \Big\{c_{\text{out}}(\sigma)^2\Big\} = \frac{\sigma_{\text{data}}^2}{\sigma^2 + \sigma_{\text{data}}^2}, \\ c_{\text{out}}(\sigma) &= \frac{\sigma\cdot\sigma_{\text{data}}}{\sqrt{\sigma^2+\sigma_{\text{data}}^2}}. \end{aligned}$$

  - The effective weight $\lambda(\sigma)$ is required to be uniform ($= 1$) across noise levels. So,

$$\lambda(\sigma) = \frac{\sigma^2 + \sigma_{\text{data}}^2}{(\sigma \cdot \sigma_{data})^2}$$

- The formula for c_{noise} was chosen empirically. The paper uses

$$c_{\text{noise}}(\sigma) = \frac{1}{4} \ln \sigma$$

  for its best algorithm.

- Lastly, we have to specify the probability distribution for sampling σ.
  - After training, the authors observed the training loss as a function of σ. They observe a U-shaped curve.
    * At low noise level, the corrupted image is already very closed to the source image. So, it is very difficult (and meaningless) to reduce noise here.
    * At high noise level, the training targets is very different from the noisy images.
  - As a result, it makes sense to concentrate efforts on the middle noise levels, where more improvements are more possible.
  - The paper picks a simple log-normal distribution for $p_{\text{train}}(\sigma)$. (In the paper, this looks like a bell curve.)
- After the improvement in training, the benefit of stochastic sampling diminishes.
  - For models trained on CIFAR-10, deterministic sampling performed better than stochastic sampling.
  - However, for class-conditional ImageNet-64, stochastic sampling still beats deterministic sampling. However, only a low amount of noise injection was needed, and the authors said the level was lower then they had expected.
  - So, more diverse datasets might benefit more from stochastic sampling.
- With all the improvements on sampling algorithm, noise schedule, conditioning, and training, the paper's ImageNet-64 model achieved SOTA FID score.

A Mathematical Facts and Some Derivations

- **Proposition 2.** *The solution to the initial value problem*

$$\begin{aligned} \frac{\mathrm{d}v(t)}{\mathrm{d}t} &= 2f(t)v(t) + g(t)^2 \\ v(0) &= 0 \end{aligned}$$

  is

$$v(t) = s(t)^2 \int_0^t \frac{g(u)^2}{s(u)^2}\,\mathrm{d}u$$

  where $s(t) = \exp(\int_0^t f(u)\,\mathrm{d}u)$.

Proof.

$$\begin{aligned}\frac{\mathrm{d}v(t)}{\mathrm{d}t} &= 2f(t)v(t) + g(t)^2 \\ \frac{\mathrm{d}v(t)}{\mathrm{d}t} - 2f(t)v(t) &= g(t)^2.\end{aligned}$$

Let $s(t) = \int_0^t f(u)\,\mathrm{d}u$. Multiplying both sides by $s(t)^{-2} = \exp(-2\int_0^t f(u)\,\mathrm{d}t)$, we have that

$$\begin{aligned}\exp\bigg(-2\int_0^t f(u)\,\mathrm{d}t\bigg)\frac{\mathrm{d}v(t)}{\mathrm{d}t} - 2f(t)\exp\bigg(-2\int_0^t f(u)\,\mathrm{d}t\bigg)v(t) &= \exp\bigg(-2\int_0^t f(u)\,\mathrm{d}t\bigg)g(t)^2 \\ \bigg\{\exp\bigg(-2\int_0^t f(u)\,\mathrm{d}t\bigg)v(t)\bigg\}' &= \exp\bigg(-2\int_0^t f(u)\,\mathrm{d}t\bigg)g(t)^2 \\ \bigg\{\frac{v(t)}{s(t)^2}\bigg\}' &= \frac{g(t)^2}{s(t)^2} \\ \frac{v(t)}{s(t)^2} - \frac{v(0)}{s(0)^2} &= \int_0^t \frac{g(u)^2}{s(u)^2}\,\mathrm{d}u.\end{aligned}$$

Because $v(0) = 0$ and $s(0) = 1$, we have that

$$\begin{aligned}\frac{v(t)}{s(t)^2} &= \int_0^t \frac{g(u)^2}{s(u)^2}\,\mathrm{d}u \\ v(t) &= s(t)^2\int_0^t \frac{g(u)^2}{s(u)^2}\,\mathrm{d}u\end{aligned}$$

as required.

- We now switch gear to show that Equation (3) yields the same probability distribution as Equation (7). One way to check whether the equations derived by the paper is correct or not is to see if it satisfied the Fokker–Planck equation.

  Theorem 3. *Let $\{\mathbf{x}(t) : t \geq 0\}$ be a stochastic process that solves the SDE*

  $$\mathrm{d}\mathbf{x} = \mathbf{f}(\mathbf{x}, t)\,\mathrm{d}t + g(t)\,\mathrm{d}\mathbf{W}. \tag{7}$$

  Then, the probability density function $p_t(\mathbf{x})$ of $\mathbf{x}(t)$ satisfies the **Fokker–Planck equation***:*

  $$\frac{\partial p_t(\mathbf{x})}{\partial t} = -\sum_{i=1}^{d}\frac{\partial}{\partial x_i}[f_i(\mathbf{x}, t)p_t(\mathbf{x})] + \frac{1}{2}\sum_{i=1}^{d}\frac{\partial^2}{\partial x_i^2}[g(t)^2 p_t(\mathbf{x})]. \tag{8}$$

- **Theorem 4.** *Let $\beta(t) : \mathbb{R} \to [0, \infty)$ be a non-negative function. Consider the SDE*

  $$\mathrm{d}\mathbf{x} = \underbrace{\bigg(\mathbf{f}(\mathbf{x}, t) - \frac{1}{2}g(t)^2\nabla_{\mathbf{x}}\log p_t(\mathbf{x})\bigg)\,\mathrm{d}t}_{\text{probability flow ODE}} + \bigg(\underbrace{\frac{1}{2}\beta(t)g(t)^2\nabla_{\mathbf{x}}\log p_t(\mathbf{x})\,\mathrm{d}t + \sqrt{\beta(t)}g(t)\,\mathrm{d}\mathbf{W}}_{\text{Langevin diffusion SDE}}\bigg). \tag{9}$$

  The probability density function $p_t(\mathbf{x})$ of its solution $\{\mathbf{x}(t) : t \geq 0\}$ satisfies the Fokker–Planck equation (8).

  Proof. Rewriting the equation above, we have that

  $$\mathrm{d}\mathbf{x} = \bigg(\mathbf{f}(\mathbf{x}, t) - \frac{1}{2}g(t)^2\nabla_{\mathbf{x}}\log p_t(\mathbf{x}) + \frac{1}{2}\beta(t)g(t)^2\nabla_{\mathbf{x}}\log p_t(\mathbf{x})\bigg)\,\mathrm{d}t + \sqrt{\beta(t)}g(t)\,\mathrm{d}\mathbf{W}.$$

Let

$$\hat{\mathbf{f}}(\mathbf{x}, \mathbf{t}) := \mathbf{f}(\mathbf{x}, t) - \frac{1}{2}g(t)^2\nabla_{\mathbf{x}} \log p_t(\mathbf{x}) + \frac{1}{2}\beta(t)g(t)^2\nabla_{\mathbf{x}} \log p_t(\mathbf{x}).$$

We have that

$$\mathrm{d}\mathbf{x} = \hat{\mathbf{f}}(\mathbf{x}, t)\,\mathrm{d}t + \sqrt{\beta(t)}g(t)\,\mathrm{d}\mathbf{W}.$$

The probability density function $p_t(\mathbf{x})$ would satisfy the Fokker–Planck equation

$$\frac{\partial p_t(\mathbf{x})}{\partial t} = -\sum_{i=1}^{d} \frac{\partial}{\partial x_i}[\hat{f}_i(\mathbf{x}, t)p_t(\mathbf{x})] + \frac{1}{2}\sum_{i=1}^{d} \frac{\partial^2}{\partial x_i^2}[\beta(t)g(t)^2 p_t(\mathbf{x})].$$

Now, we have that

$$\hat{f}_i(\mathbf{x}, t) = f_i(\mathbf{x}, t) - \frac{1}{2}g(t)^2\frac{\partial \log p_t(\mathbf{x})}{\partial x_i} + \frac{1}{2}\beta(t)g(t)^2\frac{\partial \log p_t(\mathbf{x})}{\partial x_i}.$$

So,

$$\begin{aligned}
&-\sum_{i=1}^{d} \frac{\partial}{\partial x_i}[\hat{f}_i(\mathbf{x}, t)p_t(\mathbf{x})] \\
&= -\sum_{i=1}^{d} \frac{\partial}{\partial x_i}\left[\left(f_i(\mathbf{x}, t) - \frac{1}{2}g(t)^2\frac{\partial \log p_t(\mathbf{x})}{\partial x_i} + \frac{1}{2}\beta(t)g(t)^2\frac{\partial \log p_t(\mathbf{x})}{\partial x_i}\right)p_t(\mathbf{x})\right] \\
&= -\sum_{i=1}^{d} \frac{\partial}{\partial x_i}\left[\left(f_i(\mathbf{x}, t) - \frac{1}{2}g(t)^2\frac{\partial \log p_t(\mathbf{x})}{\partial x_i}\right)p_t(\mathbf{x})\right] - \frac{1}{2}\sum_{i=1}^{d} \frac{\partial}{\partial x_i}\left[\left(\beta(t)g(t)^2\frac{\partial \log p_t(\mathbf{x})}{\partial x_i}\right)p_t(\mathbf{x})\right].
\end{aligned}$$

Next,

$$\frac{1}{2}\sum_{i=1}^{d} \frac{\partial^2}{\partial x_i^2}[\beta(t)g(t)^2 p_t(\mathbf{x})] = \frac{1}{2}\sum_{i=1}^{d} \frac{\partial}{\partial x_i}\left[\beta(t)g(t)^2\frac{\partial p_t(\mathbf{x})}{\partial x_i}\right] = \frac{1}{2}\sum_{i=1}^{d} \frac{\partial}{\partial x_i}\left[\left(\beta(t)g(t)^2\frac{\partial \log p_t(\mathbf{x})}{\partial x_i}\right)p_t(\mathbf{x})\right]$$

Adding the two equations together, we have that

$$\begin{aligned}
&-\sum_{i=1}^{d} \frac{\partial}{\partial x_i}[\hat{f}_i(\mathbf{x}, t)p_t(\mathbf{x})] + \frac{1}{2}\sum_{i=1}^{d} \frac{\partial^2}{\partial x_i^2}[\beta(t)g(t)^2 p_t(\mathbf{x})] \\
&= -\sum_{i=1}^{d} \frac{\partial}{\partial x_i}\left[\left(f_i(\mathbf{x}, t) - \frac{1}{2}g(t)^2\frac{\partial \log p_t(\mathbf{x})}{\partial x_i}\right)p_t(\mathbf{x})\right] \\
&= -\sum_{i=1}^{d} \frac{\partial}{\partial x_i}\left[\left(f_i(\mathbf{x}, t) - \frac{1}{2}\frac{g(t)^2}{p_t(\mathbf{x})}\frac{\partial p_t(\mathbf{x})}{\partial x_i}\right)p_t(\mathbf{x})\right] \\
&= -\sum_{i=1}^{d} \frac{\partial}{\partial x_i}[f_i(\mathbf{x}, t)p_t(\mathbf{x})] + \frac{1}{2}\sum_{i=1}^{d} \frac{\partial}{\partial x_i}\left[g(t)^2\frac{\partial p_t(\mathbf{x})}{\partial x_i}\right] \\
&= -\sum_{i=1}^{d} \frac{\partial}{\partial x_i}[f_i(\mathbf{x}, t)p_t(\mathbf{x})] + \frac{1}{2}\sum_{i=1}^{d} \frac{\partial^2}{\partial x_i^2}[g(t)^2 p_t(\mathbf{x})].
\end{aligned}$$

In other words,

$$\frac{\partial p_t(\mathbf{x})}{\partial t} = -\sum_{i=1}^{d} \frac{\partial}{\partial x_i}[f_i(\mathbf{x}, t)p_t(\mathbf{x})] + \frac{1}{2}\sum_{i=1}^{d} \frac{\partial^2}{\partial x_i^2}[g(t)^2 p_t(\mathbf{x})].$$

as required. □

- A lot more machinery is required to show that the conditional density $p_{t|T}$ corresponding to Equation (4) is equivalent to that corresponding to the standard equation (7). First, we need the backward Kolmogorov equation.

  Theorem 5. *Let $\{\mathbf{x}(t) : t \geq 0\}$ be a stochastic process that satisfies the SDE*

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

  Let $T > 0$. Then, for any $0 \leq t < T$, we have that the conditonal probability $p_{T|t}(\mathbf{x}'|\mathbf{x})$ satisfies the following **backward Kolmogorov equation***:*

  $$-\frac{\partial p_{T|t}(\mathbf{x}'|\mathbf{x})}{\partial t} = \sum_{i=1}^{d} f_i(\mathbf{x}, t)\frac{\partial p_{T|t}(\mathbf{x}'|\mathbf{x})}{\partial x_i} + \frac{1}{2}g(t)^2 \sum_{i=1}^{d} \frac{\partial^2 p_{T|t}(\mathbf{x}'|\mathbf{x})}{\mathrm{d}x_i^2}.$$

- **Theorem 6.** *Let $\{\mathbf{x}(t) : t \geq 0\}$ be a stochastic solution that is the solution of the SDE*

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

  Let $T > 0$. Then, for any $0 \leq t < T$, the conditional distribution $p_{t|T}$ satisfies the **reverse-time Kolmogorov equation***:*

  $$-\frac{\partial p_{t|T}(\mathbf{x}|\mathbf{x}')}{\partial t} = -\sum_{i=1}^{d} \frac{\partial}{\partial x_i}\left[\left(- f_i(\mathbf{x}, t) + g(t)^2 \frac{\partial \log p_t(\mathbf{x})}{\partial x_i}\right) p_{t|T}(\mathbf{x}|\mathbf{x}')\right] + \frac{1}{2}\sum_{i=1}^{d} \frac{\partial^2}{\partial x_i^2}\big[g(t)^2 p_{t|T}(\mathbf{x}|\mathbf{x}')\big].$$

  Proof. Let $0 \leq t < T$. Let

  $$p_{t,T}(\mathbf{x}, \mathbf{x}') = p(\mathbf{x}(t) = \mathbf{x} \wedge \mathbf{x}(T) = \mathbf{x}').$$

  By the definition of conditional probability, we have that

  $$p_{t,T}(\mathbf{x}, \mathbf{x}') = p_t(\mathbf{x}) p_{T|t}(\mathbf{x}'|\mathbf{x}).$$

So,

$$
\begin{aligned}
&\frac{\partial p_{t,T}(\mathbf{x},\mathbf{x}')}{\partial t} \\
&= \frac{\partial}{\partial t}\big(p_{T|t}(\mathbf{x}'|\mathbf{x})p_t(\mathbf{x})\big) \\
&= \frac{\partial p_t(\mathbf{x})}{\partial t}p_{T|t}(\mathbf{x}'|\mathbf{x}) + p_t(\mathbf{x})\frac{\partial p_{T|t}(\mathbf{x}'|\mathbf{x})}{\partial t} \\
&= \bigg(-\sum_{i=1}^{d}\frac{\partial}{\partial x_i}[f_i(\mathbf{x},t)p_t(\mathbf{x})] + \frac{1}{2}\sum_{i=1}^{d}\frac{\partial^2}{\partial x_i^2}[g(t)^2 p_t(\mathbf{x})]\bigg)p_{T|t}(\mathbf{x}'|\mathbf{x}) \\
&\qquad p_t(\mathbf{x})\bigg(-\sum_{i=1}^{d}f_i(\mathbf{x},t)\frac{\partial p_{T|t}(\mathbf{x}'|\mathbf{x})}{\partial x_i} - \frac{1}{2}g(t)^2\sum_{i=1}^{d}\frac{\partial^2 p_{T|t}(\mathbf{x}'|\mathbf{x})}{\mathrm{d}x_i^2}\bigg) \\
&= -\sum_{i=1}^{d}p_{T|t}(\mathbf{x}'|\mathbf{x})\frac{\partial}{\partial x_i}[f_i(\mathbf{x},t)p_t(\mathbf{x})] + \frac{1}{2}\sum_{i=1}^{d}p_{T|t}(\mathbf{x}'|\mathbf{x})\frac{\partial^2}{\partial x_i^2}[g(t)^2 p_t(\mathbf{x})] \\
&\qquad -\sum_{i=1}^{d}[f_i(\mathbf{x},t)p_t(\mathbf{x})]\frac{\partial p_{T|t}(\mathbf{x}'|\mathbf{x})}{\partial x_i} - \frac{1}{2}g(t)^2 p_t(\mathbf{x})\sum_{i=1}^{d}\frac{\partial^2 p_{T|t}(\mathbf{x}'|\mathbf{x})}{\mathrm{d}x_i^2} \\
&= -\sum_{i=1}^{d}\frac{\partial}{\partial x_i}\big[f_i(\mathbf{x},t)p_t(\mathbf{x})p_{T|t}(\mathbf{x}'|\mathbf{x})\big] \\
&\qquad + \frac{1}{2}\sum_{i=1}^{d}\bigg(g(t)^2 p_{T|t}(\mathbf{x}'|\mathbf{x})\frac{\partial^2 p_t(\mathbf{x})}{\partial x_i^2}\bigg) - \frac{1}{2}\sum_{i=1}^{d}\bigg(g(t)^2 p_t(\mathbf{x})\frac{\partial^2 p_{T|t}(\mathbf{x}'|\mathbf{x})}{\partial x_i^2}\bigg) \\
&= -\sum_{i=1}^{d}\frac{\partial}{\partial x_i}\big[f_i(\mathbf{x},t)p_{t,T}(\mathbf{x},\mathbf{x}')\big] \\
&\qquad + \frac{1}{2}\sum_{i=1}^{d}\bigg(g(t)^2 p_{T|t}(\mathbf{x}'|\mathbf{x})\frac{\partial^2 p_t(\mathbf{x})}{\partial x_i^2}\bigg) - \frac{1}{2}\sum_{i=1}^{d}\bigg(g(t)^2 p_t(\mathbf{x})\frac{\partial^2 p_{T|t}(\mathbf{x}'|\mathbf{x})}{\partial x_i^2}\bigg).
\end{aligned}
$$

Now, note that

$$
\begin{aligned}
&-\frac{1}{2}\frac{\partial^2}{\partial x_i^2}\big[g(t)^2 p_{t,T}(\mathbf{x},\mathbf{x}')\big] \\
&= -\frac{1}{2}\frac{\partial^2}{\partial x_i^2}\big[g(t)^2 p_t(\mathbf{x})p_{T,t}(\mathbf{x}'|\mathbf{x})\big] \\
&= -\frac{1}{2}g(t)^2 p_{T|t}(\mathbf{x}',\mathbf{x})\frac{\partial^2 p_t(\mathbf{x})}{\partial x_i^2} - g(t)^2\frac{\partial p_{T|t}(\mathbf{x}',\mathbf{x})}{\partial x_i}\frac{\partial p_t(\mathbf{x})}{\partial x_i} - \frac{1}{2}g(t)^2 p_t(\mathbf{x})\frac{\partial^2 p_{T|t}(\mathbf{x}'|\mathbf{x})}{\partial x_i^2}.
\end{aligned}
$$

So,

$$
\begin{aligned}
&\frac{1}{2}g(t)^2 p_{T|t}(\mathbf{x}',\mathbf{x})\frac{\partial^2 p_t(\mathbf{x})}{\partial x_i^2} - \frac{1}{2}g(t)^2 p_t(\mathbf{x})\frac{\partial^2 p_{T|t}(\mathbf{x}'|\mathbf{x})}{\partial x_i^2} \\
&= -\frac{1}{2}\frac{\partial^2}{\partial x_i^2}\big[g(t)^2 p_{t,T}(\mathbf{x},\mathbf{x}')\big] + g(t)^2\frac{\partial p_{T|t}(\mathbf{x}',\mathbf{x})}{\partial x_i}\frac{\partial p_t(\mathbf{x})}{\partial x_i} + g(t)^2 p_{T|t}(\mathbf{x}',\mathbf{x})\frac{\partial^2 p_t(\mathbf{x})}{\partial x_i^2} \\
&= -\frac{1}{2}\frac{\partial^2}{\partial x_i^2}\big[g(t)^2 p_{t,T}(\mathbf{x},\mathbf{x}')\big] + \frac{\partial}{\partial x_i}\bigg[g(t)^2 p_{T|t}(\mathbf{x}'|\mathbf{x})\frac{\partial p_t(\mathbf{x})}{\partial x_i}\bigg].
\end{aligned}
$$

As a result,

$$\begin{aligned}\frac{\partial p_{t,T}(\mathbf{x},\mathbf{x}')}{\partial t} &= -\sum_{i=1}^{d}\frac{\partial}{\partial x_i}\big[f_i(\mathbf{x},t)p_{t,T}(\mathbf{x},\mathbf{x}')\big] - \frac{1}{2}\sum_{i=1}^{d}\frac{\partial^2}{\partial x_i^2}\big[g(t)^2 p_{t,T}(\mathbf{x},\mathbf{x}')\big] + \sum_{i=1}^{d}\frac{\partial}{\partial x_i}\bigg[g(t)^2 p_{T|t}(\mathbf{x}'|\mathbf{x})\frac{\partial p_T(\mathbf{x})}{\partial x_i}\bigg]\\ &= -\sum_{i=1}^{d}\frac{\partial}{\partial x_i}\big[f_i(\mathbf{x},t)p_{t,T}(\mathbf{x},\mathbf{x}')\big] - \frac{1}{2}\sum_{i=1}^{d}\frac{\partial^2}{\partial x_i^2}\big[g(t)^2 p_{t,T}(\mathbf{x},\mathbf{x}')\big] + \sum_{i=1}^{d}\frac{\partial}{\partial x_i}\bigg[g(t)^2 p_{t,T}(\mathbf{x},\mathbf{x}')\frac{\partial \log p_T(\mathbf{x})}{\partial x_i}\bigg]\\ &= -\sum_{i=1}^{d}\frac{\partial}{\partial x_i}\bigg[\Big(f_i(\mathbf{x},t) - g(t)^2\frac{\partial \log p_t(\mathbf{x})}{\partial x_i}\Big)p_{t,T}(\mathbf{x},\mathbf{x}')\bigg] - \frac{1}{2}\sum_{i=1}^{d}\frac{\partial^2}{\partial x_i^2}\big[g(t)^2 p_{t,T}(\mathbf{x},\mathbf{x}')\big].\end{aligned}$$

Deviding both sizes by $p_T(\mathbf{x}')$, we have that

$$\begin{aligned}\frac{\partial p_{t|T}(\mathbf{x}|\mathbf{x}')}{\partial t} &= -\sum_{i=1}^{d}\frac{\partial}{\partial x_i}\bigg[\Big(f_i(\mathbf{x},t) - g(t)^2\frac{\partial \log p_t(\mathbf{x})}{\partial x_i}\Big)p_{t|T}(\mathbf{x}|\mathbf{x}')\bigg] - \frac{1}{2}\sum_{i=1}^{d}\frac{\partial^2}{\partial x_i^2}\big[g(t)^2 p_{t|T}(\mathbf{x}|\mathbf{x}')\big]\\ -\frac{\partial p_{t|T}(\mathbf{x}|\mathbf{x}')}{\partial t} &= \sum_{i=1}^{d}\frac{\partial}{\partial x_i}\bigg[\Big(f_i(\mathbf{x},t) - g(t)^2\frac{\partial \log p_t(\mathbf{x})}{\partial x_i}\Big)p_{t|T}(\mathbf{x}|\mathbf{x}')\bigg] + \frac{1}{2}\sum_{i=1}^{d}\frac{\partial^2}{\partial x_i^2}\big[g(t)^2 p_{t|T}(\mathbf{x}|\mathbf{x}')\big]\\ -\frac{\partial p_{t|T}(\mathbf{x}|\mathbf{x}')}{\partial t} &= -\sum_{i=1}^{d}\frac{\partial}{\partial x_i}\bigg[\Big(-f_i(\mathbf{x},t) + g(t)^2\frac{\partial \log p_t(\mathbf{x})}{\partial x_i}\Big)p_{t|T}(\mathbf{x}|\mathbf{x}')\bigg] + \frac{1}{2}\sum_{i=1}^{d}\frac{\partial^2}{\partial x_i^2}\big[g(t)^2 p_{t|T}(\mathbf{x}|\mathbf{x}')\big].\end{aligned}$$

We are done. □

- **Theorem 7.** *Let $\{\mathbf{x}(t) : t \geq 0\}$ be the solution to the SDE*

$$\mathrm{d}\mathbf{x} = \big(\mathbf{f}(\mathbf{x},t) - g(t)^2\nabla_{\mathbf{x}}\log p_t(\mathbf{x})\big)\,\mathrm{d}t + g(t)\,\mathrm{d}\overline{\mathbf{W}}$$

where $\overline{\mathbf{W}}$ is the Brownian motion that runs backward in time. Then, the probability density $p_t(\mathbf{x})$ of $\mathbf{x}(t)$ agrees with the probability distribution of the solution of

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

Proof. (Not at all rigourous...) Marginalizing over $\mathbf{x}(T)$, the reverse-time Kolmogorov equation becomes

$$-\frac{\partial p_t(\mathbf{x})}{\partial t} = -\sum_{i=1}^{d}\frac{\partial}{\partial x_i}\bigg[\Big(-f_i(\mathbf{x},t) + g(t)^2\frac{\partial \log p_t(\mathbf{x})}{\partial x_i}\Big)p_t(\mathbf{x})\bigg] + \frac{1}{2}\sum_{i=1}^{d}\frac{\partial^2}{\partial x_i^2}\big[g(t)^2 p_t(\mathbf{x})\big].$$

We now make a substitution $t \to \tau$ with $\tau = T - t$. We have that

$$\frac{\partial p_\tau(\mathbf{x})}{\partial \tau} = -\sum_{i=1}^{d}\frac{\partial}{\partial x_i}\bigg[\Big(-f_i(\mathbf{x},\tau) + g(\tau)^2\frac{\partial \log p_\tau(\mathbf{x})}{\partial x_i}\Big)p_\tau(\mathbf{x})\bigg] + \frac{1}{2}\sum_{i=1}^{d}\frac{\partial^2}{\partial x_i^2}\big[g(\tau)^2 p_\tau(\mathbf{x})\big].$$

The above equation is the Fokker–Planck equation of the SDE

$$\mathrm{d}\mathbf{x} = \Big(-\mathbf{f}(\mathbf{x},\tau) + g(\tau)^2\nabla_{\mathbf{x}}\log p_\tau(\mathbf{x})\Big)\,\mathrm{d}\tau + g(\tau)\,\mathrm{d}\overline{\mathbf{W}}.$$

Making a substution $\tau \to t$ again, we have that

$$\mathrm{d}\mathbf{x} = \Big(\mathbf{f}(\mathbf{x},t) - g(t)^2\nabla_{\mathbf{x}}\log p_t(\mathbf{x})\Big)\,\mathrm{d}t + g(t)\,\mathrm{d}\overline{\mathbf{W}}.$$

Since $p_t(\mathbf{x})$ satisfies the reverse-time Kolmogorov equation, it also satisfies the Fokker–Planck equation, which means that it agrees with the probability density of the solution of $\mathrm{d}\mathbf{x} = \mathbf{f}(\mathbf{x},t)\,\mathrm{d}t + g(t)\,\mathrm{d}\mathbf{W}$. □

References

[And82] Brian D.O. Anderson. Reverse-time diffusion equation models. *Stochastic Processes and their Applications*, 12(3):313–326, 1982.

[DN21] Prafulla Dhariwal and Alexander Nichol. Diffusion models beat gans on image synthesis. In M. Ranzato, A. Beygelzimer, Y. Dauphin, P.S. Liang, and J. Wortman Vaughan, editors, *Advances in Neural Information Processing Systems*, volume 34, pages 8780–8794. Curran Associates, Inc., 2021.

[JMLPT$^+$21] Alexia Jolicoeur-Martineau, Ke Li, Rémi Piché-Taillefer, Tal Kachman, and Ioannis Mitliagkas. Gotta go fast when generating data with score-based models, 2021.

[KAAL22] Tero Karras, Miika Aittala, Timo Aila, and Samuli Laine. Elucidating the design space of diffusion-based generative models, 2022.

[Khu22] Pramook Khungurn. Score-based generative models. `https://pkhungurn.github.io/notes/notes/ml/score-based-generative-models/score-based-generative-models.pdf`, 2022. Accessed: 2023-01-15.

[ND21] Alex Nichol and Prafulla Dhariwal. Improved denoising diffusion probabilistic models, 2021.

[SME20] Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models, 2020.

[SS19] Simo Särkkä and Arno Solin. *Applied Stochastic Differential Equations*. Cambridge University Press, 2019.

[SSDK$^+$21] Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. In *International Conference on Learning Representations*, 2021.