Generative models based on ODE and SDE

Denis Rakitin
rakitindenis32@gmail.com

Contents

1 Introduction

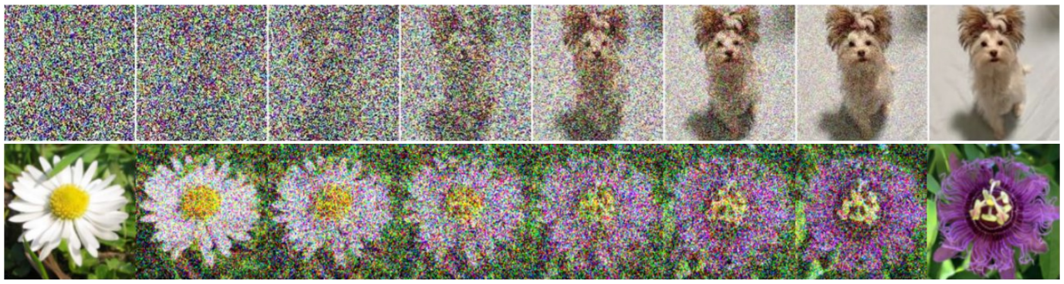

Figure 1: Top: denoising process, defined by a diffusion model [17]. Bottom: stochastic interpolation between two pictures [1].

This tutorial aims to introduce the reader to the mathematical methods which are widely used in contemporary generative modeling. Big practical success of diffusion models [7, 3, 17], which generate a sequence of pictures instead of just the target one, led to development of a novel family of models that describe some dynamic processes. Almost all processes occuring in the real world can be described by differential equations, which will be the basis of the generative models, reviewed here. We will cover methods based on either ordinary (ODE) or stochastic (SDE) differential equations, applicable for generation [17, 11, 18, 2, 1], paired [18, 2, 1, 12] and unpaired [13, 15, 10, 6] domain translation, formalized as an instance of the optimal transport problem.

2 Diffusion models

Generally speaking, diffusion models define a process of step-by-step noising of a picture and try to learn the reverse process, which allows to generate new images starting with a pure noise. There are different approaches for formalizing this concept. The earlier score-based models as NCSN [16] learn score functions of probability distributions at all noise levels and sample with a consequent Langevin dynamics [19] going from larger to smaller noise levels. In [7] the backward denoising process is trained as a latent variable model with a variational distribution on noisy images, corresponding to a forward noising process. Finally, the authors of [17] consider a continuous-time noising SDE and construct the reverse, which can be seen as a unifying framework for previous approaches. On the one hand, the backward SDE explicitly requires knowing score functions at all noise levels, which is similar to the earlier score-based models. On the other hand, discrete-time sampling schemes from the backward SDE result in a discrete-time process very similar to the one in [7]. In this section, we will cover the original score-based approach and models, based on SDE.

2.1 Score-based models

The general problem of generative modeling consists of constructing the probabilistic model p_θ given a data set $X_1, \dots, X_n \sim p_{data}$ in such a way that $p_\theta \approx p_{data}$.

There are mainly 3 families of non-diffusion generative models: Generative Adversarial Networks (GANs) [5], Variational Autoencoders (VAEs) [9] and Normalizing Flows (NFs) [4, 14]. Despite different training procedures, all of them share the same simple generation scheme: picture x is obtained by transforming noise z by a trained network G_θ: $x = G_\theta(z)$. In contrast, score-based models do not learn the generator explicitly, but try to approximate the score function of the distribution: $s_\theta(x) \approx \nabla_x \log p_{data}(x)$. But why is this statistic valuable?

2.1.1 Score function

First, it can be used for finding mode of the distribution by gradient ascent: if

$$X_{t+1} = X_t + \gamma \nabla \log p_{data}(X_t),$$

then

$$X_t \xrightarrow[t\to\infty]{} x* = \arg\max_x \log p_{data}(x) = \arg\max_x p_{data}(x).$$

Of course, in general multi-modal case method converges to one of the stationary points of $\log p_{data}(x)$.

Furthermore, it can be used to obtain samples from the distribution using Langevin dynamics [19]:

$$X_{t+1} = X_t + \gamma \nabla \log p_{data}(X_t) + \sqrt{2\gamma}\, \varepsilon_t, \tag{1}$$

where $\varepsilon_t \sim \mathcal{N}(0, I)$ is independent with X_t. In regular cases, one can prove that

$$p_{X_t} \xrightarrow[t \to \infty]{} p^\gamma,$$

where p^γ is a distribution close to p_{data} that depends on a discretizarion step and converges to p_{data}, when γ converges to zero:

$$p^\gamma \xrightarrow[\gamma \to 0]{} p_{data}.$$

Given this, score-based models obtain sample from the model by running Langevin dynamics with trained approximation of the score function s_θ instead of the true one:

$$X_{t+1} = X_t + \gamma s_\theta(X_t) + \sqrt{2\gamma}\, \varepsilon_t.$$

The only question remained is how to train it. Ideally, we would like to match our approximation with the true score function on the samples from data set and solve the Score Matching objective

$$\mathbb{E}\|s_\theta(X) - \nabla \log p_{data}(X)\|^2 \to \min_\theta, \tag{2}$$

where $X \sim p_{data}$. Unfortunately, this objective has 2 issues:

1. Trivially, we do not know the true score function and cannot compute it to perform regression;
2. It is common to assume that such high-dimensional structured data like images lie in a much less-dimensional manifold. In this case, data distribution p_{data} does not have density in a regular sence. Even if the data distribution is not strictly concentrated on a manifold, but is very close, there will be very rapid transitions from zero to high-density regions, which will result in large unstable values of $\nabla \log p_{data}$.

2.1.2 Denoising score matching

To address the second problem, one can make the data distribution more smooth by, for example, adding gaussian noise:

$$\hat{X} = X + \sigma\varepsilon,\ \varepsilon \sim \mathcal{N}(0, I),$$

or, more generally, deleting part of the object and replacing it with gaussian noise:

$$\hat{X} = \alpha X + \sigma\varepsilon,\ \varepsilon \sim \mathcal{N}(0, I),\ \alpha < 1.$$

In terms of distributions, we defined a conditional distribution

$$p_{\hat{X}|X}(\hat{x}|x) = \mathcal{N}(\hat{x} \mid \alpha x, \sigma^2 I).$$

The perturbed variable $\hat{X}$ always has density, which can be represented as

$$p_{\hat{X}}(\hat{x}) = \int p_{\hat{X}|X}(\hat{x}|x) p_X(x) \mathrm{d}x. \tag{3}$$

Consequently, it has a score function $\nabla \log p_{\hat{X}}(\hat{x})$, which can be theoretically learned by optimizing the score matching objective

$$\mathbb{E}\|s_\theta(\hat{X}) - \nabla \log p_{\hat{X}}(\hat{X})\|^2 \to \min_\theta,$$

which is, however, still intractable.

The representation of density in the Equation 3 is important, because it rewrites a complicated density $p_{\hat{X}}$ as an integral of a very easy conditional density $p_{\hat{X}|X}$ with respect to the distribution p_X, from which we have a data set of samples. I turns out that the score function of $p_{\hat{X}}$ has a similar representation. To derive it, we start with differentiating logarithm (so-called log-derivative trick):

$$\nabla_{\hat{x}} \log p_{\hat{X}}(\hat{x}) = \frac{\nabla_{\hat{x}} p_{\hat{X}}(\hat{x})}{p_{\hat{X}}(\hat{x})}.$$

Next, we use the representation of density $p_{\hat{X}}(\hat{x})$ in Eq. 3 and obtain

$$\frac{\nabla_{\hat{x}} \int p_{\hat{X}|X}(\hat{x}|x) p_X(x) \mathrm{d}x}{p_{\hat{X}}(\hat{x})} = \frac{\int \nabla_{\hat{x}}\, p_{\hat{X}|X}(\hat{x}|x) p_X(x) \mathrm{d}x}{p_{\hat{X}}(\hat{x})}.$$

Finally, we apply the reversed log-derivative trick to $\nabla_{\hat{x}} p_{\hat{X}|X}(\hat{x}|x)$ and obtain

$$\frac{\int \nabla_{\hat{x}} \log p_{\hat{X}|X}(\hat{x}|x) \cdot p_{\hat{X}|X}(\hat{x}|x) p_X(x) \mathrm{d}x}{p_{\hat{X}}(\hat{x})} = \int \nabla_{\hat{x}} \log p_{\hat{X}|X}(\hat{x}|x) \cdot p_{X|\hat{X}}(x|\hat{x}) \mathrm{d}x,$$

which is just the conditional expectation of the conditional score function. Thereby, we obtained:

$$\nabla_{\hat{x}} \log p_{\hat{X}}(\hat{x}) = \mathbb{E}\left[\nabla_{\hat{x}} \log p_{\hat{X}|X}(\hat{X}|X) \mid \hat{X} = \hat{x}\right]. \tag{4}$$

Given this, the score matching objective can be rewritten as

$$\mathbb{E}\left\|s_\theta(\hat{X}) - \mathbb{E}\left[\nabla_{\hat{x}} \log p_{\hat{X}|X}(\hat{X}|X) \mid \hat{X}\right]\right\|^2 \to \min_\theta. \tag{5}$$

Conditional expectation is very convenient to work with since it is the best predictor in the L_2 sense: for all pairs of r.v. (X, Y)

$$g^*(x) = \mathbb{E}[Y \mid X = x] = \arg\min_g \mathbb{E}\|g(X) - \mathbb{E}[Y|X]\|^2 = \arg\min_g \mathbb{E}\|g(X) - Y\|^2,$$

which means that the objective in Eq. 5 is equivalent to the so-called «denoising score matching» objective

$$\mathbb{E}\|s_\theta(\hat{X}) - \nabla_{\hat{x}} \log p_{\hat{X}|X}(\hat{X}|X)\|^2 \to \min_\theta. \tag{6}$$

Surprisingly, by addressing the problem of sharp density transitions, we also implicitly solved the intractability of the objective! The conditional score function is just the score function of the normal distribution, which can be calculated. For d-dimensional normal distribution

$$p_{\hat{X}|X}(\hat{x}|x) = \mathcal{N}(\hat{x}|\alpha x,\, \sigma^2 I) = \frac{1}{(2\pi\sigma^2)^{d/2}} \exp\left(-\frac{1}{2\sigma^2}\|\hat{x} - \alpha x\|^2\right)$$

log-density is equal to

$$\log p_{\hat{X}|X}(\hat{x}|x) = \text{const} - \frac{1}{2\sigma^2}\|\hat{x} - \alpha x\|^2,$$

which gives the score function

$$\nabla_{\hat{x}} \log p_{\hat{X}|X}(\hat{x}|x) = -\frac{1}{\sigma^2}(\hat{x} - \alpha x) = \frac{1}{\sigma^2}(\alpha x - \hat{x}).$$

Finally, we arrive at optimizing the objective

$$\mathbb{E}\left\|s_\theta(\hat{X}) - \frac{1}{\sigma^2}(\alpha X - \hat{X})\right\|^2 \to \min_\theta,$$

which minimal value is obtained at the score function of the perturbed data distribution:

$$s_{\theta^*}(\hat{x}) = \nabla_{\hat{x}} \log p_{\hat{X}}(\hat{x}) = \nabla_{\hat{x}} \log \int p_{data}(x) \mathcal{N}(\hat{x}|\alpha x,\, \sigma^2 I) \mathrm{d}x.$$

To sample from a trained score-based model, one can just apply Langevin dynamics, which is summarized in the Algorithm 1.

Formally, we obtained algorithm which allows to sample from a slightly modified data distribution. In theory, one can obtain a very close approximation of p_{data} by setting $\alpha \approx 1$ and $\sigma^2 \approx 0$. In practice, however, one should keep in mind that when they get close to these values, transitions of density become sharp, score function starts to take large values and the training procedure becomes unstable. This leads to a trade-off between precision of the approximation and stability of the model.

Algorithm 1 Sampling from a score-based model

```
$X^0 \sim p_0$
for $m = 1, \dots, M$ do
    $\varepsilon^{(m)} \sim \mathcal{N}(0, I)$ — independent with $X^{(m-1)}$
    $X^{(m)} = X^{(m-1)} + \gamma_m s_\theta(X^{(m-1)}) + \sqrt{2\gamma_m}\,\varepsilon^{(m)}$        ▷ Langevin step
end for
return $X^{(M)}$
```

2.1.3 Noise Conditional Score Networks

To overcome the necessity of choosing α and σ^2 and balancing between two qualities of the model, authors of [16] present a very elegant idea: consider a sequence of modified variables $\{X_t\}_{t=1}^T$ instead of one $\hat{X}$. This sequence of variables will represent a process of gradual noising of the image and cover the whole spectrum of noisy distributions: from sharp distributions close to p_{data} to the pure noise like $\mathcal{N}(0, I)$. Formally, this sequence should possess 3 qualities:

1. The first variable X_1 should be close to the data distribution: $p_{X_1} \approx p_{data}$. This will ensure that sampling from p_{X_1} will be almost equivalent to sampling from p_{data};
2. The last variable X_T should be a very simple distribution to sample from, for example, $\mathcal{N}(0, \sigma^2)$;
3. Distributions of X_t and X_{t+1} should be close to each other. This will make X_{t+1} a good initialization point for sampling from p_{X_t}.

Below, we will use notation of the form $p_t(x_t) := p_{X_t}(x_t)$ or $p_{t|s}(x_t|x_s) := p_{X_t|X_s}(x_t|x_s)$ to make it shorter. Authors consider a so-called variance-exploding (VE) process and define

$$X_t = X_0 + \sigma_t^2 \varepsilon, \tag{7}$$

where $\varepsilon \sim \mathcal{N}(0, I)$ is independent from X_0 and σ_t is an increasing sequence of variances. Equivalently,

$$p_{t|0}(x_t|x_0) = \mathcal{N}(x_t|x_0, \sigma_t^2 I).$$

Taking $\sigma_0 \approx 0$, $\sigma_t \approx \sigma_{t+1}$ and σ_T^2 of such magnitude, that $p_T \approx \mathcal{N}(0, \sigma_T^2)$, one ensures to match all the 3 requirements. This sequence of distributions corresponds to adding more and more noise to the original distribution, until it becomes indistinguishable from the pure noise with large magnitude.

The more popular process now is the variance preserving process, which defines a Markov chain

$$X_{t+1} = \sqrt{1 - \beta_t} X_t + \sqrt{\beta_t} \varepsilon_t, \tag{8}$$

where $\varepsilon_t \sim \mathcal{N}(0, I)$ is independent from X_t. This defines a conditional distribution given previous time step:

$$p_{t+1|t}(x_{t+1}|x_t) = \mathcal{N}(x_{t+1}|\sqrt{1 - \beta_t} x_t, \beta_t I).$$

Given this, one can calculate the conditional distribution given the initial variable and obtain [7]

$$p_{t|0}(x_t|x_0) = \mathcal{N}(x_t|\alpha_t x_0, \sigma_t^2 I),$$

where $\alpha_t = \sqrt{\prod_{s=1}^t (1 - \beta_s)} \to 0$ and $\sigma_t^2 = 1 - \prod_{s=1}^t (1 - \beta_s) \to 1$ given a proper choice of β_t. Choosing small enough β_t to ensure $p_{t+1} \approx p_t$ and fulfilling $\alpha_T \approx 0$ and $\sigma_T^2 \approx 1$, one will satisfy all the 3 requirements.

Now, given a sequence of modified distributions with easy conditional distributions, one can train score for all of them with denoising score matching:

$$\mathbb{E}\|s_\theta(X_t, t) - \nabla \log p_{t|0}(X_t|X_0)\|^2 \to \min_\theta.$$

In practice, training T neural networks is very inefficient, that is why $s_\theta(x, t)$ is defined as one neural network with two inputs. Besides, scores for different t are connected and the training signal from one time step is beneficial for another. Thus, the final training procdure consists of taking the weighted sum of denoising score matching losses from all the time steps:

$$\sum_{t=1}^{T} \gamma(t)\mathbb{E}\|s_\theta(X_t, t) - \nabla \log p_{t|0}(X_t|X_0)\|^2 \to \min_\theta . \tag{9}$$

This model (paired with the sampling Algorithm 2) is called Noise Conditional Score Network [16]. Theoretically, $s_\theta(x,t)$ matches $\nabla \log p_t(x)$ after training. Sampling procedure consists of the same Langevin dynamics, but now performed sequentially for all of the time steps backwards. Here the 3 properties of the sequence p_t become crucial: generating from p_T is easy, sample from p_t is a good point to start dynamic for p_{t-1} and sample from p_1 is almost a sample from p_{data}. Formally, the sampling procedure is written in the Algorithm 2.

Algorithm 2 Sampling from a Noise Conditional Score Network (NCSN)

```
$X_T^{(M)} \sim p_T$
for $t = T-1, \dots, 1$ do
    $X_t^{(0)} = X_{t+1}^{(M)}$
    for $m = 1, \dots, M$ do                                          ▷ Langevin dynamics for $p_t$
        $\varepsilon_t^{(m)} \sim \mathcal{N}(0, I)$ — independent with $X_t^{(m-1)}$
        $X_t^{(m)} = X_t^{(m-1)} + \gamma_m s_\theta(X_t^{(m-1)}, t) + \sqrt{2\gamma_m}\,\varepsilon_t^{(m)}$      ▷ Langevin step
    end for
end for
return $X_1^{(M)}$
```

We obtained a model with sequential sampling, which needs to use a neural network multiple times. At the same time, it has a very efficient *simulation-free* training procedure, which does not require sampling from the model (and which distinguishes it from energy-based models and continuous normalizing flows). This makes this procedure inefficient in terms of sampling time, but allows to extract a lot of additional information compared to using NN just once. This combination is believed to be very powerful in practice, that is why constructing the analogues for problems other than generation could be very beneficial.

The only major thing that is improved in the newer algorithms is the sampling scheme. It seems like running Langevin dynamics for each time step can be optimized in a way to perform just one step from t to $t-1$. One of the possible ways to derive such sampling scheme is through stochastic differential equations.

2.2 Preliminaries on ODEs and SDEs

2.2.1 Ordinary differential equations

Ordinary differential equation, written as

$$\dot{X} = f(X, t),$$

or, equivalently,

$$\mathrm{d}X_t = f(X_t, t)\mathrm{d}t, \tag{10}$$

defines a set of functions by defining derivative at each point. The most convenient interpretation for us is that t defines time, X_t defines position of a particle at time t and $f(X_t, t)$ represents its velocity vector at time t (physical meaning of the derivative). This interpretation is clearly seen from the Euler scheme, that approximates the solution as:

$$X_{t+h} = X_t + h\dot{X}_t + \overline{o}(h) = X_t + hf(X_t, t) + \overline{o}(h) \approx X_t + hf(X_t, t). \tag{11}$$

Euler scheme tells us that position of the particle after small time h can be obtained by moving particle from the current position X_t along its velocity field $f(X_t, t)$ at a distance, proportional to the change in time h. This approximate solving scheme will allow us to take a first look at SDEs without heavy theory.

Knowing velocity field is not enough to define a unique trajectory of the particle. We also need to know its initial position. The task of solving system

$$\begin{cases} \mathrm{d}X_t = f(X_t, t)\mathrm{d}t\,; \\ X_0 = z \end{cases} \tag{12}$$

of the differential equation and the initial value is called Cauchy problem or initial value problem.

By adding the initial condition $X_0 = z$, which means that the particle starts moving at the point z, we fully defined a physical system. Consequently, we defined a unique trajectory on some time segment. Of course, this is not always the case, but some conditions on the velocity $f(X_t, t)$ guarantee existence and uniqueness of the solution on some time segment $[0, T]$. We will not think about it and will assume existence and uniqueness of the solution.

Euler scheme allows to approximate continuous-time processes with discrete-time ones. But this connection also works in the opposite: given a discrete-time process, one can construct its continuous-time generalization, calculate its derivative and represent it as a solution of the corresponding ODE. Let's take, for example, the variance-preserving process, defined in the Eq. 7, and remove its stochastic part. We will obtain a deterministic process of step-by-step deleting of an object (instead of step-by-step noising):

$$X_{t+1} = \sqrt{1-\beta_t}X_t.$$

The goal is to construct its continuous-time generalization. Let's take now $t \in [0, 1]$, small h and define the connection between X_t and X_{t+h} instead of X_{t+1}. In the original process, coefficient β_t defines portion of object that should be deleted after a unit of time. A logical continuous-time generalization is

$$X_{t+h} = \sqrt{1-h\beta_t}X_t,$$

which says that the deleted portion of the object is proportional to the time spent. Approximating the square root by Taylor expansion and rearranging terms, we obtain:

$$\frac{X_{t+h} - X_t}{h} = \frac{1}{h}\left(\sqrt{1-h\beta_t}X_t - X_t\right) = X_t\frac{1 - \frac{h\beta_t}{2} + \overline{o}(h) - 1}{h} = -\frac{\beta_t}{2}X_t + \overline{o}(1).$$

By taking $h \to 0$, we see that the continuous-time analogue can be defined as the solution of the ODE

$$\mathrm{d}X_t = -\frac{\beta_t}{2}X_t\mathrm{d}t, \tag{13}$$

which is equal to

$$X_t = X_0 \exp\left(-\frac{1}{2}\int\limits_0^t \beta_s \mathrm{d}s\right),$$

which is itself a very natural way to define a deleting process, especially, when taking constant $\beta_t \equiv \beta$:

$$X_t = X_0 \exp\left(-\frac{\beta t}{2}\right).$$

2.2.2 Stochastic differential equations: informal

Stochastic differential equations are far more difficult to define despite their clear physical interpretation. Talking about ODEs, we treated solution of the ODE as a trajectory of the particle that moves under deterministic force. But what if there is also a stochastic force that affects direction of the velocity? This happens, for example, in quantum mechanics, in which some particles move in a purely stochastic way. A natural modification in this case would be to add a stochastic term to the deterministic velocity field and obtain equation of the form

$$\mathrm{d}X_t = f(X_t, t)\mathrm{d}t + \text{randomness}.$$

In discrete-time one can define a purely stochastic trajectory as a random walk. Let X_0 = 0 and $X_1, X_2, \ldots, X_n, \ldots$ be independent variables that define direction of the walk at the corresponding moment: $\mathsf{P}(X_i = 1) = \mathsf{P}(X_i = -1) = 1/2$. Then, position of the random walk at the moment n is $S_n = \sum_{i=1}^{n} X_i$. This example shows that the «derivative» of a random walk, which in discrete time can be naturally defined as $S_{n+1} - S_n$, is just a set of independent variables with $\mathbb{E}\left(S_{n+1} - S_n\right) = 0$ and Var $\left(S_{n+1} - S_n\right) = 1$.

We would like to define a continuous-time analogue of the random walk and its derivative. Given previous observations, the latter seems to be easier to define: let $(Z_t, t \in [0, 1])$ be a set of independent $\mathcal{N}(0, I)$ variables, which is called a white noise process. Then one can define an equation of the form

$$\mathrm{d}X_t = (f(X_t, t) + Z_t)\,\mathrm{d}t.$$

Where's the problem? One should integrate the white noise Z_t over time to obtain the trajectory, but Z_t is nowhere continuous and is not integrable. This motivates to construct the continuous-time random walk first.

Definition 1 *The continuous-time (d-dimensional) random walk is called Wiener process (or process of Brownian motion), is denoted as W_t, and posesses 4 properties:*

1. $W_0 = 0$*;*
2. W_t *is a continuous process;*
3. *It has independent increments: for all time points $t_1 < t_2 < \ldots < t_n$ variables $W_{t_1}, W_{t_2} - W_{t_1}, \ldots, W_{t_n} - W_{t_{n-1}}$ are independent;*
4. *Increments $W_t - W_s$ are normally distributed $\mathcal{N}(0, (t-s)I)$ variables.*

First two properties are clear. Third is the random walk property that tells that the direction of the particle is independent of its current position. Fourth tells that the magnitude of the increments is stationary over time (it only depends on the time difference), which generalizes discrete processes represented as sums of i.i.d. random variables.

Now, given a continuous-time random walk, represented as a Wiener process, one writes a stochastic differential equation as

$$\mathrm{d}X_t = f(X_t, t)\mathrm{d}t + G(X_t, t)\mathrm{d}W_t, \tag{14}$$

where $G(x, t)$ is in general case matrix, called the *diffusion matrix*. Most of the time, we will use a scalar *diffusion coefficient* $g(t)$ instead.

But what is the second term formally? The first idea is to say that $\mathrm{d}W_t = W_t'\mathrm{d}t$. However, Wiener process is almost surely nowhere differentiable, and the previous definition has no sence. One should actually treat this equation in an integral form

$$X_t = X_0 + \int_0^t f(X_s, s)\mathrm{d}s + \int_0^t G(X_s, s)\mathrm{d}W_s \tag{15}$$

and define the latter term (which is called an *Itô integral*) first. However, we will come to this later and now define SDEs by the corresponding discretization scheme. The Euler Scheme, defined in Equation 11, naturally generizes to the Euler-Maruyama scheme for solving SDEs, which we will use as a pseudo-definition:

$$X_{t+h} \approx X_t + hf(X_t, t) + G(X_t, t)(W_{t+h} - W_t) \tag{16}$$

Recursively applying this scheme up to the initial value X_0, one can see that X_t can be represented as a function of increments of the Wiener process before time t, which are independent with $W_{t+h} - W_t$. Given $W_{t+h} - W_t \sim \mathcal{N}(0, h)$, one can rewrite scheme as

$$X_{t+h} \approx X_t + hf(X_t, t) + \sqrt{h}\,G(X_t, t)\varepsilon_t, \tag{17}$$

where $\varepsilon_t \sim \mathcal{N}(0, I)$ is indepedent with X_t.

Given a pseudo-definition, let's move on to the examples. In Equation 13 we defined an ODE, corresponding to a deterministic part of the VP process. Now, armed with a «definition» of SDEs, we can finish and construct continuous-time analogue of the entire VP process. As previously, we replace X_{t+1} with X_{t+h} and β_t with $h\beta_t$ and obtain

$$X_{t+h} = \sqrt{1 - h\beta_t}X_t + \sqrt{h\beta_t}\varepsilon_t,$$

where $\varepsilon_t \sim \mathcal{N}(0, I)$ is independent with X_t. As previously, we use Taylor expansion of the square root and obtain

$$X_{t+h} - X_t = -h\frac{\beta_t}{2}X_t + \overline{o}(h) + \sqrt{h\beta_t}\varepsilon_t \approx -h\frac{\beta_t}{2}X_t + \sqrt{h\beta_t}\varepsilon_t.$$

Comparing it to the Euler-Maruyama scheme, one can see that this is a discretization of the SDE

$$\mathrm{d}X_t = -\frac{\beta_t}{2}X_t\mathrm{d}t + \sqrt{\beta_t}\mathrm{d}W_t, \tag{18}$$

which has many different names: VP-SDE [17], Langevin equation, Ornstein-Uhlenbeck process. Regardless of the name, it defines a continuous-time process of a gradual noising of an object.

But didn't we use any stochastic schemes that look like discretization of an SDE? In Equation 1 we defined Langevin dynamics:

$$X_{t+1} = X_t + \gamma\nabla\log p(X_t) + \sqrt{2\gamma}\varepsilon_t,$$

where $\varepsilon_t \sim \mathcal{N}(0, I)$ is independent with X_t. Denoting neighbour values of the Langevin dynamics as X_t and X_{t+h} and taking step size γ at time t equal to $h\beta_t/2$, we have

$$X_{t+h} = X_t + h\frac{\beta_t}{2}\nabla\log p(X_t) + \sqrt{h\beta_t}\varepsilon_t.$$

As previously, this is an instance of the Euler-Maruyama scheme for the SDE

$$\mathrm{d}X_t = \frac{\beta_t}{2}\nabla\log p(X_t)\mathrm{d}t + \sqrt{\beta_t}\mathrm{d}W_t, \tag{19}$$

which is nothing but a continuous-time Langevin dynamics! Actually, the VP-SDE, which we also said to be called Langevin equation, corresponds to the continuous Langevin dynamics for a distribution p, which score function is equal to $-x$. This is a standard normal distribution, which automatically explains (in not the most obvious way) why the discrete VP process converges to $\mathcal{N}(0, I)$.

2.2.3 Continuity equation

ODEs and SDEs describe evolution of the particle's position over time. One of the fundamental questions that one could ask next is how can we describe evolution of the distribution of the particle? In this section, we will answer this question.

Since SDEs only describe change in the position of the particle, one also needs to define the initial point X_0. Here we assume that it is generated from a distribution p_0 and X_0 is independent with all the noise that comes from the Wiener process. We consider SDEs with scalar diffusion coefficient $g(t)$ and define the corresponding system

$$\begin{cases} \mathrm{d}X_t = f(X_t, t)\mathrm{d}t + g(t)\mathrm{d}W_t, \\ X_0 \sim p_0. \end{cases} \tag{20}$$

Our goal is to describe how the distribution $p_t(x) := p_{X_t}(x)$ of the particle changes in time, which almost always means calculating derivative:

$$\frac{\partial}{\partial t}p_t(x) = ?$$

Since we work with SDEs through the discretization, the most convenient way to analyze this time-derivative is through the limit:

$$\frac{\partial}{\partial t}p_t(x) = \lim_{h\to 0}\frac{1}{h}(p_{t+h}(x) - p_t(x)).$$

Thus, the problem reduces to analyze how density changes after a small time interval. Remember that the variables are connected throught the Euler-Maruyama scheme:

$$X_{t+h} = X_t + hf(X_t, t) + \sqrt{h}g(t)\,\varepsilon_t,$$

where, as always, $\varepsilon_t \sim \mathcal{N}(0, I)$ is independent with X_t. This connection is, in fact, pretty simple: the first ODE part $X_t + hf(X_t, t)$ is nothing but a differentiable function of X_t, for which (if it is bijective and the inverse is differentiable) there is a change of variables formula. The second part consists in adding an independent noise, which changes density by a convolution.

The first part consists of applying the function $\varphi(x) = x + hf(x, t)$ to X_t. Remember that this is an approximation of the function, which solves ODE forward from time t to time $t + h$. ODE theory says that under regularity conditions this function is bijective (uniqueness of the solution) and differentiable in both directions. This allows us to apply the change of variables formula:

$$p_{\varphi(X_t)}(y) = p_{X_t}(\varphi^{-1}(y)) \left|\det \frac{\partial \varphi^{-1}}{\partial y}\right|.$$

The same formula applied in the opposite direction gives

$$p_{X_t}(x) = p_{\varphi(X_t)}(\varphi(x)) \left|\det \frac{\partial \varphi}{\partial x}\right|,$$

which will be more convenient to analyze. We substitute φ and obtain

$$p_{X_t}(x) = p_{\varphi(X_t)}\left(x + hf(x,t)\right) \left|\det\left(I + h\frac{\partial}{\partial x} f(x,t)\right)\right|.$$

Let's simplify the $\det$ term first. We are interested in all terms that are not $\overline{o}(h)$. In the determinant formula for a matrix A one sums terms of the form $\prod_{i=1}^{d} a_{i\sigma_i}$ over all permutations σ. If one of the elements is off the diagonal, it has a pair, which gives h^2 in product and is $\overline{o}(h)$. This means that the only thing remained is the diagonal term and the whole $\det$ expression is equal

$$\prod_{i=1}^{d}\left(1 + h\frac{\partial}{\partial x_i} f_i(x,t)\right) + \overline{o}(h)$$

We see that the only terms that are not $\overline{o}(h)$ after opening brackets are products that contain 0 or 1 h. This gives

$$1 + h\sum_{i=1}^{d} \frac{\partial}{\partial x_i} f_i(x,t) + \overline{o}(h) = 1 + h \operatorname{div} f(x,t) + \overline{o}(h),$$

where div is the divergence operator, which takes a function ψ and returns $\sum_{i=1}^{d} \partial_{x_i}\psi_i(x) = \operatorname{Tr}(\partial_x \psi(x))$. Taking small enough h, one can obtain a positive value, so taking the absolute value is not necessary.

The density part is even easier to deal with: use the Taylor expansion:

$$p_{\varphi(X_t)}(x + hf(x,t)) = p_{\varphi(X_t)}(x) + h\left\langle \nabla p_{\varphi(X_t)}(x),\, f(x,t)\right\rangle + \overline{o}(h).$$

Combining the two parts, we obtain

$$\left(p_{\varphi(X_t)}(x) + h\left\langle \nabla p_{\varphi(X_t)}(x),\, f(x,t)\right\rangle + \overline{o}(h)\right) \cdot \left(1 + h \operatorname{div} f(x,t) + \overline{o}(h)\right) =$$
$$= p_{\varphi(X_t)}(x) + h \cdot \left(\left\langle \nabla p_{\varphi(X_t)}(x),\, f(x,t)\right\rangle + p_{\varphi(X_t)}(x) \operatorname{div} f(x,t)\right) + \overline{o}(h).$$

Expressing scalar product and divergence as sums, we get

$$p_{\varphi(X_t)}(x) + h\sum_{i=1}^{d}\left(f_i(x,t)\frac{\partial}{\partial x_i} p_{\varphi(X_t)}(x) + p_{\varphi(X_t)}(x)\frac{\partial}{\partial x_i} f_i(x,t)\right) + \overline{o}(h).$$

Expression under brackets is nothing but a derivative of a product, which gives

$$p_{\varphi(X_t)}(x) + h\sum_{i=1}^{d}\frac{\partial}{\partial x_i}\left(f_i(x,t) p_{\varphi(X_t)}(x)\right) + \overline{o}(h).$$

The sum is the divergence of the product $f(x,t)p_{\varphi(X_t)}(x)$, which finally gives

$$p_{X_t}(x) = p_{\varphi(X_t)}(x) + h\,\mathrm{div}\Big(f(x,t)p_{\varphi(X_t)}(x)\Big) + \overline{o}(h). \tag{21}$$

Remember that $\varphi(X_t) = X_t + hf(X_t,t)$. If we work with ODE, then $\varphi(X_t) = X_{t+h}$ and

$$p_t(x) = p_{t+h}(x) + h\,\mathrm{div}\Big(f(x,t)p_{t+h}(x)\Big) + \overline{o}(h),$$

which is equivalent to

$$\frac{p_{t+h}(x) - p_t(x)}{h} = -\mathrm{div}\Big(f(x,t)p_{t+h}(x)\Big) + \overline{o}(1).$$

Taking the limit, we obtain

$$\boxed{\frac{\partial}{\partial t}p_t(x) = -\mathrm{div}\Big(f(x,t)p_t(x)\Big)} \tag{22}$$

which is called the *continuity equation* and describes evolution of a particle that moves under ODE.

2.2.4 Fokker-Planck equation

The continuity equation is a very important object which can be investigated separately, but we also need its extension on the SDE case. Previously, we expressed X_{t+h} through the Euler-Maruyama scheme:

$$X_{t+h} = \varphi(X_t) + Y_t,$$

where $Y_t = \sqrt{h}g(t)\,\varepsilon_t$, and found density of $\varphi(X_t)$ in the Equation 21. Since $\varphi(X_t)$ and Y_t are independent, density of X_{t+h} is just a convolution

$$p_{X_{t+h}}(x) = \int p_{\varphi(X_t)}(x-y)p_{Y_t}(y)\mathrm{d}y = \mathbb{E}\,p_{\varphi(X_t)}\big(x - Y_t\big) = \mathbb{E}\,p_{\varphi(X_t)}\big(x - \sqrt{h}g(t)\,\varepsilon_t\big)$$

As always, use the Taylor expansion (up to the second term, since $\sqrt{h}^2 = h$):

$$\mathbb{E}\left(p_{\varphi(X_t)}(x) - \Big\langle \nabla p_{\varphi(X_t)}(x), \sqrt{h}g(t)\,\varepsilon_t\Big\rangle + \frac{1}{2}\Big\langle \nabla^2 p_{\varphi(X_t)}(x)\,\sqrt{h}g(t)\,\varepsilon_t, \sqrt{h}g(t)\,\varepsilon_t\Big\rangle\right) +$$

$$+\mathbb{E}\,\overline{o}\Big(hg^2(t)\|\varepsilon_t\|^2\Big).$$

This expression is large but convenient to work with:

1. The first summand is deterministic, so $\mathbb{E}\,p_{\varphi(X_t)}(x) = p_{\varphi(X_t)}(x)$;
2. The second is a linear function of a zero-mean variable, which is zero:

$$\mathbb{E}\left\langle \nabla p_{\varphi(X_t)}(x), \sqrt{h}g(t)\,\varepsilon_t\right\rangle = \left\langle \nabla p_{\varphi(X_t)}(x), \sqrt{h}g(t)\,\mathbb{E}\varepsilon_t\right\rangle = 0$$

3. The last term under expectation is $\overline{o}(h)$. By interchanging small-$\overline{o}$ and expectation (which is a very rude operation), we obtain $\overline{o}(h)$.

Now, we have a much smaller expression

$$p_{\varphi(X_t)}(x) + \mathbb{E}\left(\frac{1}{2}\Big\langle \nabla^2 p_{\varphi(X_t)}(x)\,\sqrt{h}g(t)\,\varepsilon_t, \sqrt{h}g(t)\,\varepsilon_t\Big\rangle\right) + \overline{o}(h) =$$

$$= p_{\varphi(X_t)}(x) + \frac{hg^2(t)}{2}\mathbb{E}\left\langle \nabla^2 p_{\varphi(X_t)}(x)\varepsilon_t, \varepsilon_t\right\rangle + \overline{o}(h).$$

The term under expectation here is the famous Hutchinson's trace estimator [8]. Rewrite it using a trace cyclic property:

$$\frac{hg^2(t)}{2}\mathbb{E}\left(\varepsilon_t^\top \nabla^2 p_{\varphi(X_t)}(x)\varepsilon_t\right) = \frac{hg^2(t)}{2}\mathbb{E}\,\mathrm{Tr}\Big(\nabla^2 p_{\varphi(X_t)}(x)\varepsilon_t\varepsilon_t^\top\Big).$$

Trace is a linear function, which we can interchange with expectation and obtain

$$\frac{hg^2(t)}{2}\operatorname{Tr}\Big(\nabla^2 p_{\varphi(X_t)}(x)\mathbb{E}\,\varepsilon_t\varepsilon_t^\top\Big).$$

Finally, since $\mathbb{E}\varepsilon_t\varepsilon_t^\top = \operatorname{Var}\varepsilon_t + (\mathbb{E}\varepsilon_t)(\mathbb{E}\varepsilon_t)^\top$ and $\varepsilon_t \sim \mathcal{N}(0, I)$, it is equal to $\operatorname{Var}\varepsilon_t = I$. We obtained

$$p_{X_{t+h}}(x) = p_{\varphi(X_t)}(x) + \frac{hg^2(t)}{2}\operatorname{Tr}\Big(\nabla^2 p_{\varphi(X_t)}(x)\Big) + \overline{o}(h) =$$
$$= p_{\varphi(X_t)}(x) + \frac{hg^2(t)}{2}\Delta p_{\varphi(X_t)}(x) + \overline{o}(h),$$

where $\Delta\psi(x) = \sum_{i=1}^d \partial^2_{x_i x_i}\psi(x)$ is the Laplace operator. Rewriting density of $\varphi(X_t)$ as we expressed in the Equation 21, we obtain

$$p_{X_{t+h}}(x) = p_{X_t}(x) - h\operatorname{div}\Big(f(x,t)p_{\varphi(X_t)}(x)\Big) + \frac{hg^2(t)}{2}\Delta p_{\varphi(X_t)}(x) + \overline{o}(h),$$

which is equivalent to

$$\frac{p_{t+h}(x) - p_t(x)}{h} = -\operatorname{div}\Big(f(x,t)p_{\varphi(X_t)}(x)\Big) + \frac{g^2(t)}{2}\Delta p_{\varphi(X_t)}(x) + \overline{o}(1).$$

Since $\varphi(X_t) = X_t + hf(X_t, t) \xrightarrow[h\to 0]{} X_t$, we take the limit $h \to 0$ and obtain

$$\boxed{\frac{\partial}{\partial t}p_t(x) = -\operatorname{div}\Big(f(x,t)p_t(x)\Big) + \frac{g^2(t)}{2}\Delta p_t(x)} \tag{23}$$

which is called the *Fokker-Planck equation* and describes how distribution of the particle, driven by SDE, evolves in time.

References

[1] Michael S Albergo, Nicholas M Boffi, and Eric Vanden-Eijnden. Stochastic interpolants: A unifying framework for flows and diffusions. *arXiv preprint arXiv:2303.08797*, 2023.

[2] Michael S Albergo and Eric Vanden-Eijnden. Building normalizing flows with stochastic interpolants. *arXiv preprint arXiv:2209.15571*, 2022.

[3] Valentin De Bortoli, James Thornton, Jeremy Heng, and Arnaud Doucet. Diffusion schrödinger bridge with applications to score-based generative modeling. *Advances in Neural Information Processing Systems*, 34:17695–17709, 2021.

[4] Laurent Dinh, David Krueger, and Yoshua Bengio. Nice: Non-linear independent components estimation. *arXiv preprint arXiv:1410.8516*, 2014.

[5] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. *Advances in neural information processing systems*, 27, 2014.

[6] Nikita Gushchin, Alexander Kolesov, Alexander Korotin, Dmitry Vetrov, and Evgeny Burnaev. Entropic neural optimal transport via diffusion processes. *arXiv preprint arXiv:2211.01156*, 2022.

[7] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in Neural Information Processing Systems*, 33:6840–6851, 2020.

[8] Michael F Hutchinson. A stochastic estimator of the trace of the influence matrix for laplacian smoothing splines. *Communications in Statistics-Simulation and Computation*, 18(3):1059–1076, 1989.

[9] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

[10] Alexander Korotin, Daniil Selikhanovych, and Evgeny Burnaev. Neural optimal transport. *arXiv preprint arXiv:2201.12220*, 2022.

[11] Yaron Lipman, Ricky TQ Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. Flow matching for generative modeling. *arXiv preprint arXiv:2210.02747*, 2022.

[12] Guan-Horng Liu, Arash Vahdat, De-An Huang, Evangelos A Theodorou, Weili Nie, and Anima Anandkumar. I^2 sb: Image-to-image schr\" odinger bridge. *arXiv preprint arXiv:2302.05872*, 2023.

[13] Xingchao Liu, Chengyue Gong, and Qiang Liu. Flow straight and fast: Learning to generate and transfer data with rectified flow. *arXiv preprint arXiv:2209.03003*, 2022.

[14] Danilo Rezende and Shakir Mohamed. Variational inference with normalizing flows. In *International conference on machine learning*, pages 1530–1538. PMLR, 2015.

[15] Yuyang Shi, Valentin De Bortoli, Andrew Campbell, and Arnaud Doucet. Diffusion schr\" odinger bridge matching. *arXiv preprint arXiv:2303.16852*, 2023.

[16] Yang Song and Stefano Ermon. Generative modeling by estimating gradients of the data distribution. *Advances in neural information processing systems*, 32, 2019.

[17] Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. *arXiv preprint arXiv:2011.13456*, 2020.

[18] Alexander Tong, Nikolay Malkin, Guillaume Huguet, Yanlei Zhang, Jarrid Rector-Brooks, Kilian Fatras, Guy Wolf, and Yoshua Bengio. Conditional flow matching: Simulation-free dynamic optimal transport. *arXiv preprint arXiv:2302.00482*, 2023.

[19] Max Welling and Yee W Teh. Bayesian learning via stochastic gradient langevin dynamics. In *Proceedings of the 28th international conference on machine learning (ICML-11)*, pages 681–688, 2011.