Tutorial on
Stochastic differential equations and Generative modelling

An Vuong - vuonga2@oregonstate.edu

July 2024

Abstract

Generative modelling is the problem of sampling from an unknown distribution where the data samples live in, which can be seen as an estimation problem where we want to estimate the true data distribution from the collected samples. While there have been many classical methods dating back to 1950s trying to tackle this problem (energy-based algorithms, importance sampling, Monte Carlo Markov Chain (MCMC), etc.), they all fail to scale as the number of data points increase. This is mainly due to the fact that these algorithms try to estimate the normalizing factor (or the partition function as known in Physics), whose calculations quickly become intractable as the number of samples or dimensions grow. In recent years, a new direction in generative modelling that can bypass this intractable calculations has emerged and achieved state-of-the-art image quality, this method is based on the study of Stochastic differential equations (SDEs) and the famous Fokker-Planck equation, it is aptly named Diffusion models. In this short tutorial, we will provide the overview of generative modelling, with the main focus on Diffusion models from the perspective of SDEs: using the Itô's process as the forward process, then applying Fokker-Planck equation to obtain the reverse process, which is used to generate new samples via ancestral sampling.

Keywords: diffusion process, stochastic differential equations, generative modelling, Fokker-Planck, sampling, estimation, optimization

1 Structure of the paper

First section of this paper will review the background needed for the development of generative modelling using SDEs. We first recall the definition of ODEs, and simple solutions for the case of linear systems. Then we introduce the basis of SDEs, from the perspective of ODEs where the input is driven by noise. Here, we discuss why simply taking the integration of SDEs will not yield the correct solutions, at least in the traditional sense. Building on this, the idea of Itô's calculus is introduced as well as some discussion on existence and uniqueness of the solutions. To finish off the background material, we recall the important Fokker-Planck equation (or Kolmogorov's forward equation) that is used to characterized the probability density induced by SDEs. We use this result to prove an important result that is widely applied in generative modelling, the existence of time-reversal SDEs.

The second section give a deep review of the seminal works of [1][2][3] that kicked start this whole field in 2019, we give detailed derivations and expression for the solutions to the forward SDEs used in these methods, with some toy examples for demonstration:

1. Langevin dynamics sampling and Score-matching
2. Variance exploding SDEs
3. Variance preserving SDEs

2 Stochastic differential equations and Itô's calculus

In this section, a brief review of Stochastic differential equations (SDEs) will be provided. It will be developed by considering Ordinary differential equations (ODEs) driven by noise. Then the integrability of these equations will

be discussed, where the Itô's Lemma and Itô's integral are introduced. Finally, the famous Fokker-Planck equation is derived and applied to generate the time-reversal SDEs, which will play a major role in the later parts.

An in-depth treatment of this subject can be found in [4].

2.1 ODEs and solutions to ODEs

While Ordinary differential equations is a rich and sophisticated mathematical field in its own right, here, for our purposes of building up to diffusion process, we only consider the following simple system:

$$\frac{d\mathbf{x}(t)}{dt} = \mathbf{f}(\mathbf{x}(t), t) + \mathbf{L}(\mathbf{x}(t), t)\mathbf{w}(t) \tag{2.1.1}$$

where $\mathbf{x}(\cdot)$ is a vector-valued function mapping time to state, $\mathbf{f}(\cdot,\cdot)$, $\mathbf{L}(\cdot,\cdot)$ are some arbitrary functions, and $\mathbf{w}(\cdot)$ can be consider as external driving force, or an input to the system. Note that in other ODEs literature $\mathbf{L}(\mathbf{x}(t), t)\mathbf{w}(t)$ can be absorbed into $\mathbf{f}(\mathbf{x}(t), t)$ which leads to a shorter representation:

$$\frac{d\mathbf{x}(t)}{dt} = \mathbf{f}(\mathbf{x}(t), t) \tag{2.1.2}$$

Here, we will work with the formulation in (2.1.1), since this will be more useful when dealing with SDEs. Note that ODEs can also take high-order forms, for e.g, let us consider a second-order 1-dimensional ODE:

$$\frac{d^2x(t)}{dt^2} + \gamma\frac{dx(t)}{dt} + \beta x(t) = w(t) \tag{2.1.3}$$

This second-order ODE can be rewritten to take the form of (2.1.1) by setting:

$$\mathbf{x}(t) = \begin{bmatrix} x(t) \\ \dot{x}(t) \end{bmatrix} \tag{2.1.4}$$

where $\dot{x}(t) = \frac{dx(t)}{dt}$, using this, (2.1.3) can be rewritten as:

$$\underbrace{\begin{bmatrix} \frac{dx(t)}{dt} \\ \frac{d\dot{x}(t)}{dt} \end{bmatrix}}_{\frac{d\mathbf{x}(t)}{dt}} = \underbrace{\begin{bmatrix} 0 & 1 \\ -\beta & -\gamma \end{bmatrix}}_{\mathbf{f}(\mathbf{x}(t),t)} \begin{bmatrix} x(t) \\ \dot{x}(t) \end{bmatrix} + \underbrace{\begin{bmatrix} 0 \\ 1 \end{bmatrix}}_{\mathbf{L}(\mathbf{x}(t),t)} w(t) \tag{2.1.5}$$

which follows the form of (2.1.1). In fact, almost all higher-order ODEs can be rewritten as a system of first-order ODEs following similar procedure. Thus, for most cases we only need to deal with first-order formulation.

Next, we will discuss the solutions to ODEs. In general, it is not possible to find closed-form solutions to ODEs problems, even when the system is linear, i.e. $\mathbf{f}(\mathbf{x}(t), t) = \mathbf{F}(t)\mathbf{x}(t)$. That being said, for linear cases, the solutions to ODEs has the following form:

$$\mathbf{x}(t) = \mathbf{\Psi}(t, t_0)\mathbf{x}(t_0) + \int_{t_0}^{t} \mathbf{\Psi}(t, \tau)\mathbf{L}(\tau)\mathbf{w}(\tau)d\tau \tag{2.1.6}$$

where $\mathbf{\Psi}(t, t_0)$ is the state transition matrix, which has the following properties:

$$\frac{\partial\mathbf{\Psi}(\tau, t)}{\partial\tau} = \mathbf{F}(\tau)\mathbf{\Psi}(\tau, t) \tag{2.1.7}$$

$$\frac{\partial\mathbf{\Psi}(\tau, t)}{\partial t} = -\mathbf{\Psi}(\tau, t)\mathbf{F}(\tau) \tag{2.1.8}$$

$$\mathbf{\Psi}(\tau, t) = \mathbf{\Psi}(\tau, s)\mathbf{\Psi}(s, t) \tag{2.1.9}$$

$$\mathbf{\Psi}(\tau, t) = \mathbf{\Psi}^{-1}(t, \tau) \tag{2.1.10}$$

$$\mathbf{\Psi}(t, t) = \mathbf{I} \tag{2.1.11}$$

Thus, if one knows $\mathbf{\Psi}(t, t_0)$, the solutions to a linear ODEs is readily obtained. For the non-linear cases, it gets more complicated, but if both $\mathbf{f}(t)$ and $\mathbf{L}(t)$ are well-behaved, the Taylor series can be used to linearize the ODEs around some operating points, then the solutions can be approximated by (2.1.6). In general, $\mathbf{\Psi}(t, t_0)$ does not have a closed-form expression, but it can be numerically computed to arbitrary precision by using the Peano-Baker series. More in-depth discussion of this can be found in [5], and a worked out example is included in Appendix A.5.1. Here we will cite a well-known solution to the Linear Time-invariant (LTI) case, i.e. $\mathbf{F}(t) = \mathbf{F}, \mathbf{L}(\mathbf{x}(t), t) = \mathbf{L}$:

Example 1. *The solution to the LTI ODE*

$$\frac{d\mathbf{x}(t)}{dt} = \mathbf{F}\mathbf{x}(t) + \mathbf{L}\mathbf{w}(t) \tag{2.1.12}$$

has the following form

$$\mathbf{x}(t) = e^{\mathbf{F}(t-t_0)}\mathbf{x}(t_0) + \int_{t_0}^{t} \mathbf{F}(t-\tau)\mathbf{L}\mathbf{w}(\tau)d\tau \tag{2.1.13}$$

Note that the state transition matrix $\mathbf{\Psi}(t, t_0) = e^{\mathbf{F}(t-t_0)}$

As a preparation for the next section, we now briefly discuss the existence and uniqueness of the solutions to ODEs:

Theorem 1. *(Picard-Lindelof) The solution to the ODE*

$$\frac{d\mathbf{x}(t)}{dt} = \mathbf{f}(\mathbf{x}(t), t), \qquad \mathbf{x}(t_0) = \mathbf{x}_0 \tag{2.1.14}$$

exists and is unique if $\mathbf{f}(\mathbf{x}(t), t)$ *is continuous in both arguments and is Lipschitz continuous in* $\mathbf{x}$*.*

Note that the theorem uses the formulation in (2.1.2). Applying to (2.1.1), this means the continuity requirement needs to be satisfied by both $\mathbf{f}(\cdot, \cdot)$ and $\mathbf{L}(\cdot, \cdot)$. As we will see in the next section, the condition of being continuous in t is problematic for SDEs, since the stochastic sample paths will be discontinuous almost everywhere.

2.2 SDEs and solutions to SDEs

Having defined ODEs, a straightforward approach to introduce SDEs is by considering the input/driving term $\mathbf{w}(t)$ from (2.1.1) as some stochastic processes. For example, recall the the definition of ODE

$$\frac{d\mathbf{x}(t)}{dt} = \mathbf{f}(\mathbf{x}(t), t) + \mathbf{L}(\mathbf{x}(t), t)\mathbf{w}(t) \tag{2.2.1}$$

if we let $\mathbf{w}(t)$ to be some white noise process, then (2.2.1) is a Stochastic differential equation. Followed is the definition of $\mathbf{w}(t)$

Definition 1. *A random process* $\mathbf{w}(t) : \mathbb{R} \longrightarrow \mathbb{R}^d$ *is called white noise if the following properties are satisfied:*

1. *Independent:* $\mathbf{w}(t) \perp\!\!\!\perp \mathbf{w}(t+\epsilon), \epsilon \neq 0$
2. *Zero mean:* $\mathbf{E}[\mathbf{w}(t)] = \mathbf{0}$
3. *(Wide-Sense) Stationary:* $\mathbf{E}[\mathbf{w}(t_1)\mathbf{w}^T(t_2)] = \delta(t_1 - t_2)$

From this definition, it is clear that the sample path of the process $\mathbf{w}(t)$ is discontinuous almost everywhere due to the independent property. This will lead to a problem when we try to apply the methods of ODEs to SDEs. For now, let us consider the LTI case and naively apply the solution of ODEs from Example 1

Example 2. *The (naive) solution to the SDEs*

$$\frac{d\mathbf{x}(t)}{dt} = \mathbf{F}\mathbf{x}(t) + \mathbf{L}\mathbf{w}(t) \tag{2.2.2}$$

where $\mathbf{w}(t)$ *is a white noise process has the form*

$$\mathbf{x}(t) = e^{\mathbf{F}(t-t_0)}\mathbf{x}(t_0) + \int_{t_0}^{t} \mathbf{F}(t-\tau)\mathbf{L}\mathbf{w}(\tau)d\tau \tag{2.2.3}$$

The first term on the right hand side of (2) is deterministic given $\mathbf{x}(t_0)$, the second term is stochastic due to the randomness of $\mathbf{w}(t)$. This is expected since the solution to SDEs should also be stochastic processes due to the stochasticity in the driving force term. But if we take a closer look at the stochastic integral, an immediate question arises: how can we perform the integration? Since the sample path generated by $\mathbf{w}(t)$ is discontinuous almost everywhere, the Riemann sum does not converge. More generally, if we try to integrate (2.1.1), where $\mathbf{w}(t)$ is white noise

$$\int \frac{d\mathbf{x}(t)}{dt} = \int \mathbf{f}(\mathbf{x}(t), t) + \int \mathbf{L}(\mathbf{x}(t), t)\mathbf{w}(t) \tag{2.2.4}$$

$$\mathbf{x}(t) - \mathbf{x}(0) = \int \mathbf{f}(\mathbf{x}(t), t)dt + \int \mathbf{L}(\mathbf{x}(t), t)\mathbf{w}(t)dt \tag{2.2.5}$$

then the second integral on the right-hand side, $\int \mathbf{L}(\mathbf{x}(t), t)\mathbf{w}(t)dt$ is undefined in the traditional sense, proof sketch is provided in Appendix A.5.2. This calls for a new integration method to deal with random processes. The method, first introduced in [6], is now widely known as the Itô's integral, the field dealing with this kind of integration is generally referred to as Itô's calculus.

2.2.1 Itô's calculus

To construct the Itô's integral, first we can think of $\mathbf{w}(t)dt$ as an infinitesimal increment of some other processes

$$\mathbf{w}(t)dt = d\boldsymbol{\beta}(t) \tag{2.2.6}$$

Since $\mathbf{w}(t)$ is white noise, $\boldsymbol{\beta}(t)$ has to satisfy the following properties:

Definition 2. *(Brownian motion)*

1. *Independent increment:* $\Delta\boldsymbol{\beta}_k = \boldsymbol{\beta}(t_{k+1}) - \boldsymbol{\beta}(t_k) \perp\!\!\!\perp \Delta\boldsymbol{\beta}_{k'}$ *if the intervals are non-overlapping*
2. *Zero-mean Gaussian increment:* $\Delta\boldsymbol{\beta}_k \sim \mathcal{N}\big(\mathbf{0}, \Delta t_k\big)$
3. *Starts at origin:* $\boldsymbol{\beta}(0) = \mathbf{0}$

The first two conditions come directly from the fact that $\mathbf{w}(t)$ is white noise, the last one is a technical initial condition. Turns out these properties define a well-studied random process called Brownian motion. From this definition, there are a few peculiar observations about Brownian motion:

1. $\boldsymbol{\beta}(t)$ is non-differentiable almost everywhere due to the independent increment property.
2. The covariance is scaled linearly with Δt_k instead of the usual and expected $(\Delta t_k)^2$.
3. White noise can be seen as weak derivative of Brownian motion, i.e. in the limit $\Delta t_k \to 0$, we can obtain $\boldsymbol{\beta}(t)$ by integrating over $\mathbf{w}(t)$. This does not mean derivative of Brownian motion exists in the traditional sense, it just means we can construct Brownian motion by summing over white noise process.

First observation is the reason why even the more general version of Riemann sum (Stieltjes) also does not converge. The second observation is the reason for the extra term in the Itô's formula, as shown below

Theorem 2. *(Itô's Formula 1) Consider a scalar Brownian process $\beta(t)$, we have*

$$\int \beta(t)d\beta(t) = \frac{1}{2}\beta^2(t) - \frac{1}{2}t \tag{2.2.7}$$

or the more popular and equivalent form

$$\frac{1}{2}d\beta^2(t) = \beta(t)d\beta(t) + \frac{1}{2}dt \tag{2.2.8}$$

The first term from (2.2.8) is normal chain rule, the second term is unique to Itô's integral, this comes from the fact that the covariance of $\beta(t)$ is scaled by Δt instead of $(\Delta t)^2$, making the 2nd-order term in the Taylor expansion non-negligible, i.e. $dt = (d\beta(t))^2$. Mathematically, this result comes from the following sum, which can be considered the definition for $\int \beta(t)d\beta(t)$

Definition 3. *Consider the sequence* $0 < t_1 < \cdots < t_k < \cdots < t$, *where* $t_k - t_{k-1} = \frac{t}{n}$, *then the integral* $\int_0^t \beta(s)d\beta(s)$ *is defined as*

$$\int_0^t \beta(s)d\beta(s) = \lim_{n\to\infty} \sum_k \beta(t_k)(\beta(t_{k+1}) - \beta(t_k)) \tag{2.2.9}$$

The equality is convergence in probability, and the limit converges to the result in Theorem 2. Proof sketch for this is provided in Appendix A.5.2, detailed treatment is referred to [4]. The Itô's Formula can also be extended to function of Brownian motion

Theorem 3. *(Itô's Formula 2) Let* $\mathbf{x}(t)$ *be a function of* $\boldsymbol{\beta}(t)$, *i.e.* $\mathbf{x}(t)$ *is a solution to some SDEs driven by* $\mathbf{w}(t) = \frac{d\boldsymbol{\beta}(t)}{dt}$, *and* $\phi(\mathbf{x}(t), t)$ *is some arbitrary scalar function, then*

$$d\phi = \frac{\partial \phi}{\partial t}dt + \sum_i \frac{\partial \phi}{\partial x_i}dx_i + \frac{1}{2}\sum_i \Big(\frac{\partial^2 \phi}{\partial x_i \partial x_j}dx_i dx_j\Big) \tag{2.2.10}$$

If we consider the scalar case $\mathbf{x}(t) = x(t)$, this simplifies to

$$d\phi = \frac{\partial \phi}{\partial t}dt + \frac{\partial \phi}{\partial x}dx + \frac{1}{2}\frac{\partial^2 \phi}{\partial x^2}(dx)^2 \tag{2.2.11}$$

Again, the first two terms on the right-hand side of (2.2.11) is normal chain rule, the third term, unique to Itô's calculus, is often referred to as quadratic variation, detailed proof can be found in Chapter 4 of [4]. The extension to vector function $\boldsymbol{\phi}(\cdot)$ is trivial since the rule is applied element wise. As an example, we can in fact recover Theorem 2 from Theorem 3, as demonstrated in following example

Example 3. *Let* $x(t) = \beta(t)$ *and* $\phi(x(t), t) = \frac{1}{2}x^2(t)$, *applying Theorem 3:*

$$\begin{aligned} d\phi &= \frac{\partial \phi}{\partial t}dt + \frac{\partial \phi}{\partial x}dx + \frac{1}{2}\frac{\partial^2 \phi}{\partial x^2}(dx)^2 && (2.2.12)\\ &= 0 + xdx + \frac{1}{2}dxdx && (2.2.13)\\ &= \beta(t)d\beta(t) + \frac{1}{2}(d\beta(t))^2 && (2.2.14)\\ &= \beta(t)d\beta(t) + \frac{1}{2}dt && (2.2.15) \end{aligned}$$

Again, we note that $dt = (d\beta(t))^2$, this fact is central to all the theory of Stochastic calculus. Being equipped with Itô's Formula, the integral $\int \mathbf{L}(\mathbf{x}(t), t)\mathbf{w}(t)dt$ is now well-defined in the sense of Itô's integral, thus general solutions for linear SDEs enjoy the form:

$$\mathbf{x}(t) = \boldsymbol{\Psi}(t, t_0)\mathbf{x}(t_0) + \int_{t_0}^t \boldsymbol{\Psi}(t, \tau)\mathbf{L}(\tau)d\boldsymbol{\beta}(\tau) \tag{2.2.16}$$

where $\boldsymbol{\beta}(t)$ is Brownian motion, and $\boldsymbol{\Psi}(t, t_0)$ is state transition matrix as defined earlier. As was the case with ODEs, solutions to SDEs, even in the linear cases, are generally not obtainable in closed-form. We provide some example solutions to simple SDEs in Appendix A.5.3. In practice, solving SDEs usually involves numerical iterations (2.2.11) using discretization methods such as Euler-Maruyama or Heun's methods, in-depth analysis can be found in [7].

2.2.2 Existence and uniqueness

Having introduced Brownian motion and Itô's integrals, it is evident that Theorem 1 cannot be applied to SDEs due to the discontinuity almost everywhere of the sample paths. Fortunately, an extension to Theorem 1 that relaxes the continuous paths requirement does exist, this theorem guarantees the existence of SDEs solutions. For uniqueness, besides the requirement of being Lipschitz continuous in x, the function $\mathbf{f}$ and $\mathbf{L}$ have to grow at most linearly with x, then there exists a strong unique solution to SDEs. Details can be found in [4].

2.3 Probability densities of the solutions and the Fokker-Planck equation

Since the solution to an SDE is itself a random process, another way to characterize the solution is by looking at the induced probability density $p(x(t)) = p(x,t)$ at time t. As the sample path $x(t)$ evolves with time according to some dynamics defined by an SDE, the distribution of $x(t)$ at time t also evolves as time progress. This evolution of a probability density can be described by the following partial differential equation (PDE), also known as the Fokker-Planck equation

Theorem 4. *(Fokker-Planck equation) Let $\mathbf{x}(t)$ be the solution to the SDE defined in Section 2.2 and $p(\mathbf{x},t)$ be the probability density of $\mathbf{x}(t)$, then $p(\mathbf{x},t)$ solves the following PDE*

$$\frac{\partial p(\mathbf{x},t)}{\partial t} = -\sum_i \frac{\partial}{\partial x_i}\big(f_i(\mathbf{x},t)p(\mathbf{x},t)\big) + \frac{1}{2}\sum_{i,j}\frac{\partial^2}{\partial x_i \partial x_j}\Big(\big[\mathbf{L}(\mathbf{x},t)\mathbf{L}^T(\mathbf{x},t)\big]_{ij}p(\mathbf{x},t)\Big) \tag{2.3.1}$$

Proof: [8] Section 5.2.

As an example, we can recover the famous heat diffusion equation from the Fokker-Planck equation

Example 4. *(Diffusion equation) Consider the 1-dimensional Brownian motion*

$$dx = 2Dd\beta(t) \tag{2.3.2}$$

Then by applying Fokker-Planck equation, we have:

$$\frac{\partial p(x,t)}{\partial t} = \frac{1}{2}\frac{\partial^2}{\partial x^2}2Dp(x,t) \tag{2.3.3}$$

$$\Rightarrow \frac{\partial p(x,t)}{\partial t} = D\frac{\partial^2 p(x,t)}{\partial x^2} \tag{2.3.4}$$

which is the well-known diffusion equation in Physics.

While the proof for Fokker-Planck equation is rather complicated, it can also be derived "easily" from Maxwell's equations, in a hand-wavy fashion. If we consider $p(x,t) \propto \rho(x,t)$, i.e. the evolution of charge density in space and time, then by following Maxwell's equation relating the curl of magnetic field to the change of electric field:

Example 5. *(Continuity equation)*

$$\nabla \times \mathbf{H} = \mathbf{J} + \frac{\partial \mathbf{D}}{\partial t} \tag{2.3.5}$$

$$\nabla \cdot \nabla \times \mathbf{H} = \nabla \cdot \mathbf{J} + \frac{\partial \nabla \cdot \mathbf{D}}{\partial t} \tag{2.3.6}$$

$$\mathbf{0} = \nabla \cdot \mathbf{J} + \frac{\partial \rho}{\partial t} \tag{2.3.7}$$

$$\frac{\partial \rho}{\partial t} = -\nabla \cdot \mathbf{J} \tag{2.3.8}$$

Equation (2.3.8) *is known as the Continuity equation in Electromagnetic and as Transport equation in Optimal transport theory. This equation holds for any conserved quantities, i.e. total electrical charges. To recover the heat diffusion equation, we note that $\mathbf{J}$ is a velocity field $\mathbf{v}$ (describing how the charges move), thus it is the first-order derivatives of spatial coordinates $\mathbf{u}$, hence*

$$\nabla \cdot \mathbf{J} = \nabla \cdot \mathbf{v} \tag{2.3.9}$$

$$= \nabla \cdot \nabla \mathbf{u} \tag{2.3.10}$$

With proper choice of $\mathbf{u}$ we get the diffusion equation. To relate this with the general Fokker-Planck equation in Theorem 4, we can imagine the effect of $\mathbf{f}(\mathbf{x},t)dt$ as introducing new charges into the system. Thus, the total charges is no longer conserved, hence equation (2.3.8) *needs to be corrected by an additional term:*

$$\frac{\partial \rho}{\partial t} = \sigma - \nabla \cdot \mathbf{J} \tag{2.3.11}$$

Equation (2.3.11) *is the general transport equation from Optimal transport theory.*

The Fokker-Planck and Continuity equations play critical roles in generative modelling, in which we want to study the evolution of the data distribution with respect to time. We will return to these equations in the later sections of this thesis. For now, we need one last ingredient to make generative modelling using SDEs work, which is the Anderson's theorem regarding the reverse processes.

2.4 Time-reversals of SDEs

For this section, let us consider the following type of 1-dimensional SDEs:

$$dx(t) = f(x(t), t)dt + l(t)d\beta(t) \tag{2.4.1}$$

where $l(t)$ does not have dependency on spatial variable x. We will simplify the notations a little bit by denoting

$$d\overrightarrow{X}_t = f(\overrightarrow{X}_t)dt + l(t)d\beta_t \tag{2.4.2}$$

where we use $\overrightarrow{X}_t$ to denote the solution of the SDE running forward in time. Let $\overrightarrow{\pi}_t = p_t(\overrightarrow{X}_t)$ denotes the probability density of $\overrightarrow{X}_t$. In this section, we want to construct a reverse SDE such that its solution $\overleftarrow{X}_t$ has the probability density $\overleftarrow{\pi}_t = p_t(\overleftarrow{X}_t)$ with the following property:

$$\overrightarrow{\pi}_t = \overleftarrow{\pi}_{T-t}, \forall t \in [0, T] \tag{2.4.3}$$

Start by applying the Fokker-Planck equation to $\overrightarrow{\pi}_t$

$$\frac{\partial \overrightarrow{\pi}_t}{\partial t} = -\frac{\partial}{\partial x}\big(f(x,t)\overrightarrow{\pi}_t\big) + \frac{1}{2}\frac{\partial^2}{\partial x^2}l^2(t)\overrightarrow{\pi}_t \tag{2.4.4}$$

$$= -\frac{\partial}{\partial x}\big(f(x,t)\overrightarrow{\pi}_t\big) + \frac{1}{2}l^2(t)\frac{\partial^2}{\partial x^2}\overrightarrow{\pi}_t \tag{2.4.5}$$

$$= -\frac{\partial}{\partial x}\big(f(x,t)\overrightarrow{\pi}_t\big) + \frac{1}{2}l^2(t)\frac{\partial}{\partial x}\Big(\overrightarrow{\pi}_t\big(\frac{\partial}{\partial x}\log\overrightarrow{\pi}_t\big)\Big) \tag{2.4.6}$$

$$= -\frac{\partial}{\partial x}\big(f(x,t)\overrightarrow{\pi}_t\big) + \frac{1}{2}l^2(t)\Big(\big(\frac{\partial}{\partial x}\log\overrightarrow{\pi}_t\big)\frac{\partial}{\partial x}\overrightarrow{\pi}_t + \big(\frac{\partial^2}{\partial x^2}\log\overrightarrow{\pi}_t\big)\overrightarrow{\pi}_t\Big) \tag{2.4.7}$$

$$= -\overrightarrow{\pi}_t\frac{\partial}{\partial x}f(x,t) - f(x,t)\frac{\partial}{\partial x}\overrightarrow{\pi}_t + \frac{1}{2}l^2(t)\big(\frac{\partial}{\partial x}\log\overrightarrow{\pi}_t\big)\frac{\partial}{\partial x}\overrightarrow{\pi}_t + \frac{1}{2}l^2(t)\big(\frac{\partial^2}{\partial x^2}\log\overrightarrow{\pi}_t\big)\overrightarrow{\pi}_t \tag{2.4.8}$$

$$= -\overrightarrow{\pi}_t\frac{\partial}{\partial x}f(x,t) + \frac{1}{2}l^2(t)\big(\frac{\partial^2}{\partial x^2}\log\overrightarrow{\pi}_t\big)\overrightarrow{\pi}_t + \Big(-f(x,t) + \frac{1}{2}l^2(t)\big(\frac{\partial}{\partial x}\log\overrightarrow{\pi}_t\big)\Big)\frac{\partial}{\partial x}\overrightarrow{\pi}_t \tag{2.4.9}$$

$$= \overrightarrow{\pi}_t\frac{\partial}{\partial x}\Big(-f(x,t) + \frac{1}{2}l^2(t)\frac{\partial}{\partial x}\log\overrightarrow{\pi}_t\Big) + \Big(-f(x,t) + \frac{1}{2}l^2(t)\frac{\partial}{\partial x}\log\overrightarrow{\pi}_t\Big)\frac{\partial}{\partial x}\overrightarrow{\pi}_t \tag{2.4.10}$$

$$= -\frac{\partial}{\partial x}\left(\Big(f(x,t) - \frac{1}{2}l^2(t)\frac{\partial}{\partial x}\log\overrightarrow{\pi}_t\Big)\overrightarrow{\pi}_t\right) \tag{2.4.11}$$

Since $\overrightarrow{\pi}_t = \overleftarrow{\pi}_{T-t}$, we further have:

$$\frac{\partial \overleftarrow{\pi}_{T-t}}{\partial t} = -\frac{\partial}{\partial x}\left(\Big(-f(x,T-t) + \frac{1}{2}l^2(T-t)\frac{\partial}{\partial x}\log\overleftarrow{\pi}_{T-t}\Big)\overleftarrow{\pi}_{T-t}\right) \tag{2.4.12}$$

$$= -\frac{\partial}{\partial x}\left(\Big(-f(x,T-t) + l^2(T-t)\frac{\partial}{\partial x}\log\overleftarrow{\pi}_{T-t} - \frac{1}{2}l^2(T-t)\frac{\partial}{\partial x}\log\overleftarrow{\pi}_{T-t}\Big)\overleftarrow{\pi}_{T-t}\right) \tag{2.4.13}$$

$$= -\frac{\partial}{\partial x}\left(\Big(-f(x,T-t) + l^2(T-t)\frac{\partial}{\partial x}\log\overleftarrow{\pi}_{T-t}\Big)\overleftarrow{\pi}_{T-t}\right) + \frac{\partial}{\partial x}\Big(\frac{1}{2}l^2(T-t)\big(\frac{\partial}{\partial x}\log\overleftarrow{\pi}_{T-t}\big)\overleftarrow{\pi}_{T-t}\Big) \tag{2.4.14}$$

$$= -\frac{\partial}{\partial x}\left(\Big(-f(x,T-t) + l^2(T-t)\frac{\partial}{\partial x}\log\overleftarrow{\pi}_{T-t}\Big)\overleftarrow{\pi}_{T-t}\right) + \Big(\frac{1}{2}l^2(T-t)\frac{\partial}{\partial x}\overleftarrow{\pi}_{T-t}\big(\frac{\partial}{\partial x}\log\overleftarrow{\pi}_{T-t}\big)\Big) \tag{2.4.15}$$

$$= -\frac{\partial}{\partial x}\left(\Big(-f(x,T-t) + l^2(T-t)\frac{\partial}{\partial x}\log\overleftarrow{\pi}_{T-t}\Big)\overleftarrow{\pi}_{T-t}\right) + \Big(\frac{1}{2}\frac{\partial^2}{\partial x^2}l^2(T-t)\overleftarrow{\pi}_{T-t}\Big) \tag{2.4.16}$$

Observe that equation (2.4.16) matches the Fokker-Planck equation from Theorem (4), and since Fokker-Planck defines a unique SDE, this means the reverse time probability density $\overleftarrow{\pi}_{T-t}$ is the result of the following SDE:

$$dX_{T-t} = \Big(- f(x, T-t) + l^2(T-t)\frac{\partial}{\partial x} \log \overleftarrow{\pi}_{T-t} \Big) dt + l(T-t) d\beta_{T-t} \tag{2.4.17}$$

$$= \Big(- f(x, T-t) + l^2(T-t) \nabla \log \overleftarrow{\pi}_{T-t} \Big) dt + l(T-t) d\beta_{T-t} \tag{2.4.18}$$

By following the same idea, we can extend this result to multidimensional case:

$$d\mathbf{x}_{T-t} = \Big(- \mathbf{f}(\mathbf{x}, T-t) + \mathbf{L}(T-t)\mathbf{L}^T(T-t) \nabla \log \overleftarrow{\boldsymbol{\pi}}_{T-t} \Big) dt + \mathbf{L}(T-t) d\boldsymbol{\beta}_{T-t} \tag{2.4.19}$$

This result says that if we know the score function $\nabla \log \overleftarrow{\boldsymbol{\pi}}_{T-t}$ and if $\mathbf{L}$ does not have spatial dependency, then we can run the SDE defined in (2.4.19) and the marginals distribution at $T-t$ will match that of the forward SDE. This has the following important implication: let $\overrightarrow{\boldsymbol{\pi}}_T$ be the terminal distribution of the forward SDE, if we initialize (2.4.19) with $\mathbf{x}_0 \sim \overrightarrow{\boldsymbol{\pi}}_T$ and run the reverse SDE, then the end result will follow the initial distribution of the forward SDE $\mathbf{x}_T \sim \overrightarrow{\boldsymbol{\pi}}_0$. Thus, if we set $\overrightarrow{\boldsymbol{\pi}}_0$ to be the data distribution, then run the forward SDE to obtain Gaussian noise, then if somehow we obtain $\nabla \log \overleftarrow{\boldsymbol{\pi}}_{T-t} = \nabla \log \overrightarrow{\boldsymbol{\pi}}_t$, we can start from Gaussian noise and run the reverse SDE to generate samples that come from $\overrightarrow{\boldsymbol{\pi}}_0$. This result is sometimes referred to as Anderson's theorem, and it is the foundation for diffusion-based generative modelling, which will be the main discussion of the next sections.

Before moving on, we will make an important note, the result from (2.4.19) guarantees that the marginals of the reverse and forward SDE matches, i.e. $\overleftarrow{\boldsymbol{\pi}}_{T-t} = \overrightarrow{\boldsymbol{\pi}}_t$. However, this does not make any claims about the relationship between the joint densities $\overleftarrow{\boldsymbol{\pi}}(\mathbf{x}_0, \dots, \mathbf{x}_T)$ and $\overrightarrow{\boldsymbol{\pi}}(\mathbf{x}_T, \dots, \mathbf{x}_0)$. In fact, in general no process exists that matches the joint distributions.

3 Overview of generative modelling

In this section, we will give a brief review of generative modelling and some important results from variational inference theory. The ultimate goal of generative modelling is to sample from some data distribution $p(\mathbf{x})$, based on the information we get from the empirical samples $\mathbf{x} \sim p(\mathbf{x})$. In other words, there are two steps involved in generative modelling:

1. Approximate the true data distribution based on some parameterization $p_{\boldsymbol{\theta}}(\mathbf{x}) \approx p(\mathbf{x})$
2. Sample from the learned/approximated $p_{\boldsymbol{\theta}}(\mathbf{x})$

The problem is it is usually intractable to learn $p_{\boldsymbol{\theta}}(\mathbf{x})$, where the difficulty often lies in the normalizing constant. As an example, by borrowing the idea of canonical ensemble from Physics, $p_{\boldsymbol{\theta}}(\mathbf{x})$ is often assumed to have the form

$$p_{\boldsymbol{\theta}}(\mathbf{x}) = \frac{1}{Z} e^{-f_{\boldsymbol{\theta}}(\mathbf{x})} \tag{3.0.1}$$

$$Z = \int_{\mathcal{X}} e^{-f_{\boldsymbol{\theta}}(\mathbf{x})} d\mathbf{x} \tag{3.0.2}$$

where $\mathcal{X}$ is the dataset and Z is the normalizing constant, also known as the partition function. While $f_{\boldsymbol{\theta}}(\mathbf{x})$ can be easily approximated by a neural network, evaluating Z is often intractable since we have to integrate over all data points, this motivates the search for methods that do not rely on Z to perform sampling. Here, we will introduce variational inference to combat this problem.

Following Bayesian setting, we assume $p_{\boldsymbol{\theta}}(\mathbf{x})$ is marginalized from a joint distribution

$$p_{\boldsymbol{\theta}}(\mathbf{x}) = \int p_{\boldsymbol{\theta}}(\mathbf{x}, \mathbf{z}) d\mathbf{z} \tag{3.0.3}$$

$$= \int p_{\boldsymbol{\theta}}(\mathbf{x}|\mathbf{z}) p_{\boldsymbol{\theta}}(\mathbf{z}) d\mathbf{z} \tag{3.0.4}$$

From the prior $p_{\boldsymbol{\theta}}(\mathbf{z})$ and the likelihood $p_{\boldsymbol{\theta}}(\mathbf{x}|\mathbf{z})$, Bayesian inference is concerned with computing the posterior $p_{\boldsymbol{\theta}}(\mathbf{z}|\mathbf{x})$. In complex settings, such as image or video processing, $p_{\boldsymbol{\theta}}(\mathbf{z}|\mathbf{x})$ is often intractable and requires approximation. To this end, another parameterization is usually introduced $q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x}) \approx p_{\boldsymbol{\theta}}(\mathbf{z}|\mathbf{x})$. From this formulation,

$E_{\boldsymbol{\phi}} \sim q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x})$, and $D_{\boldsymbol{\phi}} \sim p_{\boldsymbol{\theta}}(\mathbf{x}|\mathbf{z})$ are often referred to as the encoder and decoder networks, respectively. The approximation $q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x}) \approx p_{\boldsymbol{\theta}}(\mathbf{z}|\mathbf{x})$ is also known as Amortized inference in the literature [9]. Next, we derive a differentiable loss that can be used to learn $E_{\boldsymbol{\phi}}$ and $D_{\boldsymbol{\phi}}$ through backpropagation. Since we want $q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x}) \approx p_{\boldsymbol{\theta}}(\mathbf{z}|\mathbf{x})$, it is natural to use the KL divergence as the minimization objective

$$D_{\mathrm{KL}}\big(q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x})||p_{\boldsymbol{\theta}}(\mathbf{z}|\mathbf{x})\big) = \mathbb{E}_{\mathbf{z}\sim q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x})}\Big[\log \frac{q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x})}{p_{\boldsymbol{\theta}}(\mathbf{z}|\mathbf{x})}\Big] \tag{3.0.5}$$

$$= \mathbb{E}_{\mathbf{z}\sim q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x})}\Big[\log \frac{q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x})p_{\boldsymbol{\theta}}(\mathbf{x})}{p_{\boldsymbol{\theta}}(\mathbf{z},\mathbf{x})}\Big] \tag{3.0.6}$$

$$= \log p_{\boldsymbol{\theta}}(\mathbf{x}) + \mathbb{E}_{\mathbf{z}\sim q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x})}\Big[\log \frac{q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x})}{p_{\boldsymbol{\theta}}(\mathbf{z},\mathbf{x})}\Big] \tag{3.0.7}$$

$$\Rightarrow \mathcal{L} := \mathbb{E}_{\mathbf{z}\sim q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x})}\Big[\log \frac{p_{\boldsymbol{\theta}}(\mathbf{z},\mathbf{x})}{q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x})}\Big] = \log p_{\boldsymbol{\theta}}(\mathbf{x}) - D_{\mathrm{KL}}\big(q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x})||p_{\boldsymbol{\theta}}(\mathbf{z}|\mathbf{x})\big) \tag{3.0.8}$$

Thanks to the non-negativity of KL divergence, $\mathcal{L}$ in (3.0.8) lower bounds the data loglikelihood, this is known as the Evidence Lower Bound (ELBO) or Variational Lower Bound (VLB). Maximizing the ELBO is, in a manner, equivalent to performing maximum log likelihood. In its current form, (3.0.8) is not very useful for optimization, fortunately there is another formulation

$$\mathcal{L} := \mathbb{E}_{\mathbf{z}\sim q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x})}\Big[\log \frac{p_{\boldsymbol{\theta}}(\mathbf{z},\mathbf{x})}{q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x})}\Big] = \mathbb{E}_{\mathbf{z}\sim q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x})}\Big[\log \frac{p_{\boldsymbol{\theta}}(\mathbf{x}|\mathbf{z})p_{\boldsymbol{\theta}}(\mathbf{z})}{q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x})}\Big] \tag{3.0.9}$$

$$= \mathbb{E}_{\mathbf{z}\sim q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x})}[\log p_{\boldsymbol{\theta}}(\mathbf{x}|\mathbf{z})] - D_{\mathrm{KL}}\big(q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x})||p_{\boldsymbol{\theta}}(\mathbf{z})\big) \tag{3.0.10}$$

In practice, one often parameterizes $p_{\boldsymbol{\theta}}(\mathbf{x}|\mathbf{z}) \sim \mathcal{N}(\mathbf{D}_{\boldsymbol{\theta}}(\mathbf{z}), \mathbf{I})$ then $\log p_{\boldsymbol{\theta}}(\mathbf{x}|\mathbf{z}) = -\frac{1}{2}||\mathbf{x} - \mathbf{D}_{\boldsymbol{\theta}}(\mathbf{z})||_2^2 + \text{const}$. Furthermore, $p_{\boldsymbol{\theta}}(\mathbf{z})$ and $q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x})$ are also usually chosen to be Gaussian $p_{\boldsymbol{\theta}}(\mathbf{z}) \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$, $q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x}) \sim \mathcal{N}(\mathbf{E}_{\boldsymbol{\phi}}(\mathbf{x}), \sigma_{\boldsymbol{\phi}}^2(\mathbf{x})\mathbf{I})$. Then (3.0.10) becomes tractable and trainable by backpropagation:

$$\mathcal{L} = -\frac{1}{2}||\mathbf{x} - \mathbf{D}_{\boldsymbol{\theta}}(\mathbf{z})||_2^2 - \frac{1}{2}\big(d\sigma_{\boldsymbol{\phi}}^2(\mathbf{x}) + ||\mathbf{E}_{\boldsymbol{\phi}}(\mathbf{x})|| - 2d\log\sigma_{\boldsymbol{\phi}}(\mathbf{x})\big) + \text{const} \tag{3.0.11}$$

where d is dimension of $\mathbf{z}$. Maximizing (3.0.11) over $\boldsymbol{\phi}$ and $\boldsymbol{\theta}$ would yield encoder $\mathbf{E}_{\boldsymbol{\phi}}(\mathbf{x})$ and decoder $\mathbf{D}_{\boldsymbol{\theta}}(\mathbf{z})$ such that $\mathbf{D}_{\boldsymbol{\theta}}(\mathbf{E}_{\boldsymbol{\phi}}(\mathbf{x})) \sim p(\mathbf{x})$. Since $\mathbf{z}$ is assumed to follow normal distribution, we can sample $\mathbf{z} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ and run the decoder $\mathbf{D}_{\boldsymbol{\theta}}(\mathbf{z})$ to generate new sample that looks like it comes from the data distribution. This is known as Variational Autoencoder, the training actually requires a small modification to reparameterize the noise, details can be found in [10]. This method is closely related to the idea of forward SDE (encoder) and reverse SDE (encoder) introduced in Section 2.2. Building on top of this idea, in the following sections we will introduce how to apply SDE to the context of generative modelling.

Before moving on, we note that the choice of $D_{\mathrm{KL}}\big(q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x})||p_{\boldsymbol{\theta}}(\mathbf{z}|\mathbf{x})\big)$ is mode-covering, meaning we want q to place its mass at the location where p does. This is crucial for generative modelling since we want to generate samples that actually come from p. Choosing the reverse KL as objective, i.e. $D_{\mathrm{KL}}\big(p_{\boldsymbol{\theta}}(\mathbf{z}|\mathbf{x})||q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x})\big)$ will not have this effect.

4 Generative modelling using SDEs

4.1 Langevin dynamics and Score matching

Here, we introduce the Langevin dynamics, which will serve as a basic for generative modelling using SDEs. The 1-dimensional Langevin SDE is given as:

$$dx = -\nabla U(x)dt + \sqrt{2}d\beta_t \tag{4.1.1}$$

where $U(x)$ is some energy function. Now, we wish to know the stationary distribution $p_t(x)$ induced by (4.1.1) as $t \to \infty$. To do so, we can apply the Fokker-Planck equation and set it to zero:

$$\frac{\partial p_\infty}{\partial t} = \frac{\partial}{\partial x}\Big(\nabla U(x)p_\infty(x)\Big) + \frac{1}{2}\frac{\partial^2}{\partial x^2}2p_\infty(x) \tag{4.1.2}$$

$$= \frac{\partial}{\partial x}\Big(\underbrace{\nabla U(x)p_\infty(x) + \frac{\partial}{\partial x}p_\infty(x)}_{J}\Big) = 0 \tag{4.1.3}$$

Equation (4.1.3) can only be true if J is constant for all x. Thus

$$\nabla U(x)p_\infty(x) + \frac{\partial}{\partial x}p_\infty(x) = 0 \tag{4.1.4}$$

$$\Rightarrow \frac{\partial}{\partial x}p_\infty(x) = -\nabla U(x)p_\infty(x) \tag{4.1.5}$$

$$\Rightarrow p_\infty(x) = e^{-U(x)} \tag{4.1.6}$$

If we set $U(x)$ to be $-\log p(x)$ then

$$p_\infty(x) = e^{-U(x)} = e^{\log p(x)} = p(x) \tag{4.1.7}$$

Equation (4.1.7) is a well-known result in stochastic sampling, and also in stochastic optimization [11]. Essentially, it means we can obtain samples from target distribution $p(x)$ if we run the following SDE for a long time:

$$dx = \nabla \log p(x)dt + \sqrt{2}d\beta_t \tag{4.1.8}$$

To that end, we can discretize (4.1.8) using Euler-Maruyama method with step size Δt to obtain

$$x_k = x_{k-1} + \nabla \log p(x_{k-1})\Delta t + \sqrt{2\Delta t}n, \quad n \sim \mathcal{N}(0,1) \tag{4.1.9}$$

where $\sqrt{\Delta t}$ appears due to the fact that variance of Brownian motion scales with Δt. This idea is illustrated in the following toy example

Example 6. *(Langevin dynamics) Let x be drawn from a simple Gaussian mixture, we will abuse notations here*

$$p(x) = \alpha\mathcal{N}(\mu_1, \sigma_1^2) + (1-\alpha)\mathcal{N}(\mu_2, \sigma_2^2) \tag{4.1.10}$$

We can then sample from $p(x)$ using equation (4.1.9) with

$$\nabla \log p(x_{k\Delta t}) = \frac{1}{p(x_{k\Delta t})}\nabla p(x_{k\Delta t}) \tag{4.1.11}$$

$$= -\frac{\frac{x-\mu_1}{\sigma_1^2}\alpha\mathcal{N}(\mu_1, \sigma_1^2) + \frac{x-\mu_2}{\sigma_2^2}(1-\alpha)\mathcal{N}(\mu_2, \sigma_2^2)}{\alpha\mathcal{N}(\mu_1, \sigma_1^2) + (1-\alpha)\mathcal{N}(\mu_2, \sigma_2^2)} \tag{4.1.12}$$

The result is shown in Figure 1 and 2, the experiment was run with $\mu_1 = -5$, $\mu_2 = 5$, $\sigma_1 = \sigma_2 = 2$, $\alpha = 0.3$. Red lines show randomly selected sample paths.

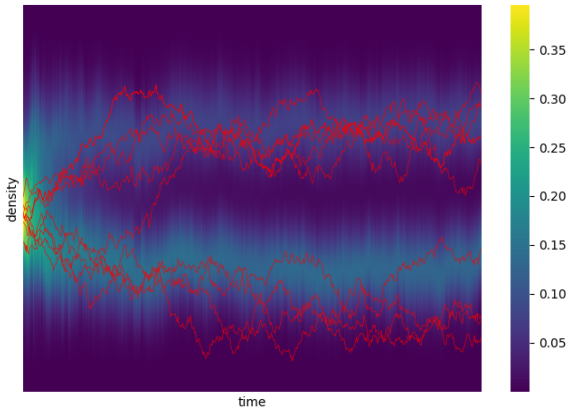

Figure 1: Evolution of the density from Example 6 . Starting from $\mathcal{N}(0,1)$, equation (4.1.9) is run to obtain samples from the Gaussian mixture.

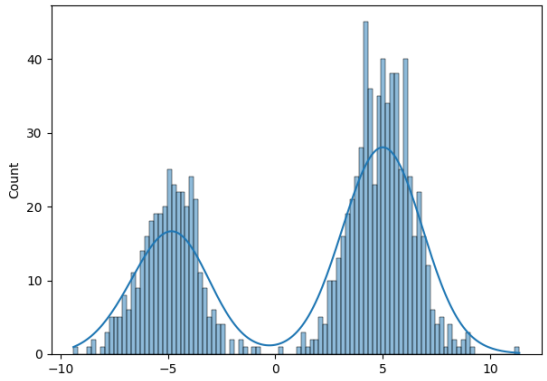

Figure 2: Samples generated by Langevin sampling, which match the true Gaussian mixture with $\mu_1 = -5$, $\mu_2 = 5$, $\sigma_1 = \sigma_2 = 2$, $\alpha = 0.3$

In practice, $\nabla \log p(x)$ is usually not available and will require some approximation. This leads to score matching methods. We note that the score mentioned here, $\nabla_x \log p(x)$ is different from the usual score in statistical learning,

where the gradient is taken with respect to the model parameters, here it is taken with respect to the data. A major motivation for score-based methods is that it can bypass the computation of the normalizing constant Z mentioned in (3.0.1), as demonstrated below:

$$\nabla_{\mathbf{x}} \log p(\mathbf{x}) = \nabla_{\mathbf{x}} \log \frac{1}{Z_t} e^{-f_{\boldsymbol{\theta}}(\mathbf{x})} \tag{4.1.13}$$

$$= \nabla_{\mathbf{x}} e^{-f_{\boldsymbol{\theta}}(\mathbf{x})} - \underbrace{\nabla_{\mathbf{x}} \log Z}_{=0} = -\nabla_{\mathbf{x}} f_{\boldsymbol{\theta}}(\mathbf{x}) \tag{4.1.14}$$

If we consider $\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x})$ to be some parameterized functions, i.e. output of a neural network, then $\nabla \log p(\mathbf{x})$ can be estimated by minimizing the following score-matching objective, sometimes also referred to as the Fisher divergence

$$\boldsymbol{\theta}^* = \arg\min_{\boldsymbol{\theta}} \ \mathbb{E}_{\mathbf{x}}\Big[\frac{1}{2}||\nabla \log p(\mathbf{x}) - \mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x})||_2^2\Big] \tag{4.1.15}$$

The idea is we can use our training dataset to learn $\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x})$, then this can be used in place of $\nabla \log p(\mathbf{x})$ during the generative process. This requires the estimation of $\nabla \log p(\mathbf{x})$ across the training dataset, which is often expensive and does not scale well with the size of training data. A major breakthrough was discovered in [12], where the authors found that

$$\mathbb{E}_{\mathbf{x}}\Big[\frac{1}{2}||\nabla \log p(\mathbf{x}) - \mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x})||_2^2\Big] \propto \mathbb{E}_{\mathbf{x}}\Big[\frac{1}{2}||\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x})||_2^2 + \text{trace}\big(\nabla \mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x})\big)\Big] \tag{4.1.16}$$

$$\approx \frac{1}{N}\sum_{i=1}^{N}\Big[\frac{1}{2}||\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}_i)||_2^2 + \text{trace}\big(\nabla \mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}_i)\big)\Big] \tag{4.1.17}$$

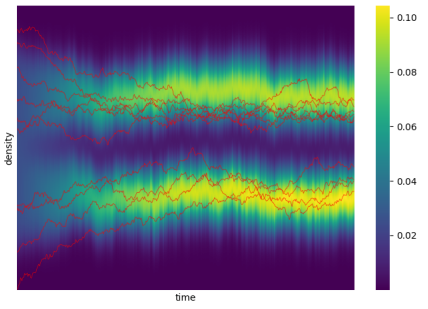

Figure 3: Example 6 started from uniform distribution $\mathcal{U}[-20, 20]$

This result is remarkable since we no longer need to empirically estimate $\nabla \log p(\mathbf{x})$. While $\nabla \mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}_i)$ is still expensive to compute, it is readily available using any auto-differentiation frameworks such as PyTorch [13]. We can further use random projection to efficiently estimate $\text{trace}\big(\nabla_{\mathbf{x}} \mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x})\big)$

$$\mathbb{E}_{\mathbf{x}}\Big[\text{trace}\big(\nabla \mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x})\big)\Big] \approx \mathbb{E}_{\mathbf{v}}\mathbb{E}_{\mathbf{x}}\Big[\mathbf{v}^T \nabla \mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x})\mathbf{v}\Big] \tag{4.1.18}$$

where $\mathbf{v}$ is random vector with unit length. This, in conjunction with Langevin dynamics form the thesis of the seminal works in [1], which obtained state-of-the-art result in images generation. Note that we started from $\mathcal{N}(0, 1)$ in Example 6, this is not a requirement. In fact, we can start from an arbitrary distribution and this method still works. For example, Figure 3 shows the same process started from a uniform distribution $\mathcal{U}[-20, 20]$. We can see that the process reaches the target Gaussian mixture regardless of the starting distribution.

To conclude this section, we would like to point out some weaknesses of this framework. Since we only observe samples from the true data distribution, which can be seen as some point mass over the support of the density, this means estimating $\nabla \log p(\mathbf{x})$ can be difficult or even intractable in low-density regions, where no samples are observed. This is similar to being stuck in local minima with gradient descent, where the gradient is zero, here $\nabla \log p(\mathbf{x})$ will be close to zero in low-density regions, thus sampling will be extremely slow if our sample points start there. The authors of [1] tried to combat this problem by artificially adding noise to the data samples, making its distribution supported everywhere. It turns out this idea is closely related to the forward and reverse SDEs discussed in Section 2.2, usually referred to as Diffusion models.

4.2 Diffusion models

From the observation in Section 4.1, estimating $\nabla \log p(\mathbf{x})$ is easier if we add a little noise to our data samples. This leads to another idea, we can slowly add noise to our samples at each time step, then we can learn $\nabla \log p_t(\mathbf{x}_t)$ of that time step using the score-matching methods, and sampling can be conducted using the reverse SDEs. Here, we recall the forward linear SDE and the reverse SDE defined in Section 2.2

$$d\mathbf{x}_t = \mathbf{F}(t)\mathbf{x}_t dt + \mathbf{L}(t) d\boldsymbol{\beta}_t \tag{4.2.1}$$

$$d\mathbf{x}_{T-t} = \Big(-\mathbf{F}(t)\mathbf{x}_t + \mathbf{L}(T-t)\mathbf{L}^T(T-t)\nabla \log p_{T-t}(\mathbf{x}_{T-t})\Big)dt + \mathbf{L}(T-t) d\boldsymbol{\beta}_{T-t} \tag{4.2.2}$$

From this, we can separate the methods into two categories: Variance Exploding (VE) SDEs, and Variance Preserving (VP) SDEs.

4.2.1 Variance Exploding methods

To apply the idea of gradually adding noise to data samples, we can define the following SDEs

$$d\mathbf{x}_t = \sqrt{\frac{d\sigma^2(t)}{dt}} d\boldsymbol{\beta}_t \tag{4.2.3}$$

where $\sigma(t)$ is a scalar-valued function, this simply means we are scaling the variance of the Brownian motion by $\frac{d\sigma(t)}{dt}$ at every time step, i.e. gradually adding more noise to $\mathbf{x}$ as time progress. The form of $\sqrt{\frac{d\sigma(t)}{dt}}$ is peculiar at first, but it has the following nice Euler-Maruyama discretization

$$\mathbf{x}_k = \mathbf{x}_{k-1} + \sqrt{\frac{\sigma^2(k) - \sigma^2(k-1)}{\Delta t}} \Delta t \mathbf{n}, \quad \mathbf{n} \sim \mathcal{N}(\mathbf{0}, \mathbf{I}) \tag{4.2.4}$$

$$= \mathbf{x}_{k-1} + \sqrt{\sigma^2(k) - \sigma^2(k-1)} \mathbf{n} \tag{4.2.5}$$

Then the corresponding reverse SDE is

$$d\mathbf{x}_{T-t} = \frac{d\sigma^2(T-t)}{dt} \nabla \log p_{T-t}(\mathbf{x}_{T-t}) dt + \sqrt{\frac{d\sigma^2(t)}{dt}} d\boldsymbol{\beta}_{T-t} \tag{4.2.6}$$

with the following discretization, $K = T/\Delta t$

$$\mathbf{x}_{K-k} = \mathbf{x}_{K-k+1} - \frac{\sigma^2(K-k) - \sigma^2(K-k+1)}{\Delta t} \nabla \log p_{K-k}(\mathbf{x}_{K-k}) \Delta t + \sqrt{\frac{\sigma^2(K-k) - \sigma^2(K-k+1)}{\Delta t}} \Delta t \mathbf{n} \tag{4.2.7}$$

$$= \mathbf{x}_{K-k+1} + \Big(\sigma^2(K-k+1) - \sigma^2(K-k)\Big) \nabla \log p_{K-k}(\mathbf{x}_{K-k}) + \sqrt{\sigma^2(k) - \sigma^2(k-1)} \mathbf{n} \tag{4.2.8}$$

Let $\alpha(k) = \sigma^2(k) - \sigma^2(k-1)$, we have the following pair of discretized SDEs

$$\mathbf{x}_k = \mathbf{x}_{k-1} + \sqrt{\alpha(k)} \mathbf{n} \tag{4.2.9}$$

$$\mathbf{x}_{K-k} = \mathbf{x}_{K-k+1} - \alpha(K-k) \nabla \log p_{K-k}(\mathbf{x}_{K-k}) + \sqrt{-\alpha(K-k)} \mathbf{n} \tag{4.2.10}$$

To create a generative model using (4.2.9) and (4.2.10), we take the following steps

1. (*Forward pass*) Generate $\mathbf{x}_t$ from $\mathbf{x}_0$ using (4.2.9). Use score-matching methods in Section 5.4 to estimate $\nabla \log p_t(\mathbf{x}_t)$.
2. (*Reverse/Sampling pass*) Starting from $\mathbf{x}_T \sim p_T(\mathbf{x})$, run (4.2.10) with the estimated $\nabla \log p_t(\mathbf{x}_t)$ from Step 1, then $\mathbf{x}_0$ will follow the data distribution.

This process requires $p_T(\mathbf{x})$ to be some simple distribution that we can easily sample from, i.e. Gaussian. Here we give a short analysis of the 1-dimensional version $p_T(x)$ using Fokker-Planck equation similar to Section 4.1

$$\frac{\partial p_t}{\partial t} = \frac{1}{2} \frac{d\sigma^2(t)}{dt} \frac{\partial^2}{\partial x^2} p_t(x) \tag{4.2.11}$$

Let us denote $k_t = \frac{1}{2} \frac{d\sigma^2(t)}{dt}$, then we have the following partial differential equation (PDE)

$$\frac{\partial p_t}{\partial t} = k_t \frac{\partial^2}{\partial x^2} p_t(x) \tag{4.2.12}$$

$$p_t(-\infty) = p_t(\infty) = 0 \tag{4.2.13}$$

$$p_0(x) = p(x) \tag{4.2.14}$$

Condition (4.2.13) is because a valid density must vanish at infinity. Equation (4.2.12) then is a well-known heat diffusion PDE with zero boundary conditions at infinity, it enjoys the following solution [14]

$$p_t(x) = \int_{-\infty}^{\infty} p(\tau) \frac{1}{\sqrt{2\pi(2K_t)}} \exp\Big(-\frac{1}{2}\frac{(x-\tau)^2}{2K_t}\Big) d\tau \tag{4.2.15}$$

$$= p(x) * \mathcal{N}(0, 2K_t), \quad K_t = \int_0^t k_t(\tau) d\tau \tag{4.2.16}$$

$$= p(x) * \mathcal{N}\Big(0, \big(\sigma^2(t) - \sigma^2(0)\big)\Big) \tag{4.2.17}$$

where $*$ denotes convolution operation. We note that the solution in [14] considers $k_t = k = \text{const}$. For time-varying k_t, the solution is almost the same, i.e. equation (4.2.15). This is due the fact that $\big(\exp(-\lambda f(t))\big)' = -\lambda f'(t)\exp(-\lambda f(t))$. One can also derive this from the discretization in (4.2.9)

$$p_t(x) = p_{k\Delta t}(x) \tag{4.2.18}$$

$$= p(x) * \mathcal{N}\Big(0, \big(\sigma^2(\Delta t) - \sigma^2(0)\big)\Big) * \mathcal{N}\Big(0, \big(\sigma^2(2\Delta t) - \sigma^2(\Delta t)\big)\Big) * \mathcal{N}\Big(0, \big(\sigma^2(3\Delta t) - \sigma^2(2\Delta t)\big)\Big) * \dots \tag{4.2.19}$$

$$= p(x) * \mathcal{N}\Big(0, \big(\sigma^2(k\Delta t) - \sigma^2(0)\big)\Big) \tag{4.2.20}$$

Since $p(x)$ is a valid distribution, assuming $p(x)$ has zero-mean[1], for large values of t, and large enough $\sigma^2(t)$

$$p_t(x) \approx \mathcal{N}\Big(0, \big(\sigma^2(t) - \sigma^2(0)\big)\Big) \tag{4.2.21}$$

Hence, we can start with $\mathbf{x}_T$ drawn from $\mathcal{N}\Big(0, \big(\sigma^2(T) - \sigma^2(0)\big)\Big)$, which is easy to do, and then run equation (4.2.10) to obtain samples from data distribution. Since the variance of the forward SDE keeps increasing as time progress, this class of methods is known as Variance Exploding diffusion process, this approach was first introduced in [15]. We provide a toy example below

Example 7. *(Variance exploding) Using the same setting as in Example 6, from* (4.2.17) *we have the density at time t*

$$p_t(x) = \Big(\alpha\mathcal{N}(\mu_1, \sigma_1^2) + (1-\alpha)\mathcal{N}(\mu_2, \sigma_2^2)\Big) * \mathcal{N}\Big(0, \big(\sigma^2(t) - \sigma^2(0)\big)\Big) \tag{4.2.22}$$

$$= \alpha\mathcal{N}\Big(\mu_1, \sigma_1^2 + \sigma^2(t) - \sigma^2(0)\Big) + (1-\alpha)\mathcal{N}\Big(\mu_2, \sigma_2^2 + \sigma^2(t) - \sigma^2(0)\Big) \tag{4.2.23}$$

Then the score is

$$\nabla \log p_t(x) = -\frac{\frac{x-\mu_1}{\sigma_1^2+\sigma^2(t)-\sigma^2(0)}\alpha\mathcal{N}(\mu_1, \sigma_1^2 + \sigma^2(t) - \sigma^2(0)) + \frac{x-\mu_2}{\sigma_2^2+\sigma^2(t)-\sigma^2(0)}(1-\alpha)\mathcal{N}(\mu_2, \sigma_2^2 + \sigma^2(t) - \sigma^2(0))}{\alpha\mathcal{N}(\mu_1, \sigma_1^2 + \sigma^2(t) - \sigma^2(0)) + (1-\alpha)\mathcal{N}(\mu_2, \sigma_2^2 + \sigma^2(t) - \sigma^2(0))} \tag{4.2.24}$$

Using this score, we can run (4.2.10) *starting from $\mathbf{x}_T \sim \mathcal{N}\Big(0, \big(\sigma^2(T) - \sigma^2(0)\big)\Big)$ to get back to the data distribution. This is demonstrated in Figure 4 and 5. In this example, $\sigma(0) = 0.001, \sigma(T) = 10$, total time steps is 5000.*

[1] In practice, this is achieved by shifting the data by the sample mean.

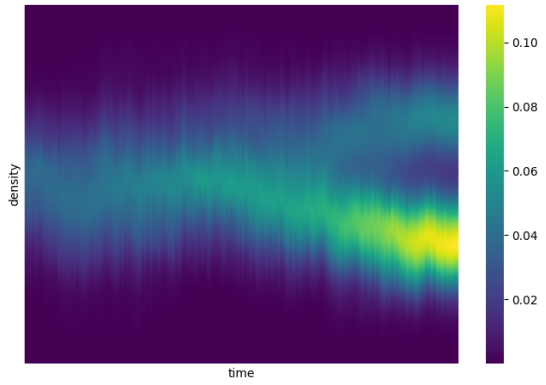

Figure 4: Evolution of the density from Example 7 . Starting from $\mathcal{N}(0, (\sigma^2(t) - \sigma^2(0)))$, (4.2.10) is run to obtain samples from the Gaussian mixture.

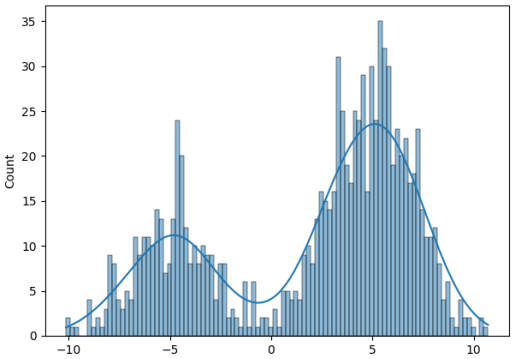

Figure 5: Samples generated by VE diffusion sampling, which match the true Gaussian mixture with $\mu_1 = -5$, $\mu_2 = 5$, $\sigma_1 = \sigma_2 = 2$, $\alpha = 0.3$

In Examples 6 and 7, it looks like using Langevin dynamics is much simpler. But in practice, for real dataset, estimating $\nabla \log p_t(\mathbf{x})$ is more tractable, since this is score of noisy version of the data and hence its distribution is supported everywhere. Furthermore, when $\sigma(t)$ changes slowly, $\nabla \log p_t(\mathbf{x})$ is small and can be easily approximated by a neural network.

As the name may have suggested, we can also define SDEs that will preserve the variance during the forward pass. This motivates the methods discussed in the next section.

4.2.2 Variance Preserving methods

At the same time as [15], the concurrent works of [3] introduced a diffusion-based generative modelling technique utilizing VP methods. In the original paper, the authors formulate the task using Markov chain approach. Here, we will reformulate it using the theory of SDEs introduced so far. Consider the SDE

$$d\mathbf{x}_t = -\frac{1}{2}\sigma(t)\mathbf{x}dt + \sqrt{\sigma(t)}d\boldsymbol{\beta}(t) \tag{4.2.25}$$

Its discretization is

$$\mathbf{x}_k = \mathbf{x}_{k-1} - \frac{1}{2}\sigma(k-1)\mathbf{x}_{k-1}\Delta t + \sqrt{\sigma(t-1)\Delta t}\mathbf{n}, \quad \mathbf{n} \sim \mathcal{N}(\mathbf{0}, \mathbf{I}) \tag{4.2.26}$$

$$= \mathbf{x}_{k-1}\big(1 - \frac{1}{2}\sigma(k-1)\Delta t\big) + \sqrt{\sigma(k-1)\Delta t}\mathbf{n} \tag{4.2.27}$$

Using the Taylor series of $\sqrt{1-x} = 1 - \frac{x}{2} + O(x^2)$, for small Δt and $0 < \sigma(t) < 1$ we have

$$\mathbf{x}_k \approx \mathbf{x}_{k-1}\sqrt{1 - \sigma(k-1)\Delta t} + \sqrt{\sigma(k-1)\Delta t}\mathbf{n} \tag{4.2.28}$$

$$\approx \mathbf{x}_{k-1}\sqrt{1 - \sigma(k)\Delta t} + \sqrt{\sigma(k)\Delta t}\mathbf{n}, \quad \text{for small } \Delta t \tag{4.2.29}$$

$$= \mathbf{x}_{k-1}\sqrt{1 - \gamma(k)} + \sqrt{\gamma(k)}\mathbf{n}, \quad \gamma(k) = \sigma(k)\Delta t \tag{4.2.30}$$

(4.2.30) is the equation used in [3] as forward pass. Using Markov chain, the authors argued that the chain defined by (4.2.30) has stationary distribution $\mathcal{N}(\mathbf{0}, \mathbf{I})$. Here, we will prove that by Fokker-Planck equation, using

1-dimensional case for simplification

$$\frac{\partial p_t}{\partial t} = \frac{1}{2}\frac{\partial}{\partial x}\sigma(t)p_t(x)x + \frac{1}{2}\frac{\partial^2}{\partial x^2}\sigma(t)p_t(x) \tag{4.2.31}$$

$$= \frac{1}{2}\sigma(t)\frac{\partial}{\partial x}p_t(x)x + \frac{1}{2}\sigma(t)\frac{\partial^2}{\partial x^2}p_t(x) = 0,\ t \to \infty \tag{4.2.32}$$

$$\Rightarrow \frac{\partial}{\partial x}\Big(p_\infty(x)x + \frac{\partial}{\partial x}p_\infty(x)\Big) = 0 \tag{4.2.33}$$

$$\Rightarrow p_\infty(x)x + \frac{\partial}{\partial x}p_\infty(x) = 0 \tag{4.2.34}$$

$$\Rightarrow p_\infty(x) \propto \exp(-\frac{1}{2}x^2) \tag{4.2.35}$$

Since $p_t(x)$ must be a valid distribution, thus $p_t(x) = \frac{1}{\sqrt{2\pi}}\exp(-\frac{1}{2}x^2)$, concluding the short proof. Another way is to show that (4.2.30) is Gaussian process (Example 11), and if data distribution has identity covariance[2], then $p_t(\mathbf{x})$ also has identity variance (Example 8). For sampling, the reverse SDE is

$$d\mathbf{x}_{T-t} = \Big(\frac{1}{2}\sigma(T-t)\mathbf{x} + \sigma(T-t)\nabla\log p_{T-t}(\mathbf{x})\Big)dt + \sqrt{\sigma(T-t)}d\boldsymbol{\beta}(T-t) \tag{4.2.36}$$

And the discretization is

$$\mathbf{x}_{K-k} = \mathbf{x}_{K-k+1} + \Big(\frac{1}{2}\sigma(K-k+1)\mathbf{x}_{K-k+1} + \sigma(K-k+1)\nabla\log p_{K-k+1}(\mathbf{x}_{K-k+1})\Big)\Delta t \tag{4.2.37}$$

$$+ \sqrt{\sigma(K-k+1)\Delta t}\mathbf{n} \tag{4.2.38}$$

$$= \mathbf{x}_{K-k+1}\big(1 + \frac{1}{2}\sigma(K-k+1)\Delta t\big) + \sigma(K-k+1)\Delta t\nabla\log p_{K-k+1}(\mathbf{x}_{K-k+1}) \tag{4.2.39}$$

$$+ \sqrt{\sigma(K-k+1)\Delta t}\mathbf{n} \tag{4.2.40}$$

$$\approx \mathbf{x}_{K-k+1}\big(1 + \frac{1}{2}\gamma(K-k)\big) + \gamma(K-k)\nabla\log p_{K-k+1}(\mathbf{x}_{K-k+1}) + \sqrt{\gamma(K-k)}\mathbf{n} \tag{4.2.41}$$

$$\approx \mathbf{x}_{K-k+1}\big(1 + \frac{1}{2}\gamma(K-k)\big) + \gamma(K-k)\nabla\log p_{K-k+1}(\mathbf{x}_{K-k+1}) \tag{4.2.42}$$

$$+ \frac{1}{2}\gamma^2(K-k)\nabla\log p_{K-k+1}(\mathbf{x}_{K-k+1}) + \sqrt{\gamma(K-k)}\mathbf{n},\ \text{for small } \Delta t \tag{4.2.43}$$

$$= \big(1 + \frac{1}{2}\gamma(K-k)\big)\Big(\mathbf{x}_{K-k+1} + \gamma(K-k)\nabla\log p_{K-k+1}(\mathbf{x}_{K-k+1})\Big) + \sqrt{\gamma(K-k)}\mathbf{n} \tag{4.2.44}$$

$$\approx \frac{1}{\sqrt{1-\gamma(K-k)}}\Big(\mathbf{x}_{K-k+1} + \gamma(K-k)\nabla\log p_{K-k+1}(\mathbf{x}_{K-k+1})\Big) + \sqrt{\gamma(K-k)}\mathbf{n} \tag{4.2.45}$$

where the last line was achieved by Taylor series of $\frac{1}{\sqrt{1-x}}$. Equation (4.2.30) is used in [3] during the sampling phase, assuming $\nabla\log p_k(\mathbf{x}_k)$ is already learned during the forward pass. Interestingly, the work in [3] does not estimate $\nabla\log p_k(\mathbf{x}_k)$. Instead, it directly estimates the noise added during the forward pass, i.e. the noise term $\mathbf{n}$ from (4.2.30). For instance, draw $\mathbf{n} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$, let $\mathbf{x}_k = \mathbf{x}_{k-1}\big(1 - \frac{1}{2}\sigma(k-1)\Delta t\big) + \sqrt{\sigma(k-1)\Delta t}\mathbf{n}$, and $\mathbf{n}_{\boldsymbol{\theta}}(\mathbf{x}_k)$ be output of a neural network that minimizes

$$||\mathbf{n} - \mathbf{n}_{\boldsymbol{\theta}}(\mathbf{x}_k)||_2^2 \tag{4.2.46}$$

then $\mathbf{n}_{\boldsymbol{\theta}}(\mathbf{x}_k) \propto \mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}_k)$, where $\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}_k)$ minimizes

$$||\nabla\log p_t(\mathbf{x}_t) - \mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}_t)||_2^2 \tag{4.2.47}$$

To prove this relation, first, we note the following property (proof in Lemma 2)

$$\mathbb{E}_{p_t(.)}\Big[||\nabla\log p_t(\mathbf{x}_t) - \mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}_t)||_2^2\Big] = \mathbb{E}_{\mathbf{x}_0\sim p_{\text{data}}}\mathbb{E}_{\mathbf{x}_t|\mathbf{x}_0}\Big[||\nabla\log p_t(\mathbf{x}_t|\mathbf{x}_0) - \mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}_t)||_2^2\Big] + \text{const} \tag{4.2.48}$$

[2]Can be achieved by standardization methods.

Next, note that $p_{t|0}(\mathbf{x}_t|\mathbf{x}_0)$ is Gaussian, this can be derived from the discretization in (4.2.30)

$$\mathbf{x}_k = \mathbf{x}_{k-1}\sqrt{1-\gamma(k)} + \sqrt{\gamma(k)}\mathbf{n} \tag{4.2.49}$$

$$= \big(\mathbf{x}_{k-2}\sqrt{1-\gamma(k-1)} + \sqrt{\gamma(k-1)}\mathbf{n}\big)\sqrt{1-\gamma(k)} + \sqrt{\gamma(k)}\mathbf{n} \tag{4.2.50}$$

$$= \dots \tag{4.2.51}$$

$$= \mathbf{x}_0 \prod_{i=0}^{k} \sqrt{1-\gamma(k-i)} + \sqrt{\sum_{i=0}^{k} \Big(\gamma(k-i)\prod_{j=0}^{i-1}\big(1-\gamma(k-j)\big)\Big)}\mathbf{n} \tag{4.2.52}$$

$$\mathbf{x}_k \sim \mathcal{N}\left(\mathbf{x}_0 \prod_{i=0}^{k} \sqrt{1-\gamma(k-i)}, \sum_{i=0}^{k} \Big(\gamma(k-i)\prod_{j=0}^{i-1}\big(1-\gamma(k-j)\big)\Big)\right) \tag{4.2.53}$$

For the continuous-time equation (4.2.25), Example 11 shows that

$$p_{t|0}(\mathbf{x}_t|\mathbf{x}_0) = \mathcal{N}\Big(\mathbf{x}_0 e^{-\int_0^t \sigma(\tau)d\tau}, \mathbf{I}(1-e^{-\int_0^t \sigma(\tau)d\tau})\Big) \tag{4.2.54}$$

Thus, we have closed-form expression for $\nabla \log p_{t|0}(\mathbf{x}_t|\mathbf{x}_0)$

$$\nabla \log p_{t|0}(\mathbf{x}_t|\mathbf{x}_0) = -\frac{\mathbf{x}_t - \mathbf{x}_0 e^{-\int_0^t \sigma(\tau)d\tau}}{1-e^{-\int_0^t \sigma(\tau)d\tau}} = -\frac{\mathbf{x}_t - \text{mean}(\mathbf{x}_t)}{\text{var}(\mathbf{x}_t)} = -\frac{\boldsymbol{\epsilon}_t}{\sqrt{\text{var}(\mathbf{x}_t)}} \tag{4.2.55}$$

where $\boldsymbol{\epsilon}_t = \frac{\mathbf{x}_t - \text{mean}(\mathbf{x}_t)}{\sqrt{\text{var}(\mathbf{x}_t)}}$, it is now apparent that $\boldsymbol{\epsilon}_t$ is exactly the same as the noise term $\mathbf{n}$ in (4.2.30). Plugging (4.2.55) into (4.2.48), we have

$$\mathbb{E}_{\mathbf{X}_0\sim p_{\text{data}}}\mathbb{E}_{\mathbf{X}_t|\mathbf{X}_0}\Big[||\nabla \log p_t(\mathbf{x}_t|\mathbf{x}_0) - \mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}_t)||_2^2\Big] = \mathbb{E}_{\mathbf{X}_0\sim p_{\text{data}}}\mathbb{E}_{\mathbf{X}_t|\mathbf{X}_0}\Big[||-\frac{\boldsymbol{\epsilon}_t}{\sqrt{\text{var}(\mathbf{x}_t)}} - \mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}_t)||_2^2\Big] \tag{4.2.56}$$

$$\propto \mathbb{E}_{\mathbf{X}_0\sim p_{\text{data}}}\mathbb{E}_{\mathbf{X}_t|\mathbf{X}_0}\Big[||\boldsymbol{\epsilon}_t + \sqrt{\text{var}(\mathbf{x}_t)}\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}_t)||_2^2\Big] \tag{4.2.57}$$

Compare this to (4.2.46) and (4.2.47), it is clear that $\mathbf{n}_{\boldsymbol{\theta}}(\mathbf{x}_t) = -\sqrt{\text{var}(\mathbf{x}_t)}\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}_t)$, concluding the proof. (4.2.57), usually referred to as denoising objective, is much easier to implement in practice compared to score-matching objective introduced in Section 4.1, since it does not involve estimating the true score function or calculating the trace of the Jacobian. In [3], this objective was derived using the ELBO introduced in Section 3, here we showed that it can be derived from SDE theory. We note that this equivalence only happens because $p_{t|0}(\mathbf{x}_t|\mathbf{x}_0)$ is Gaussian, otherwise $\nabla \log p_{t|0}(\mathbf{x}_t|\mathbf{x}_0)$ would be intractable, thus the denoising objective would be invalid.

5 Appendix

5.1 ODE examples

Example 8. *(Linear Time-varying) Following the analysis in Section 4.2.2, $\boldsymbol{\Sigma}(t)$ denote variance of $p_t(\mathbf{x})$. Then, according to Section 5.5 of [8], $\boldsymbol{\Sigma}(t)$ has the following form*

$$\frac{d\boldsymbol{\Sigma}(t)}{dt} = -\sigma(t)\boldsymbol{\Sigma}(0) + \sigma(t)\mathbf{I} \tag{5.1.1}$$

Since $\sigma(t)$ is scalar-valued, by [5], the state transition matrix is

$$\boldsymbol{\Phi}(t, t_0) = e^{\int_{t_0}^{t} -\sigma(\tau)d\tau} \tag{5.1.2}$$

Hence, we have the solution

$$\begin{aligned}
\boldsymbol{\Sigma}(t) &= \boldsymbol{\Sigma}(0)e^{-\int_0^t \sigma(\tau)d\tau} + \int_0^t e^{-\int_\tau^t \sigma(\eta)d\eta}\sigma(\tau)d\tau\mathbf{I} && (5.1.3)\\
&= \boldsymbol{\Sigma}(0)e^{-\int_0^t \sigma(\tau)d\tau} + e^{-\int_0^t \sigma(\tau)d\tau}\int_0^t e^{\int_0^\tau \sigma(\eta)d\eta}\sigma(\tau)d\tau\mathbf{I} && (5.1.4)\\
&= \boldsymbol{\Sigma}(0)e^{-\int_0^t \sigma(\tau)d\tau} + e^{-\int_0^t \sigma(\tau)d\tau}\int_0^t d\Big(e^{\int_0^\tau \sigma(\eta)d\eta}\Big)\mathbf{I} && (5.1.5)\\
&= \boldsymbol{\Sigma}(0)e^{-\int_0^t \sigma(\tau)d\tau} + \mathbf{I}e^{-\int_0^t \sigma(\tau)d\tau}e^{\int_0^\tau \sigma(\eta)d\eta}\Big|_{\tau=0}^t && (5.1.6)\\
&= \boldsymbol{\Sigma}(0)e^{-\int_0^t \sigma(\tau)d\tau} + \mathbf{I}e^{-\int_0^t \sigma(\tau)d\tau}\Big(e^{\int_0^t \sigma(\eta)d\eta} - 1\Big) && (5.1.7)\\
&= \boldsymbol{\Sigma}(0)e^{-\int_0^t \sigma(\tau)d\tau} + \mathbf{I} - \mathbf{I}e^{-\int_0^t \sigma(\tau)d\tau} && (5.1.8)\\
&= \mathbf{I} + e^{-\int_0^t \sigma(\tau)d\tau}\Big(\boldsymbol{\Sigma}(0) - \mathbf{I}\Big) && (5.1.9)
\end{aligned}$$

5.2 Riemann's sum, Itô's integral, and related proofs

Riemann's sum. Consider the following integral, where $w(t)$ is white noise process

$$\int_{t_0}^t f(t)w(t)dt \tag{5.2.1}$$

Its Riemann's sum representation is

$$\lim_{n\to\infty}\sum_k f(t_k^*)w(t_k^*)(t_k - t_{k-1}) \tag{5.2.2}$$

where $t_k - t_{k-1} = \frac{t-t_0}{n}$, and $t_{k-1} \le t^* \le t_k$. The Riemann's integral exists if the limit converges with any choice of $t_k^* : t_{k-1} \le t_k^* \le t_k$. This is clearly not the case, since $w(t_k^*)$ has different realizations with different t_k^*, hence, the Riemann's integral does not exist.

However, if we fix t_k^*, i.e. to be the starting point $t_k^* = t_k$, then the following limit is unique

$$\lim_{n\to\infty}\sum_k f(t_k)w(t_k)(t_{k+1} - t_k) = \lim_{n\to\infty}\sum_k f(t_k)(\beta(t_{k+1}) - \beta(t_k)) \tag{5.2.3}$$

Itô's integral. The previous choice of t_k has an important implication, consider the following integral

$$\begin{aligned}
\int_0^t \beta(s)d\beta(s) &= \lim_{n\to\infty}\sum_k \beta(t_k)(\beta(t_{k+1}) - \beta(t_k)) && (5.2.4)\\
&= \lim_{n\to\infty}\sum_k \Big(\beta(t_k)\beta(t_{k-1}) - \beta^2(t_k)\Big) && (5.2.5)\\
&= \lim_{n\to\infty}\sum_k \Big(-\frac{1}{2}\beta^2(t_k) + \beta(t_k)\beta(t_{k-1}) - \frac{1}{2}\beta^2(t_{k-1}) + \frac{1}{2}\beta^2(t_k) - \frac{1}{2}\beta^2(t_{k-1})\Big) && (5.2.6)\\
&= \lim_{n\to\infty}\sum_k \Big(-\frac{1}{2}\big(\beta(t_k) - \beta(t_{k-1})\big)^2 + \frac{1}{2}\big(\beta^2(t_k) - \beta^2(t_{k-1})\big)\Big) && (5.2.7)
\end{aligned}$$

Since $\beta(t_k) - \beta(t_{k-1})$ has variance Δt, the limits are evaluated as follow

$$\begin{aligned}
\lim_{n\to\infty}\sum_k \Big(-\frac{1}{2}\big(\beta(t_k) - \beta(t_{k-1})\big)^2\Big) &= -\frac{1}{2}t && (5.2.8)\\
\lim_{n\to\infty}\sum_k \Big(\frac{1}{2}\big(\beta^2(t_k) - \beta^2(t_{k-1})\big)\Big) &= \frac{1}{2}\beta^2(t) && (5.2.9)
\end{aligned}$$

hence,

$$\int_0^t \beta(s)d\beta(s) = -\frac{1}{2}t + \frac{1}{2}\beta^2(t) \tag{5.2.10}$$

or in differential form

$$d\Big(\frac{1}{2}\beta^2(t)\Big) = \beta(t)d\beta(t) + \frac{1}{2}t \tag{5.2.11}$$

Note the new term $\frac{1}{2}t$, which did not appear in normal chain rule, this is a direct consequence of fixing $t_k^* = t_k$.

Lemma 1. *Let $f(t)$ be some function that is square-integrable, i.e. $\int_0^t f^2(s)ds < \infty$, and $\beta(t)$ be a some Brownian motion, then*

$$\int_0^t f(s)d\beta \sim \mathcal{N}\Big(0, \int_0^t f^2(s)ds\Big) \tag{5.2.12}$$

Proof. Since Riemann's sum of $\int_0^t f(s)d\beta$ exists if we fix the midpoints, let $t_k = \frac{k}{2^n}t$, then

$$\int_0^t f(s)d\beta = \lim_{n\to\infty} \sum_{k=0}^{2^n-1} f(t_k)(\beta_{t_{k+1}} - \beta_{t_k}) \tag{5.2.13}$$

and since increment of Brownian motion follows $\mathcal{N}(0, \Delta t)$, where $\Delta t = 2^{-n}t$, thus

$$\sum_{k=0}^{2^n-1} f(t_k)(\beta_{t_{k+1}} - \beta_{t_k}) \sim \mathcal{N}\Big(0, \sum_{k=0}^{2^n-1} f^2(t_k)2^{-n}t\Big) \tag{5.2.14}$$

Now we can take the limit

$$\lim_{n\to\infty} \sum_{k=0}^{2^n-1} f^2(t_k)2^{-n}t = \lim_{n\to\infty} \sum_{k=0}^{2^n-1} f^2(t_k)\Delta t \tag{5.2.15}$$

$$= \int_0^t f^2(s)ds \tag{5.2.16}$$

this completes the proof.

5.3 SDE examples

Example 9. *(Stock price) In this example we let $P(t)$ denotes the stock price at time t, we will model the relative price change $\frac{dP}{P}$ as the following LTI SDE:*

$$\frac{dP}{P} = \mu dt + \sigma d\beta \tag{5.3.1}$$

where $P = P(t)$ and $\beta = \beta(t)$ is Brownian motion, μ and σ are constants, we have omitted the temporal argument for the sake of ease to read.

Now we can apply Itô's formula from (2.2.11) to the function $\log P$

$$d\log P = \frac{1}{P}dP - \frac{1}{2}\frac{1}{P^2}dP^2 \tag{5.3.2}$$

$$= \frac{1}{P}(\mu P dt + \sigma P d\beta) - \frac{1}{2}\frac{1}{P^2}(\mu P dt + \sigma dP\beta)^2 \tag{5.3.3}$$

$$= \mu dt + \sigma d\beta - \frac{1}{2}(\mu^2 \underbrace{dtdt}_{=0} + + \mu\sigma \underbrace{dtd\beta}_{=0} + \sigma^2 \underbrace{d\beta d\beta}_{=dt})^2 \tag{5.3.4}$$

$$= (\mu - \frac{1}{2}\sigma^2)dt + \sigma d\beta \tag{5.3.5}$$

$$\Rightarrow \log P - \log P_0 = (\mu - \frac{1}{2}\sigma^2)t + \sigma w(t) \tag{5.3.6}$$

$$\Rightarrow P(t) = P_0 e^{(\mu - \frac{1}{2}\sigma^2)t + \sigma w(t)} \tag{5.3.7}$$

where $w(t)$ is white noise, i.e. $w(t) \sim \mathcal{N}(0,1)$ if $\beta(t)$ has identity diffusion matrix.

Example 10. *(Ornstein-Uhlenbeck process) The one-dimensional mean-reverting process, also known as the Ornstein-Uhlenbeck process, is given as*

$$dx = -\mu x dt + \sigma d\beta \tag{5.3.8}$$

where $\beta = \beta(t)$ is Brownian motion, μ and σ are constants, we again have omitted the temporal argument for the sake of ease to read.

Here, instead of $\log x$*, we applied Itô's formula to the function $xe^{\mu t}$*

$$\begin{aligned} dxe^{\mu t} &= e^{\mu t}dx + \mu x e^{\mu t}dt && (5.3.9)\\ &= e^{\mu t}(-\mu x dt + \sigma d\beta) + \mu x e^{\mu t}dt && (5.3.10)\\ &= \sigma e^{\mu t}d\beta && (5.3.11)\\ \Rightarrow xe^{\mu s}\Big|_0^t &= \int_0^t \sigma e^{\mu s}d\beta && (5.3.12)\\ \Rightarrow xe^{\mu t} &= x_0 + \int_0^t \sigma e^{\mu s}d\beta && (5.3.13)\\ \Rightarrow x(t) &= x_0 e^{-\mu t} + \int_0^t \sigma e^{-\mu(t-s)}d\beta && (5.3.14) \end{aligned}$$

By applying Lemma 1, we have:

$$\begin{aligned} \int_0^t \sigma e^{-\mu(t-s)}d\beta &\sim \mathcal{N}\Big(0, \int_0^t \sigma^2 e^{-2\mu(t-s)}ds\Big) && (5.3.15)\\ &\sim \mathcal{N}\Big(0, \frac{\sigma^2}{2\mu}(1-e^{-2\mu t})\Big) && (5.3.16) \end{aligned}$$

thus,

$$x(t) = x_0 e^{-\mu t} + \frac{\sigma}{\sqrt{2\mu}}w(t) \tag{5.3.17}$$

where $w(t) \sim \mathcal{N}(0, 1-e^{-2\mu t})$

Note that the mean of $x(t)$ will exponentially go to 0 as time progress, hence the name mean-reverting process.

Example 11. *(Time-variant Ornstein-Uhlenbeck process) Consider the following SDE*

$$dx = -\mu(t)x dt + \eta(t)d\beta \tag{5.3.18}$$

where $\beta = \beta(t)$ is Brownian motion, $\mu(t)$ and $\eta(t)$ are now time-varying function.

Here, instead of $xe^{\mu t}$, we applied Itô's formula to the function $xe^{\int_0^t \mu(\tau)d\tau}$

$$\begin{aligned} dxe^{\int_0^t \mu(\tau)d\tau} &= e^{\int_0^t \mu(\tau)d\tau}dx + \mu(t)xe^{\int_0^t \mu(\tau)d\tau}dt && (5.3.19)\\ &= e^{\int_0^t \mu(\tau)d\tau}(-\mu(t)xdt + \eta(t)d\beta) + \mu(t)xe^{\int_t^t \mu(\tau)d\tau}dt && (5.3.20)\\ &= \eta(t)e^{\int_0^t \mu(\tau)d\tau}d\beta && (5.3.21)\\ dxe^{\gamma(t)} &= \eta(t)e^{\gamma(t)}d\beta, \quad \gamma(t) = \int_0^t \mu(\tau)d\tau && (5.3.22)\\ \Rightarrow xe^{\gamma(s)}\Big|_0^t &= \int_0^t \eta(s)e^{\gamma(s)}d\beta && (5.3.23)\\ \Rightarrow xe^{\gamma(t)} &= x_0 + \int_0^t \eta(s)e^{\gamma(s)}d\beta && (5.3.24)\\ \Rightarrow x(t) &= x_0 e^{-\gamma(t)} + \int_0^t \eta(s)e^{-\gamma(t)+\gamma(s)}d\beta && (5.3.25) \end{aligned}$$

By applying Lemma 1, we have:

$$\int_0^t \eta e^{-\mu(t-s)}d\beta \sim \mathcal{N}\Big(0, \int_0^t \eta^2(s)e^{-2\gamma(t)+2\gamma(s)}ds\Big) \tag{5.3.26}$$

thus $x(t)$ is still a Gaussian process. Now, suppose $\mu(t) = \frac{1}{2}\sigma(t)$, and $\eta(t) = \sqrt{\sigma(t)}$, then

$$\int_0^t \eta^2(s) e^{-2\gamma(t)+2\gamma(s)} ds = e^{-\int_0^t \sigma(\tau)d\tau} \int_0^t \sigma(s) e^{\int_0^s \sigma(\tau)d\tau} ds \tag{5.3.27}$$

$$= e^{-\int_0^t \sigma(\tau)d\tau} \int_0^t d\Big(e^{\int_0^s \sigma(\tau)d\tau}\Big) \tag{5.3.28}$$

$$= e^{-\int_0^t \sigma(\tau)d\tau} e^{\int_0^s \sigma(\tau)d\tau}\Big|_{s=0}^t \tag{5.3.29}$$

$$= e^{-\int_0^t \sigma(\tau)d\tau}\Big(e^{\int_0^t \sigma(\tau)d\tau} - 1\Big) = 1 - e^{-\int_0^t \sigma(\tau)d\tau} \tag{5.3.30}$$

Thus, given x_0, $p_{t|0}(x_t|x_0) = \mathcal{N}(x_0 e^{-\int_0^t \sigma(\tau)d\tau}, 1 - e^{-\int_0^t \sigma(\tau)d\tau})$

Example 12. *(Stationary distribution of SDE) This example considers the following linear time-variant SDE*

$$dx = -\frac{1}{2}\sigma(t)x dt + \sqrt{\sigma(t)} d\beta(t) \tag{5.3.31}$$

Using the initial condition $p(x, 0) = \delta(x - x_0)$, from Example 11, we have

$$p_{t|0}(x_t|x_0) = \mathcal{N}(x_0 e^{-\int_0^t \sigma(\tau)d\tau}, 1 - e^{-\int_0^t \sigma(\tau)d\tau}) \tag{5.3.32}$$

$$= \frac{1}{\sqrt{2\pi(1-\kappa_t)^2}} \exp\Big(-\frac{1}{2}\frac{\big(x - x_0\kappa_t\big)^2}{1-\kappa_t}\Big) \tag{5.3.33}$$

$$\kappa_t = e^{-\int_0^t \sigma(\tau)d\tau} \tag{5.3.34}$$

Thus, by chain rule

$$p(x,t) = p_{t|p_0(x)}(x_t|p_0(x)) = \int p_{t|0}(x_t|x_0) p_0(x_0) dx_0 \tag{5.3.35}$$

$$= \int \frac{1}{\sqrt{2\pi(1-\kappa_t)^2}} \exp\Big(-\frac{1}{2}\frac{\big(x - s\kappa_t\big)^2}{1-\kappa_t}\Big) p_0(s) ds \tag{5.3.36}$$

From this, we can see that $p(x, t)$ converges to $\mathcal{N}(0, 1)$ exponentially fast. If we define $\circledast_\alpha$ as the following convolution-like operation

$$f(x) \circledast_\alpha g(x) = \int f(s) g(x - \alpha s) ds \tag{5.3.37}$$

then

$$p(x,t) = p_0(x) \circledast_{\kappa_t} \mathcal{N}\Big(0, 1 - \kappa_t\Big) \tag{5.3.38}$$

5.4 Score matching and diffusion models

Lemma 2.

$$\mathbb{E}_{p_t(.)}\Big[||\nabla \log p_t(\mathbf{x}_t) - \mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}_t)||_2^2\Big] = \mathbb{E}_{\mathbf{X}_0 \sim p_{data}} \mathbb{E}_{\mathbf{X}_t|\mathbf{X}_0}\Big[||\nabla \log p_t(\mathbf{x}_t|\mathbf{x}_0) - \mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}_t)||_2^2\Big] + const \tag{5.4.1}$$

Proof. We have

$$\mathbb{E}_{p_t(.)}\Big[||\nabla \log p_t(\mathbf{x}_t) - \mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}_t)||_2^2\Big] = \mathbb{E}_{p_t(.)}\Big[||\nabla \log p_t(\mathbf{x}_t)||_2^2 + ||\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}_t)||_2^2 - 2\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}_t)^T \nabla \log p_t(\mathbf{x}_t)\Big] \tag{5.4.2}$$

$$\mathbb{E}_{p_t(.)}\Big[\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}_t)^T \nabla \log p_t(\mathbf{x}_t)\Big] = \int_{\mathbf{x}_t} \mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}_t)^T p_t(\mathbf{x}_t) \nabla \log p_t(\mathbf{x}_t) d\mathbf{x}_t \tag{5.4.3}$$

$$= \int_{\mathbf{x}_t} \mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}_t)^T \nabla p_t(\mathbf{x}_t) d\mathbf{x}_t = \int_{\mathbf{x}_t} \mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}_t)^T \int_{\mathbf{x}_0} \nabla p_{t,0}(\mathbf{x}_t, \mathbf{x}_0) d\mathbf{x}_0 d\mathbf{x}_t \tag{5.4.4}$$

$$= \int_{\mathbf{x}_t} \mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}_t)^T \int_{\mathbf{x}_0} p_0(\mathbf{x}_0) \nabla p_{t|0}(\mathbf{x}_t|\mathbf{x}_0) d\mathbf{x}_0 d\mathbf{x}_t \;\; (\nabla \textit{is taken w.r.t } \mathbf{x}_t) \tag{5.4.5}$$

$$= \int_{\mathbf{x}_0} p_0(\mathbf{x}_0) \int_{\mathbf{x}_t} \mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}_t)^T p_{t|0}(\mathbf{x}_t|\mathbf{x}_0) \nabla \log p_{t|0}(\mathbf{x}_t|\mathbf{x}_0) d\mathbf{x}_t d\mathbf{x}_0 \tag{5.4.6}$$

$$= \mathbb{E}_{\mathbf{x}_0 \sim p_{data}} \mathbb{E}_{\mathbf{x}_t|\mathbf{x}_0}\Big[||\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}_t)^T \nabla p_{t|0}(\mathbf{x}_t|\mathbf{x}_0)||_2^2\Big] \tag{5.4.7}$$

$$\mathbb{E}_{p_t(.)}\Big[||\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}_t)||_2^2\Big] = \int_{\mathbf{x}_t} ||\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}_t)||_2^2 p_t(\mathbf{x}_t) d\mathbf{x}_t \tag{5.4.8}$$

$$= \int_{\mathbf{x}_0} p_0(\mathbf{x}_0) \int_{\mathbf{x}_t} ||\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}_t)||_2^2 p_{t|0}(\mathbf{x}_t|\mathbf{x}_0) d\mathbf{x}_t d\mathbf{x}_0 \tag{5.4.9}$$

$$= \mathbb{E}_{\mathbf{x}_0 \sim p_{data}} \mathbb{E}_{\mathbf{x}_t|\mathbf{x}_0}\Big[||\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}_t)||_2^2\Big] \tag{5.4.10}$$

$$\mathbb{E}_{p_t(.)}\Big[||\nabla \log p_t(\mathbf{x}_t)||_2^2\Big] = const \tag{5.4.11}$$

From (5.4.7)*,* (5.4.10)*, and* (5.4.11)*,* (5.4.1) *is proved.*

References

[1] Y. Song and S. Ermon, "Generative modeling by estimating gradients of the data distribution," *Advances in neural information processing systems*, vol. 32, 2019.

[2] J. Song, C. Meng, and S. Ermon, "Denoising diffusion implicit models," *arXiv preprint arXiv:2010.02502*, 2020.

[3] J. Ho, A. Jain, and P. Abbeel, "Denoising diffusion probabilistic models," *Advances in neural information processing systems*, vol. 33, pp. 6840–6851, 2020.

[4] B. Øksendal and B. Øksendal, *Stochastic differential equations*. Springer, 2003.

[5] W. J. Rugh, *Linear system theory*. Prentice-Hall, Inc., 1996.

[6] K. Ito, K. Itô, K. Itô, J. Mathématicien, K. Itô, and J. Mathematician, *On stochastic differential equations*. American Mathematical Society New York, 1951, vol. 4.

[7] P. E. Kloeden, E. Platen, P. E. Kloeden, and E. Platen, *Stochastic differential equations*. Springer, 1992.

[8] S. Särkkä and A. Solin, *Applied stochastic differential equations*. Cambridge University Press, 2019, vol. 10.

[9] A. Agrawal and J. Domke, "Amortized variational inference for simple hierarchical models," *Advances in Neural Information Processing Systems*, vol. 34, pp. 21 388–21 399, 2021.

[10] D. P. Kingma and M. Welling, *Auto-encoding variational bayes*, 2022. arXiv: `1312.6114 [stat.ML]`.

[11] M. Welling and Y. W. Teh, "Bayesian learning via stochastic gradient langevin dynamics," in *Proceedings of the 28th international conference on machine learning (ICML-11)*, Citeseer, 2011, pp. 681–688.

[12] A. Hyvärinen and P. Dayan, "Estimation of non-normalized statistical models by score matching.," *Journal of Machine Learning Research*, vol. 6, no. 4, 2005.

[13] A. Paszke, S. Gross, S. Chintala, *et al.*, "Automatic differentiation in pytorch," 2017.

[14] M. J. Hancock, *Infinite spatial domains and the fourier transform*, 2006.

[15] Y. Song, J. Sohl-Dickstein, D. P. Kingma, A. Kumar, S. Ermon, and B. Poole, "Score-based generative modeling through stochastic differential equations," *arXiv preprint arXiv:2011.13456*, 2020.