

Stochastic Differential Equations(SDE) for Large-scale Machine Learning: Applications on Least Squares Stochastic Gradient Descent and Score-Based Generative Models

Selena Ge

RG4012@NYU.EDU

New York University Shanghai

Supervisor: Prof. Mathieu Lauriere

Abstract

My senior thesis explores the application of Stochastic Differential Equations (SDE) in machine learning, focusing on two key models: Stochastic Gradient Descent (SGD) for least squares problems and diffusion models, particularly Score-Based Generative Models. In the first part, we will find out the connection between SGD and SDE through detailed mathematical analysis and numerical simulations. In the second part, we will focus on diffusion models and investigate the contributions of Song et al. (2020) in demonstrating how Denoising Diffusion Probabilistic Models (DDPM) and Score Matching Langevin Dynamics (SMLD) align with the SDE framework. Through numerical experiments and detailed mathematical proofs, we try to emphasize the theoretical significance of SDE in the field of machine learning from different angles.

Keywords: Stochastic Differential Equations, Score-based Generative Model, DDPM, SMLD, Least-Squares, Stochastic Gradient Descent, Machine Learning

Contents

1	Introduction	4
2	Literature Review	5
2.1	Stochastic Differential Equations (SDE) and Stochastic Gradient Descent (SGD)	5
2.2	Denoising Diffusion Probabilistic Models (DDPM) and Score Matching Langevin Dynamics (SMLD)	6
3	Stochastic Gradient Descent (SGD) on Least Squares Problems	7
3.1	The least square problem: population loss.	7
3.2	The stochastic gradient descent	8
3.3	Simulations based on SGD	11
4	Continuous model of SGD & SDE	14
4.1	The requirement of a SDE model: connect SDE with SGD	15
4.2	Explicit form of the SDE models	17
4.3	Simulation for the OU process through the Euler-Maruyama method	20
5	Score-based Generative Model through SDE	21
5.1	Score Matching Langevin Dynamics (SMLD)	22
5.1.1	Problem Setting	22
5.1.2	Deduction of (Denoising) Score Matching	22
5.1.3	Score Matching Langevin Dynamics (SMLD)	26
5.2	Denoising diffusion probabilistic model (DDPM)	27
5.2.1	Diffusion models	28
5.2.2	Denoising autoencoders	34

5.2.3	Simulation for DDPM	37
5.3	DDPM and SMLD with SDE	40
5.3.1	Forward and reverse process with SDE	40
5.3.2	Connection between DDPM and SMLD with SDE	43
5.3.3	Connection between noise and score	45
6	Conclusion	47
7	Acknowledgement	48
8	Appendix A: Introduction to SDE	49
8.1	What is SDE(Stochastic Differential Equations)	49
8.2	Numerical Methods for SDE	53
8.3	Empirical Risk Minimization (ERM)	54
8.4	Stochastic Modified Equations(SME)	56
8.4.1	Heuristic Motivations	56
8.5	Ornstein-Uhlenbeck Process	57
8.5.1	Denfinition	57
8.5.2	Properties of Mean Reversion	58
8.5.3	Analytical Solution	58
9	Appendix B: Python code for simulations	59

1 Introduction

Stochastic Differential Equations (SDE) serve as an important tool in the mathematical modeling of randomness to deal with uncertainty. In this thesis, we explore the application of SDE to two significant machine learning models: Stochastic Gradient Descent (SGD) for least squares problems and Diffusion models, with a specific focus on score-based generative models introduced by Song et al. (2020).

In the first part, we focus on SGD, a foundational optimization technique in machine learning which is widely used to deal with high-dimensional data. This thesis uses Gaussian distributions as an explicit example to investigate the relationship between SGD and SDE. We identify the conditions under which SGD and SDE converge, providing detailed mathematical derivations of when they align with each other and revealing how they can be used effectively in least squares problems.

In the second part, we examine recent advancements in diffusion models with a main focus on the work of Song et al. (2020). Their research on score-based generative modeling through SDE has provided cutting-edge insights into generative modeling techniques. This thesis discusses Song et al. (2020)’s contributions in detail, focusing on how Score Matching Langevin Dynamics (SMLD)(Chao et al. (2022)) and Denoising Diffusion Probabilistic Models (DDPM)(Ho et al. (2020)) can be applied in the SDE framework. We will provide detailed mathematical proofs to clarify the connection between SMLD, DDPM, and SDE, demonstrating how score-based methods can reverse the noise addition process inherent in diffusion models.

To sum up, this thesis aims to emphasize the theoretical importance of SDE in machine learning. The methodology employed will involve numerical experiments, data visualization, and meticulous mathematical derivations to help explain each concept

comprehensively. Through simulations conducted on SGD with Gaussian Distributions, Ornstein-Uhlenbeck (OU) processes, and the DDPM model, we demonstrate the accuracy and efficiency of our approach, contributing to the integration of these machine learning paradigms within the framework of SDE.

2 Literature Review

2.1 Stochastic Differential Equations (SDE) and Stochastic Gradient Descent (SGD)

The development of Stochastic Differential Equations (SDE) has laid the groundwork for understanding dynamic systems influenced by randomness. Khasminskii (2011) provided foundational insights into the stability and applications of SDE by analyzing the long-term behavior of solutions to these equations under random perturbations. Moreover, Varga (2010) explored advanced mathematical tools for analyzing SDE by applying advanced techniques, such as Gershgorin circle theorems.

Stochastic Gradient Descent (SGD) emerged as a powerful optimization technique in machine learning. Initially presented by Robbins and Monro (1951), SGD’s effectiveness lies in its ability to incorporate stochasticity for high-dimensional data and large-scale optimization problems. Subsequent studies like Cevher and Vũ (2019) and Shamir and Zhang (2013) focused on the convergence properties of SGD, especially in non-smooth and over-parameterized optimization problems. Bach and Moulines (2011) provided a comprehensive analysis of stochastic approximation algorithms. This study underscores the versatility of SGD, demonstrating its effectiveness across a wide range of machine-learning problems.

The connection between SDE and SGD then became a crucial point for researchers

aiming to understand stochastic optimization dynamics. Li et al. (2019) introduced stochastic modified equations, providing a rigorous mathematical framework for understanding SGD’s dynamics. Benaïm (2006) explored the dynamics of stochastic approximation algorithms, offering further insights into how SDE underpin the stochastic nature of SGD. Moreover, Pesme et al. (2021) examined the implicit bias of SGD in high-dimensional learning scenarios, revealing the importance of stochasticity in optimization.

2.2 Denoising Diffusion Probabilistic Models (DDPM) and Score Matching Langevin Dynamics (SMLD)

Recent advancements in generative models have led to the development of score-based generative models, which use SDE to understand data distribution gradients such as the ones in DDPM and SMLD models. Song and Ermon (2019) were among the first to explicitly introduce the concept of using score matching to estimate gradients of the data distribution. This was further advanced by Ho et al. (2020), who proposed Denoising Diffusion Probabilistic Models (DDPM) and demonstrated their ability to generate high-quality samples by learning the reverse process of diffusion.

Continuously, Song et al. (2020) expanded on this concept, connecting score-based generative models with SDE, and laid out a framework that includes Score Matching Langevin Dynamics (SMLD) for generative modeling. The work by Chao et al. (2022) then emphasizes the importance of denoising likelihood score matching in conditional score-based data generation. Pabbaraju et al. (2023) discuss the theoretical advantages of score matching. De Bortoli et al. (2024) further investigate target score matching, contributing to the evolution of generative models that rely on SDE for high-dimensional data synthesis.

3 Stochastic Gradient Descent (SGD) on Least Squares

Problems

In this section, we are going to explore the application of stochastic gradient descent (SGD) to the classic least squares problem, which is a fundamental technique commonly applied in various machine learning problems. We begin by defining the least squares problem in the context of a regression setting. Our goal is to minimize the discrepancy between predicted outputs and actual outcomes under a linear assumption. We subsequently explain the mathematical proofs of the problem and introduce the SGD method as a powerful tool for optimizing this type of loss function. We will explain how SGD adapts to the intricacies of the least squares framework and conduct an in-depth analysis of its implementation and performance. The detailed explanation of fundamental SDE concepts and formulas referenced in this and the following sections can be found in Appendix A.

3.1 The least square problem: population loss.

We consider a regression problem with random input/output pair $(X, Y) \in \mathbb{R}^d \times \mathbb{R}$ distributed according to the *joint law* ρ . More specifically, the input X is distributed according to a d -dimensional Gaussian vector $\mathcal{N}(0, I_d)$. For the output, we assume that there exists $\theta^* \in \mathbb{R}^d$ and $\xi \sim \mathcal{N}(0, \sigma^2)$ a Gaussian random variable independent of X of mean zero and variance σ^2 , such that $Y = \langle \theta^*, X \rangle + \xi$. To learn the rule linking inputs to outputs, we take a linear family of predictors $\{f_\theta : x \mapsto \langle \theta, x \rangle, \theta \in \mathbb{R}^d\}$ and aim at minimizing the average *square loss* on the penalty $\ell(\theta, (X, Y)) = \frac{1}{2}(\langle \theta, X \rangle - Y)^2$

$$L(\theta) := \frac{1}{2} \mathbb{E}_{(X, Y) \sim \rho} \left[(\langle \theta, X \rangle - Y)^2 \right] , \quad (1)$$

where ρ is the joint law of (X, Y) .

Proof of (1):

$$\begin{aligned}
L(\theta) &= \frac{1}{2} \mathbb{E}_{(X,Y) \sim \rho} \left[(\langle \theta, X \rangle - Y)^2 \right] \\
&= \frac{1}{2} \mathbb{E}_{(X) \sim N(0, Id)} \mathbb{E}_{(\xi) \sim N(0, \sigma^2)} \left[(\langle \theta, X \rangle - (\langle \theta^*, X \rangle + \xi))^2 \right] \\
&= \frac{1}{2} \mathbb{E}_{(X) \sim N(0, Id)} \mathbb{E}_{(\xi) \sim N(0, \sigma^2)} \left[(\langle \theta - \theta^*, X \rangle - \xi)^2 \right] \\
&= \frac{1}{2} \mathbb{E}_{(X) \sim N(0, Id)} \mathbb{E}_{(\xi) \sim N(0, \sigma^2)} \left[\langle \theta - \theta^*, X \rangle^2 - 2\xi \langle \theta - \theta^*, X \rangle + \xi^2 \right] \\
&= \frac{1}{2} \mathbb{E}_{(X) \sim N(0, Id)} \left[\langle \theta - \theta^*, X \rangle^2 \right] + \frac{1}{2} \mathbb{E}_{(\xi) \sim N(0, \sigma^2)} \left[\xi^2 \right], \text{ since } X \text{ and } \xi \text{ are independent} \\
&= \frac{1}{2} \|\theta - \theta^*\|^2 + \frac{1}{2} \sigma^2, \text{ since } X \text{ is a Gaussian variable, } \mathbb{E}[\langle \theta - \theta^*, X \rangle^2] = \|\theta - \theta^*\|^2 \\
&= \frac{1}{2} (\|\theta - \theta^*\|^2 + \sigma^2)
\end{aligned}$$

Remark 1 *If we knew θ^* , we could use a technique called gradient descent to solve the minimization problem $\min_{\theta} L(\theta)$.*

3.2 The stochastic gradient descent

The stochastic gradient descent (SGD) aims at minimizing a function through unbiased estimates of its gradient. While this method has been developed for a different purpose in the early 50's (Robbins and Monro, 1951), it is remarkable how SGD fits perfectly the modern large-scale machine learning framework (Bottou et al., 2018). Indeed, the SGD iterative procedure corresponds to sample at each time $t \in \mathbb{N}^*$ an independent draw $(x_t, y_t) \sim \rho$ and update the predictor θ with respect to the local gradient calculated on this sample:

$$\theta_{t+1} = \theta_t - \gamma \nabla_{\theta} \ell(\theta_t, (x_t, y_t)), \quad (2)$$

where $\gamma > 0$ is the step size. **This method only rests and having access data $((x_1, y_1), \dots, (x_t, y_t))$, this is the reason why it is so popular in data science!**

Remark 2 $\nabla_{\theta}\ell(\theta_t, (x_t, y_t))$ is an unbiased estimate of $\nabla_{\theta}L(\theta_t)$.¹.

Proof of Remark 2: We know that $\ell(\theta, (X, Y)) = \frac{1}{2}(\langle \theta, X \rangle - Y)^2$. $\nabla_{\theta}\ell(\theta_t, (x_t, y_t)) = x_t(\langle \theta_t, x_t \rangle - y_t)$.

$$\begin{aligned}
\mathbb{E}(\nabla_{\theta}\ell(\theta_t, (x_t, y_t))) &= \mathbb{E}[x_t(\langle \theta_t, x_t \rangle - y_t)] \\
&= \mathbb{E}[x_t(\langle \theta_t, x_t \rangle - y_t + \xi - \xi)] \\
&= \mathbb{E}[x_t(\langle \theta_t - \theta^*, x_t \rangle - \xi)] \\
&= \mathbb{E}[x_t\langle \theta_t - \theta^*, x_t \rangle] - 2\mathbb{E}(x_t \cdot \xi) \\
&= \mathbb{E}[x_t \cdot x_t^T \cdot (\theta_t - \theta^*)], \text{ since } x \text{ is independent of } \xi \\
&= \mathbb{E}[x_t \cdot x_t^T](\theta_t - \theta^*) \\
&= \theta_t - \theta^*
\end{aligned}$$

1. We say that a random variable X is an unbiased estimate of a vector x , if $\mathbb{E}(X) = x$

Also, $L(\theta_t) = \frac{1}{2} \mathbb{E}_{(X,Y) \sim \rho} [(\langle \theta, X \rangle - Y)^2]$.

$$\begin{aligned}
\nabla_{\theta} L(\theta_t) &= \frac{1}{2} \mathbb{E}_{(X,Y) \sim \rho} [2X \cdot (\langle \theta_t, X \rangle - Y)] \\
&= \mathbb{E}_{(X,Y) \sim \rho} [X \cdot (\langle \theta_t, X \rangle - \langle \theta^*, X \rangle - \xi)] \\
&= \mathbb{E}_{(X,Y) \sim \rho} [X \cdot (\langle \theta_t - \theta^*, X \rangle - \xi)] \\
&= \mathbb{E}[X \cdot \langle \theta_t - \theta^*, X \rangle] - \mathbb{E}[X \cdot \xi] \\
&= \mathbb{E}[X \cdot X^T \cdot (\theta_t - \theta^*)] \\
&= \mathbb{E}[X \cdot X^T] \cdot (\theta_t - \theta^*), \text{ since } x \text{ is independent of } \xi \\
&= \theta_t - \theta^*
\end{aligned}$$

Therefore $\mathbb{E}[\nabla_{\theta} \ell(\theta_t, (x_t, y_t))] = \nabla_{\theta} L(\theta_t) = \theta_t - \theta^*$

Lemma 3 *The explicit form of the derivation of Eq. (2) for least-squares is*

$$\theta_{t+1} = \theta_t - \gamma x_t (\langle \theta_t, x_t \rangle - y_t)$$

A convenient way to present these dynamics is to force the apparition of the true gradient and rewrite the rest as a noise (or martingale increment). Indeed,

Lemma 4 *The SGD recursion reads*

$$\theta_{t+1} = \theta_t - \gamma(\theta_t - \theta_*) + \gamma m(\theta_t, (x_t, y_t)) , \quad (3)$$

where $m(\theta_t, (x_t, y_t)) := \mathbb{E}_{\rho} [(\langle \theta_t, X \rangle - Y) X] - (\langle \theta_t, x_t \rangle - y_t) x_t$.

From Lemma 3, we have $\theta_{t+1} = \theta_t - \gamma x_t(\langle \theta_t, x_t \rangle - y_t)$.

The first term in $m(\theta_t, (x_t, y_t))$ is the gradient of $L(\theta_t)$ which is equal to $\theta_t - \theta^*$.

Therefore $m(\theta_t, (x_t, y_t)) = \theta_t - \theta^* - x_t(\langle \theta_t, x_t \rangle - y_t)$.

$$\begin{aligned} \theta_t - \gamma(\theta_t - \theta^*) + \gamma m(\theta_t, (x_t, y_t)) &= \theta_t - \gamma(\theta_t - \theta^*) + \gamma[\theta_t - \theta^* - x_t(\langle \theta_t, x_t \rangle - y_t)] \\ &= \theta_t - \gamma x_t(\langle \theta_t, x_t \rangle - y_t) \\ &= \theta_t - \gamma x_t(\langle \theta_t, x_t \rangle - y_t) \\ &= \theta_{t+1} \end{aligned}$$

Lemma 5 *In equation (3) the variable $m(\theta_t, (x_t, y_t))$ is centered.²*

From Lemma 5 we know that

$$\begin{aligned} m(\theta_t, (x_t, y_t)) &= \mathbb{E}_\rho[(\langle \theta_t, X \rangle - Y)X] - (\langle \theta_t, x_t \rangle - y_t)x_t \\ &:= \mathbb{E}_\rho(Z) - Z, \text{ if we define } Z := X(\langle \theta_t, X \rangle - Y) \end{aligned}$$

Then $\mathbb{E}(m) = \mathbb{E}(\mathbb{E}(Z)) - E(Z) = E(Z) - E(Z) = 0$.

$\Rightarrow m$ is centered.

3.3 Simulations based on SGD

We use the equation of Lemma 4 to write a Python code that simulates the SGD dynamics until time $t = 1000$, with step-size $\gamma = 0.01$, initialization $\theta_0 = \mathbf{0}$, the zero vector, $\theta^* = [0.1, -0.2, 1, 0.5, -0.5]$ and $\sigma = 0.1$.

- (i) Display the test error curve upon time $\|\theta_t - \theta^*\|^2$ for several runs of the dynamics (meaning different data).

2. We say that a random variable X is centered if $E(X) = 0$.

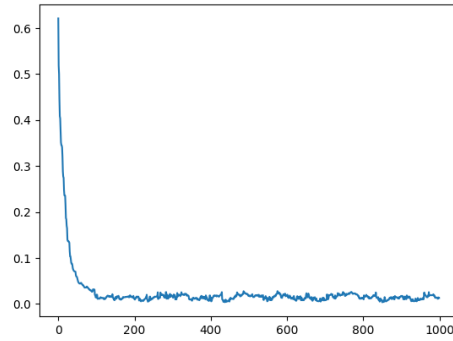


Figure 1: Simulation of test error curve

The test error drops to approximately 0 around $t=200$. This means the SGD dynamics become convergent around $t=200$.

(ii) Display the two first coordinates of $(\theta_t)_t$ as well as the ones of θ^* .

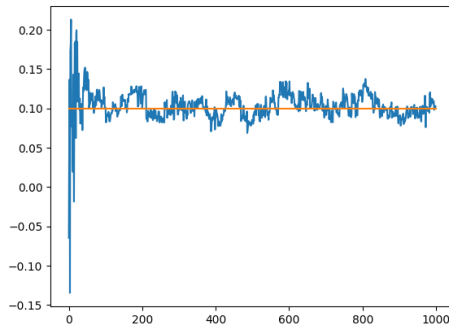


Figure 2: Simulation of θ_1

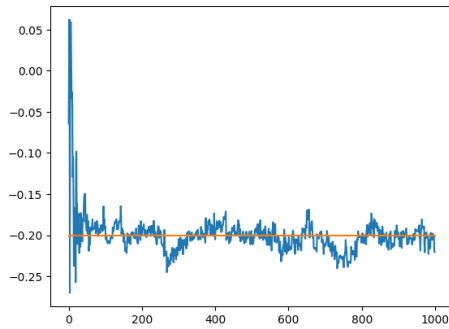


Figure 3: Simulation of θ_2

(iii) Change the variance σ to 1 or even 3: it fluctuates more when σ become higher.

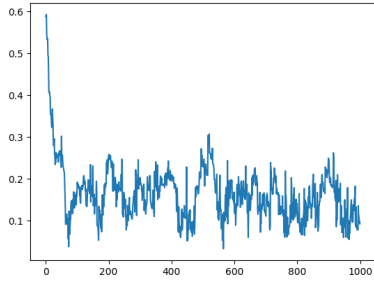


Figure 4: Test Error
when $\sigma = 1$

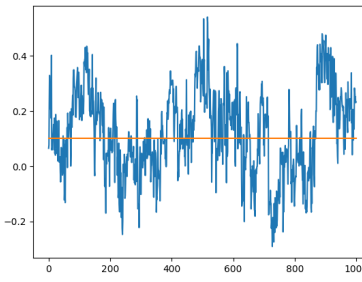


Figure 5: Simulation of θ_1
when $\sigma = 1$

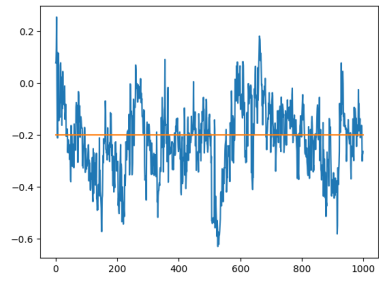


Figure 6: Simulation of θ_2
when $\sigma = 1$

(iv) When the step size is 10 times bigger, it explodes since the gradient changes too far each time.

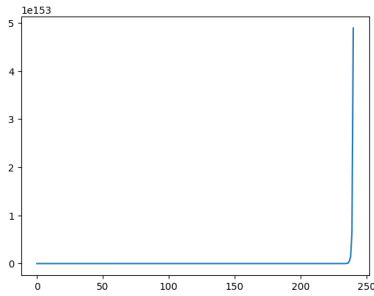


Figure 7: Test Error when
step-size bigger

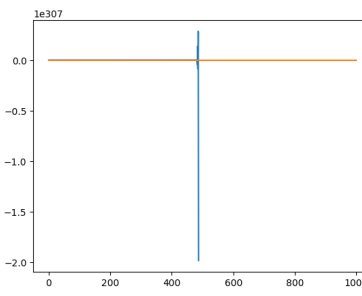


Figure 8: θ_1 when
step-size bigger

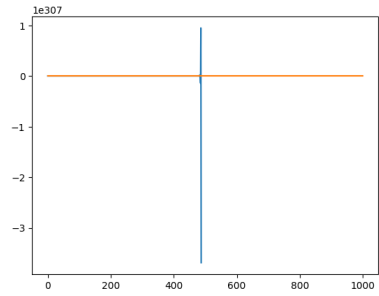


Figure 9: θ_2 when
step-size bigger

When the step size is 10 times smaller, it moves slowly but we can see that it becomes more accurate after convergence.

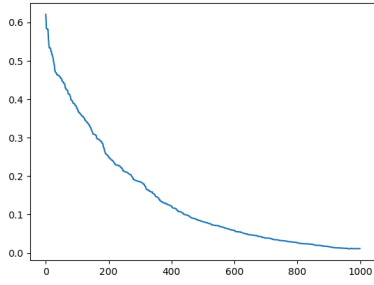


Figure 10: Test Error when
step-size smaller

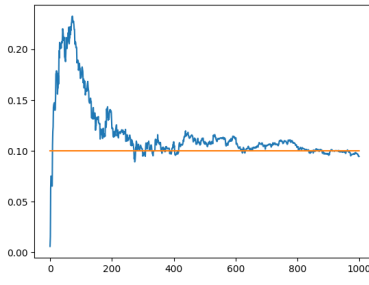


Figure 11: θ_1 when
step-size smaller

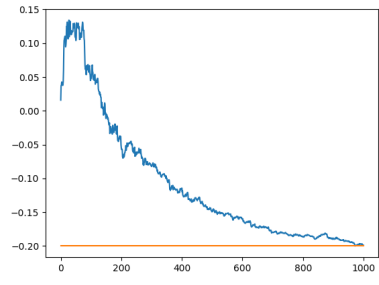


Figure 12: θ_2 when
step-size smaller

(v) Change the step size to make it depend on the iterations: $\gamma = 0.1/t$. The simulation balanced between accuracy and velocity.

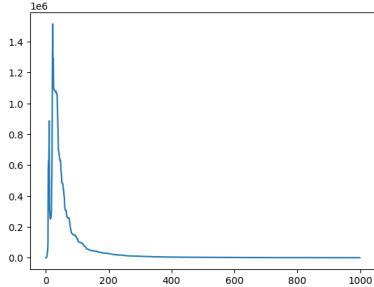


Figure 13: Test Error:
Improved

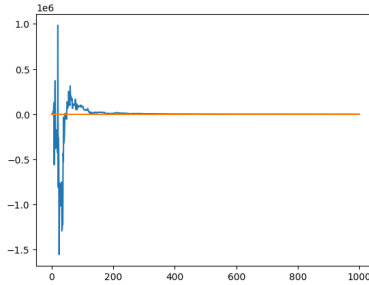


Figure 14: θ_1 : Improved

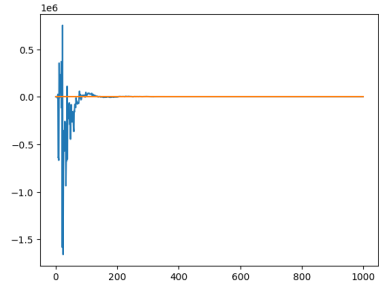


Figure 15: θ_2 : Improved

4 Continuous model of SGD & SDE

In this section, we give stochastic differential equations (SDE) models for SGD. For the first time, we provide general necessary conditions to model well SGD. We instantiate them more precisely in a second time.

4.1 The requirement of a SDE model: connect SDE with SGD

As said above, the decomposition of the SGD recursion as in equations (3) is generic feature of the stochastic descent (Benaïm, 2006), as it has led to the celebrated *ODE method* (Harold et al., 1997) to study stochastic approximation of this type. Going further, this decomposition between a *drift* ∇L and *local martingale term* $m(\cdot, (x, y))$ is reminiscent of the decomposition occurring for Itô processes, i.e. solution to an SDE of the type

$$d\theta_t = b(t, \theta_t)dt + \sigma(t, \theta_t)dB_t, \quad (4)$$

where $(B_t)_{t \geq 0}$ is a Brownian motion of \mathbb{R}^d . These types of models have been largely studied in the literature (Khasminskii, 2011), as, beyond their large modeling abilities, they offer useful tools, e.g. Itô calculus, when it comes to mathematical analysis. One of the aims of this article is to link the SGD dynamics to processes that are exemplary in the SDE literature. In order to establish this link, we first have to answer the following question

What SDE model fits well with the SGD dynamics?

This question has received a lot of attention in the last decade, and a good principle to answer this is to turn to *stochastic modified equations* (Li et al., 2019). This is a natural way to build models of SDE since they are consistent in the infinitesimal step-size limit with SGD. In order to build such a model, there are two requirements

Remark 6 *Bases on (Li et al., 2019, Section 3.1), we conclude that*

- (i) *The drift term $b(t, \theta_t)$ should match $[-\nabla L(\theta_t)]$.*

(ii) The noise factor σ should have the same covariance as $[\gamma \nabla L(\theta_t)]$, i.e.

$$[\gamma \mathbb{E}[(\nabla L(\theta_t) - \nabla l(\theta_t))(\nabla L(\theta_t) - \nabla l(\theta_t))^T | \theta_t].]$$

Proof of Remark 6: From equation (1), we know that

$$L(\theta) = \frac{1}{2} E_{(X,Y) \sim \rho} ((\langle \theta, x \rangle - Y)^2)$$

Its gradient has already been computed before in Lemma 3, which is:

$$\nabla_{\theta} L(\theta_t) = (\theta_t - \theta^*) Id = \begin{pmatrix} \theta_t - \theta^* \\ \vdots \\ \theta_t - \theta^* \end{pmatrix} \in \mathbb{R}^d,$$

From equation(3) in the tutorial we have:

$$\begin{aligned} \theta_{t+1} - \theta_t &= -\gamma(\theta_t - \theta^*) + \gamma m(\theta_t, (x_t, y_t)) \\ &= -\gamma \nabla L(\theta_t) + \gamma(\mathbb{E}_{\rho}((\langle \theta_t, x_t \rangle - y_t)x_t) - (\langle \theta_t, x_t \rangle - y_t)x_t) \\ &= -\gamma \nabla L(\theta_t) + \gamma(\nabla L(\theta_t) - \nabla l(\theta_t)) \end{aligned}$$

where $l(\theta) = \frac{1}{2}(\langle \theta_t, x_t \rangle - y_t)^2$.

Considering equation (4) in the tutorial, which is:

$$d\theta_t = b(t, \theta_t)dt + \sigma(t, \theta_t)dB_t$$

If we apply the Euler-Maruyama discretization with step-size γ , approximating $X_{k\gamma}$ by \hat{X}_k , we obtain the following discrete iteration:

$$\hat{\theta}_{t+1} - \hat{\theta}_t = \gamma b(t, \hat{\theta}_t) + \sqrt{\gamma} \sigma(t, \hat{\theta}_t) Z_k \quad (5)$$

where $Z_k := B_{(k+1)\gamma} - B_{k\gamma}$ are d-dimensional i.i.d standard normal random variables.

Matching equation (5) with the previous deduction of equation (3), we know that:

- (i) The drift term $b(t, \theta_t)$ should match $-\nabla L(\theta_t)$.
- (ii) The noise factor σ should have the same covariance as $\gamma \nabla L(\theta_t)$, i.e. $\gamma \mathbb{E}[(\nabla L(\theta_t) - \nabla l(\theta_t))(\nabla L(\theta_t) - \nabla l(\theta_t))^T | \theta_t]$. This means $\sigma(t, \theta_t) = \sqrt{\gamma \Sigma(\theta)}$.

Then $d\theta_t = -\nabla L(\theta_t)dt + \sqrt{\gamma \Sigma(\theta)}dB_t$.

Besides technical assumptions, these are the two requirements presented in Li et al. (2019, Theorem 3) to show that the SDE model is *consistent* in the small step-size limit with the SGD recursion.

4.2 Explicit form of the SDE models

Let us first write explicitly the multiplicative noise factor $\sigma(t, \theta_t)$ in the population case. Then, we derive the expression of the SDE models that we analyze later. In the population case, the calculation has already been made in Ali et al. (2020) and, defining $r_X(\theta) = \langle \theta_t, X \rangle - Y \in \mathbb{R}$, the residual random variable, then we have that

$$\sigma(t, \theta_t) \sigma(t, \theta_t)^\top = \gamma \left(\mathbb{E}_\rho [r_X(\theta_t)^2 X X^\top] - \mathbb{E}_\rho [r_X(\theta_t) X] \mathbb{E}_\rho [r_X(\theta_t) X]^\top \right).$$

Let us simplify the covariance and assume that for all $\theta \in \mathbb{R}^d$,

$$\sigma(\theta) := \sqrt{\gamma\sigma^2}I_d. \quad (6)$$

We hence study the SDE.

Lemma 7 *The derivation of the SDE with the simplification given by Eq. (6) is:*

$$d\theta_t = -\nabla L(\theta_t)dt + \sqrt{\gamma\Sigma(\theta)}dB_t = -(\theta_t - \theta^*)Idt + (\sqrt{\gamma\sigma^2}Id)dB_t$$

Lemma 8 *Each coordinate of θ_t satisfy an Ornstein-Uhlenbeck with parameters that can be specified.*

Proof of Lemma 8: In the SDE model, we have:

$$d\theta_t = -(\theta_t - \theta^*)Idt + (\sqrt{\gamma\sigma^2}Id)dB_t \quad (7)$$

To show that each coordinate of θ_t satisfies an Ornstein-Uhlenbeck process, we need to match each parameter in this SDE with the standard form of an Ornstein-Uhlenbeck process:

$$d\theta_t = \kappa(\theta - \theta_t)dt + \sigma dW_t \quad (8)$$

Matching equation (7) and equation (8), we have:

- $\kappa = 1$.

- $\theta = \theta^*$, the long-term mean of the process matches θ^* .
- $\sigma = \sqrt{\gamma\sigma^2}$, the volatility term matches the noise factor.

Remark 9

The mean of the process is $\mathbb{E}(\theta_t) = \theta^ + (\mathbb{E}(\theta_0) - \theta^*)e^{-t}$.* (9)

The variance of the process is $\text{Var}(X_t) = \frac{\gamma\sigma^2}{2}(1 - e^{-2t})$ (10)

Proof of Remark 9

$$d\mathbb{E}(\theta_t) = \kappa(\theta - \mathbb{E}(\theta_t))dt$$

This is an ODE in $\mathbb{E}(\theta_t)$ which can be solved to give:

$$\mathbb{E}(\theta_t) = \theta^* + (\mathbb{E}(\theta_0) - \theta^*)e^{-t}$$

The variance of the process can be found by solving another differential equation.

The variance is given by:

$$\text{Var}(\theta_t) = \mathbb{E}(\theta_t^2) - \mathbb{E}(\theta_t)^2$$

To find $\mathbb{E}(\theta_t^2)$, we use Ito's formula and the fact that $dW_t^2 = dt$ to set up an ODE for it:

$$\frac{d}{dt}\mathbb{E}(\theta_t^2) = 2\kappa\mathbb{E}(\theta_t) - 2\kappa\mathbb{E}(\theta_t^2) + \sigma^2$$

Solving the ODE and plugging in the corresponding parameters gives:

$$\text{Var}(X_t) = \frac{\gamma\sigma^2}{2}(1 - e^{-2t})$$

Remark 10 *The process converges to Gaussian Distribution with mean θ^* and variance $\frac{\gamma\sigma^2}{2}$, since the mean reversion term represents a force that pulls the process back towards the mean θ^* when θ_t deviates from it.*

4.3 Simulation for the OU process through the Euler-Maruyama method

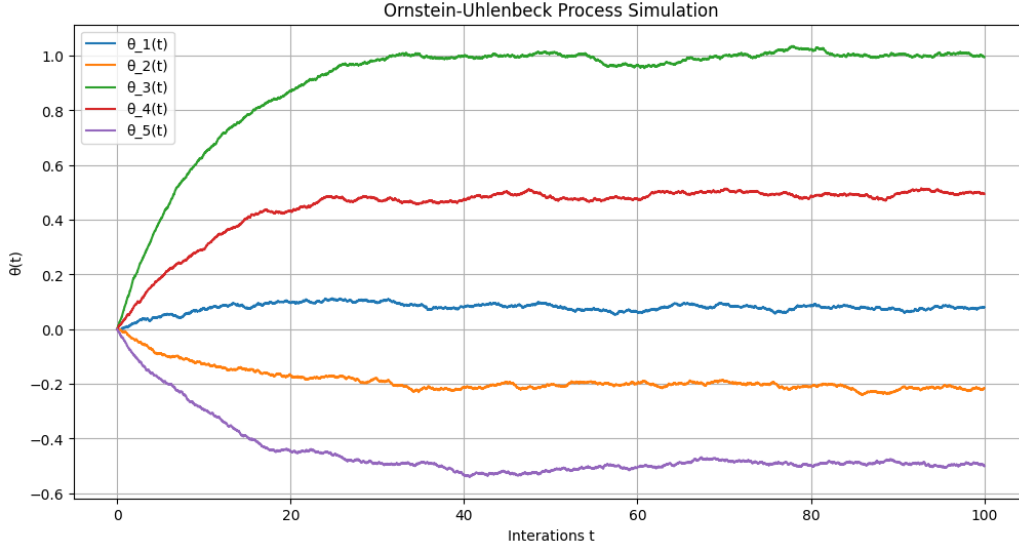


Figure 16: Simulation of OU process

5 Score-based Generative Model through SDE

Score-based generative models establish a new paradigm for the unsupervised learning of probability distributions by directly calculating the data gradient, $\nabla_x \log p_d(x)$, to generate new samples. By characterizing the probability distribution through score functions—gradients of the log probability—it is possible to employ SDE for sampling by following the gradient flow of p_d . This section explains the connections between score-based generative models and SDE, which are built on the basis of two cutting-edge diffusion models: *Score Matching Langevin Dynamics (SMLD)* and *Denoising Diffusion Probability Models (DDPM)*. In particular, the following points are explored:

1. How the estimated score approximates the gradient $\nabla_x \log p_d(x)$, which facilitates the generation of new samples from p_d , forming the basis of the SMLD model.
2. How the noise scheduling and diffusion process enables the DDPM model to iteratively generate high-quality samples from the learned data distribution.
3. How the DDPM and SMLD models are linked through stochastic differential equations (SDE), with both models using score-based generative techniques to reverse the noise addition process.

In the following sections, detailed deductions and explanations will be provided based on the seminal works of Chao et al. (2022) on SMLD, Ho et al. (2020) on DDPM, and Song et al. (2020) on Score-based generative modeling with SDE.

5.1 Score Matching Langevin Dynamics (SMLD)

5.1.1 PROBLEM SETTING

Consider a set of n data samples x_1, \dots, x_n in a set X drawn from an unknown true distribution $p_d(x)$. Although we do not know the actual distribution, we aim to model it using an estimated distribution $p_m(x; \theta)$, where θ is a parameter (or a vector of parameters). Our objective is to determine the value of θ such that $p_m(x; \theta)$ closely resembles $p_d(x)$.

In a maximum likelihood estimation (MLE) approach, the parameter θ is chosen to maximize the (log) likelihood of the observed data:

$$\hat{\theta}_{MLE} = \arg \max_{\theta} \log p_m(x; \theta), \quad (11)$$

However, this is often unfeasible due to an intractable normalizing constant. Specifically, $p_m(x; \theta)$ can be expressed using an unnormalized density $\tilde{p}(x; \theta)$ and a normalizing constant:

$$p_m(x; \theta) = \frac{\tilde{p}(x; \theta)}{Z_{\theta}}, \quad Z_{\theta} = \int_X \tilde{p}(x; \theta) dx$$

The normalizing constant Z_{θ} ensures the integral of $\frac{\tilde{p}(x; \theta)}{Z_{\theta}}$ equals one, enabling convergence. Yet, evaluating the integral Z_{θ} is frequently impractical. To tackle this challenge, we introduce the *Score Matching* technique, which approximates the integral without explicitly dealing with the normalizing constants.

5.1.2 DEDUCTION OF (DENOISING) SCORE MATCHING

The motivation of score matching is to identify a parameter θ such that the gradient of the model's log-likelihood approximates the gradient of the true data distribution's log-likelihood. Rather than directly maximizing the likelihood function (Eq. 11), we

can focus on working with the unnormalized density $\tilde{p}(x; \theta)$, sidestepping the need for the normalized density $p(x; \theta)$ which is often unknown to us.

The score function represents the gradient of the log-likelihood with respect to the data x :

$$s_\theta(x) = \nabla \log p_m(x; \theta) \tag{12}$$

The primary step is to eliminate the normalizing constant. The score function can be expanded as:

$$\nabla_x \log p_m(x; \theta) = \nabla_x \log \tilde{p}(x; \theta) - \underbrace{\nabla_x Z_\theta}_0$$

where the last term vanishes since the normalizing constant does not depend on x .

While the score function does not directly represent the original model distribution, it still contains valuable information about the distribution's gradient structure. Specifically, it informs us of the distribution's gradient for a particular value of θ , offering insight into the function's first-order structure.

The goal of score matching is to align the score function of the model distribution with the score function of the data distribution, $\nabla \log p_d(x)$. It aims to minimize the *Fisher divergence* between these two score functions. The Fisher divergence quantifies the discrepancy between the gradient of the log-probability of the data distribution and that of the model distribution. Minimizing the Fisher divergence ensures that the model's score approximates the data distribution's score. For simplicity, we first

consider the 1D case, which can be further extended to multidimensional scenarios.

$$\hat{\theta}_{SM} = \arg \min_{\theta} D_F(p_d, p_m) \quad (13)$$

$$= \arg \min_{\theta} \frac{1}{2} \int p_d(x) [\|\nabla_x \log p_d(x) - \nabla_x \log p_m(x; \theta)\|_2^2] \quad (14)$$

$$= \arg \min_{\theta} \frac{1}{2} \mathbb{E}_{p_d} [\|\nabla_x \log p_d(x) - \nabla_x \log p_m(x; \theta)\|_2^2]. \quad (15)$$

$$= \arg \min_{\theta} \mathbb{E}_{p_d} [\underbrace{\frac{1}{2} (\nabla_x \log p_d(x))^2}_{\text{constant}} - \nabla_x \log p_m(x; \theta) \nabla_x \log p_d(x) + \frac{1}{2} (\nabla_x \log p_m(x; \theta))^2]. \quad (16)$$

The first term can be ignored as it only contains the data density and is independent of θ . The third term is easy to approximate using a finite sample of data since it doesn't depend on the data density. Thus, the focus shifts to the second term. To expand the expectation and gradient:

$$\mathbb{E}_{p_d} [-\nabla_x \log p_m(x; \theta) \nabla_x \log p_d(x)] = - \int_{-\infty}^{\infty} \nabla_x \log p_m(x; \theta) \nabla_x \log p_d(x) p_d(x) dx \quad (17)$$

$$= - \int_{-\infty}^{\infty} \nabla_x \log p_m(x; \theta) \frac{\nabla_x p_d(x)}{p_d(x)} p_d(x) dx \quad (18)$$

$$= - \int_{-\infty}^{\infty} \nabla_x \log p_m(x; \theta) \nabla_x p_d(x) dx \quad (19)$$

Using integration by parts:

$$\begin{aligned} \int_a^b u(x) v'(x) dx &= [u(x) v(x)]_a^b - \int_a^b u'(x) v(x) dx \\ &= u(b) v(b) - u(a) v(a) - \int_a^b u'(x) v(x) dx \end{aligned}$$

With $u(x) = \nabla_x \log p_m(x; \theta)$ and $v'(x) = \nabla_x p_d(x)$, the right hand side of Eq.(19) become:

$$\underbrace{-\lim_{b \rightarrow \infty} \nabla_x \log p_m(b; \theta) p_d(b)}_0 + \underbrace{\lim_{a \rightarrow -\infty} \nabla_x \log p_m(a; \theta) p_d(a)}_0 + \int_{-\infty}^{\infty} \nabla_x^2 p_m(x; \theta) p_d(x) dx$$

Hyvärinen and Dayan (2005) assume (as a regularity condition in their Theorem 1) that for any θ :

$$p_d(x) \nabla_x \log p_m(x; \theta) \rightarrow 0, \text{ as } \|x\|_2 \rightarrow \infty$$

This assumption allows us to omit the first two terms. The expectation of the second term then approximates:

$$\mathbb{E}[\nabla_x^2 \log p_m(x)]$$

Combining all the terms, we derive:

$$D_F(p_d, p_m) = \mathbb{E}_{p_d}[\nabla_x^2 \log p_m(x) + \frac{1}{2} \mathbb{E}_{p_d}(\nabla_x \log p_m(x))^2] + \text{const} \quad (20)$$

$$= \mathbb{E}_{p_d}[\nabla_x^2 \log p_m(x) + \frac{1}{2} (\nabla_x \log p_m(x))^2] + \text{const} \quad (21)$$

In the multidimensional case:

$$D_F(p_d, p_m) \propto L(\theta) \triangleq \mathbb{E}_{p_d}[\text{tr}(\nabla_x^2 \log p_m(x; \theta)) + \frac{1}{2} \|\nabla_x \log p_m(x; \theta)\|_2^2] \quad (22)$$

This rewrites the objective purely in terms of $\log p_m(x; \theta)$, which doesn't depend on the normalizing constant or the data distribution. Using the score function (12) to

simplify the expression, we have the loss function:

$$L(\theta) = \mathbb{E}_{p_m(x)}[tr(\nabla_x s_\theta(x)) + \frac{1}{2}\|s_\theta(x)\|^2] \quad (23)$$

However, computing $tr(\nabla_x s_\theta(x))$ makes score matching challenging for deep networks and high-dimensional data. A popular approach to large-scale score matching is called *Denoising Score Matching*, which completely avoids computing $tr(\nabla_x s_\theta(x))$. This method first perturbs the data point x with a predefined noise distribution $q_\theta(\tilde{x}|x)$, then uses score matching to estimate the score of the perturbed data distribution $q_\theta(\tilde{x}) \triangleq \int q_\theta(\tilde{x}|x)p_d(x)dx$. The objective is equivalent to:

$$-\frac{1}{2}\mathbb{E}_{q_\theta(\tilde{x}|x)p_d(x)}[\|s_\theta(\tilde{x}) - \nabla_{\tilde{x}} \log q_\theta(\tilde{x}|x)\|^2] \quad (24)$$

The optimal score network (denoted as $s_\theta^*(x)$) that minimizes Eq(24) satisfies $s_\theta^*(x) = \nabla_x \log q_\theta(x)$ almost surely. However, $s_\theta^*(x) = \nabla_x \log q_\theta(x) \approx \nabla_x \log p_d(x)$ holds true only if the noise is sufficiently small such that $q_\theta(x) \approx p_{\text{data}}(x)$.

5.1.3 SCORE MATCHING LANGEVIN DYNAMICS (SMLD)

After training neural networks to learn the distribution of the score function, the next question is how to use the score function to generate samples from this distribution. The method employed for this task is known as *Langevin Dynamics*:

$$x_{i+1} = x_i + \epsilon \nabla_x \log p_d(x) + \sqrt{2\epsilon} z_i, z_i \sim \mathcal{N}(\mathbf{0}, \mathbf{I}), i = 0, 1, \dots, K \quad (25)$$

This sampling process is iterative, and ϵ is sufficiently small. The initial value x_0 is randomly initialized and updated using the iterative formula above. When the number of iterations K is large, x converges to a sample from the desired distribution.

To sum up, in this subsection, we have explored Score Matching Langevin Dynamics (SMLD) as a method to generate samples from the distribution of the score function. We discussed how neural networks learn the distribution of the score function and detailed how Langevin Dynamics can be used to iteratively sample from this distribution. By leveraging an iterative process involving gradients of the log-likelihood and small step sizes, SMLD provides a robust approach to sampling high-dimensional distributions.

5.2 Denoising diffusion probabilistic model (DDPM)

Diffusion models are a class of generative models that simulate a step-by-step transformation of data through a diffusion process, gradually adding noise to the data in successive steps. These models aim to reverse this process, reconstructing the original data by removing the noise, thereby learning a mapping from a simple noise distribution back to the target data distribution. This approach relies on a Markov chain framework, where each step conditions on the previous one, making it possible to model complex distributions by approximating the data distribution through a sequence of Gaussian transformations. The primary objective of diffusion models is to learn this reverse diffusion process, enabling the generation of new samples from the learned distribution by progressively reducing noise, which ultimately yields realistic data points.

Denoising Diffusion Probabilistic Models (DDPM) are a specific type of diffusion model that utilizes a noise-conditioned score network to approximate the reverse diffusion process. By training a neural network to predict the noise added at each step, DDPMs enable effective sampling from complex data distributions. In the subsequent sections, we will first introduce the logic of fundamental diffusion models, and

then explain the structure of DDPMs, their sampling methods, and the mathematical principles that underpin them highly based on the contribution of Ho et al. (2020).

5.2.1 DIFFUSION MODELS

Diffusion models can be understood as a development from Markov Hierarchical Variational Autoencoders (MHVAE), which themselves are expansions of traditional VAEs. Within this framework, latent variable models are established by the expression $p_\theta(x_0) := \int p_\theta(x_{0:T}) dx_{1:T}$, where x_1, \dots, x_T represent latent variables corresponding in size to the observed data $x_0 \sim q(x_0)$. The comprehensive distribution $p_\theta(x_{0:T})$, referred to as the reverse process, is characterized as a Markov chain with Gaussian transitions that are learned, beginning from $p(x_T) = \mathcal{N}(x_T; \mathbf{0}, \mathbf{I})$:

$$p_\theta(x_{0:T}) := p(x_T) \prod_{t=1}^T p_\theta(x_{t-1}|x_t), \quad p_\theta(x_{t-1}|x_t) := \mathcal{N}(x_{t-1}; \mu_\theta(x_t, t), \Sigma_\theta(x_t, t))$$

The estimated posterior $q(x_{1:T}|x_0)$, identified as the forward or diffusion process, manifests as a consistent Markov chain that incrementally incorporates Gaussian noise into the data following a prescribed variance sequence β_1, \dots, β_T :

$$q(x_{1:T}|x_0) := \prod_{t=1}^T q(x_t|x_{t-1}), \quad q(x_t|x_{t-1}) := \mathcal{N}(x_t; \sqrt{1 - \beta_t}x_{t-1}, \beta_t\mathbf{I}) \quad (26)$$

This forward process serves as the encoder, progressively embedding observable samples x_0 into the final state x_T through the introduction of Gaussian noise, thus defining $q(\cdot)$. Conversely, the reverse process acts as a decoder, where the parameters are adapted through learning. Each transitional step t is solely reliant on the immediately preceding step $t - 1$, thus forming a sequential Markov chain.

Training is achieved by optimizing the standard variational bound on negative log-

likelihood:

$$\mathbb{E}[-\log p_\theta(x_0)] \leq \mathbb{E}_q[-\log \frac{p_\theta(x_{0:T})}{q(x_{1:T}|x_0)}] = \mathbb{E}_q[-\log p(x_T) - \sum_{t \geq 1} \log \frac{p_\theta(x_{t-1}|x_t)}{q(x_t|x_{t-1})}] =: L \quad (27)$$

Proof of (27): By Jensen's inequality,

$$\begin{aligned} \log p(x_0) &= \log \int p(x_{0:T}) dx_1 \\ &= \log \int \frac{p(x_{0:T}) q(x_{1:T}|x_0)}{q(x_{1:T}|x_0)} dx_{1:T} \\ &= \log \mathbb{E}_q(x_{1:T}|x_0) \left[\frac{p(x_{0:T})}{q(x_{1:T}|x_0)} \right] \\ &\geq \mathbb{E}_q(x_{1:T}|x_0) \left[\log \frac{p(x_{0:T})}{q(x_{1:T}|x_0)} \right] \\ &= \mathbb{E}_q(x_{1:T}|x_0) \left[\log \frac{p(x_T) \prod_{t=1}^T p_\theta(x_{t-1}|x_t)}{\prod_{t=1}^T q(x_t|x_{t-1})} \right] \end{aligned}$$

The variances β_t of the forward process can either be learned through parameterization or maintained as constant hyperparameters. The reverse process's expressiveness is partially derived from the Gaussian conditionals chosen in $p_\theta(x_{t-1}|x_t)$, due to the fact that both processes share the same functional form when β_t is small. Using the notations $\alpha_t := 1 - \beta_t$ and $\bar{\alpha}_t := \prod_{s=1}^t \alpha_s$, we have:

$$q(x_t|x_0) = \mathcal{N}(x_t; \sqrt{\bar{\alpha}_t}x_0, (1 - \bar{\alpha}_t)\mathbf{I}) \quad (28)$$

Proof of (28):

$$\begin{aligned}
x_t &= \sqrt{\alpha_t}x_{t-1} + \sqrt{1 - \alpha_t}\epsilon_t, \quad \text{where } \epsilon_t \text{ belongs to } \mathcal{N}(\mathbf{0}, \mathbf{I}) \\
&= \sqrt{\alpha_t}(\sqrt{\alpha_{t-1}}x_{t-2} + \sqrt{1 - \alpha_{t-1}}\epsilon_{t-1}) + \sqrt{1 - \alpha_t}\epsilon_t \\
&= \sqrt{\alpha_t\alpha_{t-1}}x_{t-2} + \underbrace{\sqrt{\alpha_t - \alpha_t\alpha_{t-1}}\epsilon_{t-1} + \sqrt{1 - \alpha_t}\epsilon_t}_{\text{sum up two Gaussian distribution with mean 0}} \\
&= \sqrt{\alpha_t\alpha_{t-1}}x_{t-2} + \underbrace{\sqrt{(\sqrt{\alpha_t - \alpha_t\alpha_{t-1}})^2 + (\sqrt{1 - \alpha_t})^2}}_{\text{express with a new Gaussian distribution}}\epsilon \\
&= \sqrt{\alpha_t\alpha_{t-1}}x_{t-2} + \sqrt{1 - \alpha_t\alpha_{t-1}}\epsilon \\
&= \dots \\
&= \sqrt{\prod_{i=1}^t \alpha_i}x_0 + \sqrt{1 - \prod_{i=1}^t \alpha_i}\epsilon \\
&= \sqrt{\bar{\alpha}_t}x_0 + \sqrt{1 - \bar{\alpha}_t}\epsilon, \text{ where } \bar{\alpha}_t = \prod_{i=1}^t \alpha_i, \epsilon \sim \mathcal{N}(\mathbf{0}, \mathbf{I})
\end{aligned}$$

Here we define that the variance of $q(x_t|x_{t-1})$ is independent with x_{t-1} , which is equal to $\beta_t\mathbf{I}$, where $0 < \beta_1 < \dots < \beta_T < 1$. This makes sense since at the beginning, the variance is small, and we add less noise to make the diffusion slow; when the variance becomes bigger with the increase of t , our noise increases, and diffusion speed increases as well.

We also define that the mean of $q(x_t|x_{t-1})$ is linearly related to x_{t-1} . According to the properties of linear Gaussian, we can regard the process as adding random Gaussian noise to each previous step.

Notation clarification:

- **Why we set β_t as a hyperparameter:** Forward process is supposed to let (1) x_t approximate to normal Gaussian distribution $\mathcal{N}(\mathbf{0}, \mathbf{I})$, (2) x_t only

depends on x_{t-1} , and (3) x_t and x_{t-1} are linear transformations of Gaussian random variables, i.e. the mean μ_{x_t} and x_{t-1} are linearly proportionate by some parameter β .

- **Why we have a square root:** When $\mu_{x_t} \rightarrow 0$, we also need to satisfy that the variance goes to identity. Using a parameter to control the changes of mean and variance simultaneously can make the diffusion process more stable. α is defined on variance, so we add a square root.
- **What is the intercept term:** $\sqrt{1 - \alpha_t}\epsilon$

By optimizing the terms of L with SGD, effective training is therefore possible for us.

We can further improve the variance reduction by rewriting L in Eq.(27) as:

$$\mathbb{E}_q \left[\underbrace{D_{KL}(q(x_T|x_0) \parallel p(X_T))}_{L_T} + \sum_{t>1} \underbrace{D_{KL}(q(x_{t-1}|x_t, x_0) \parallel p_\theta(x_{t-1}|x_T))}_{L_{t-1}} - \underbrace{\log p_\theta(x_0|x_1)}_{L_0} \right] \quad (29)$$

Proof of (29):

$$\begin{aligned} L &= \mathbb{E}_q \left[-\log \frac{p_\theta(x_{0:T})}{q(x_{1:T}|x_0)} \right] \\ &= \mathbb{E}_q \left[-\log p(x_T) - \sum_{t \geq 1} \log \frac{p_\theta(x_{t-1}|x_t)}{q(x_t|x_{t-1})} \right], \quad (\text{by (27)}) \\ &= \mathbb{E}_q \left[-\log p(x_T) - \sum_{t > 1} \log \frac{p_\theta(x_{t-1}|x_t)}{q(x_t|x_{t-1})} - \log \frac{p_\theta(x_0|x_1)}{q(x_1|x_0)} \right], \quad (\text{drag out } t = 1 \text{ term}) \end{aligned}$$

By Conditional Independence, since the process is a Markov chain, we have $q(x_t|x_{t-1}) = q(x_t|x_{t-1}, x_0)$. And by Naive Bayesian Inference, we know that:

$$q(x_t|x_{t-1}, x_0) = q(x_{t-1}|x_t, x_0) \frac{q(x_{t-1}|x_0)}{q(x_t|x_0)} \quad (30)$$

Plugging equation (30) into our loss function, we continue to get:

$$\begin{aligned} L &= \mathbb{E}_q[-\log p(x_T) - \sum_{t>1} \log \frac{p_\theta(x_{t-1}|x_t)}{q(x_{t-1}|x_t, x_0)} \cdot \frac{q(x_{t-1}|x_0)}{q(x_t|x_0)} - \log \frac{p_\theta(x_0|x_1)}{q(x_1|x_0)}] \\ &= \mathbb{E}_q[-\log \frac{p(x_T)}{q(x_T|x_0)} - \sum_{t>1} \log \frac{p_\theta(x_{t-1}|x_t)}{q(x_{t-1}|x_t, x_0)} - \log p_\theta(x_0|x_1)] \end{aligned}$$

Rewrite the second term using KL divergence for discrete random variables, which is:

$$D_{KL}(P \parallel Q) = \sum_x P(x) \log \left(\frac{P(x)}{Q(x)} \right) = \mathbb{E}_P[\log \frac{P(x)}{Q(x)}] \quad (31)$$

we have:

$$L = \mathbb{E}_q[D_{KL}(q(x_T|x_0) \parallel p(x_T)) + \sum_{t>1} D_{KL}(q(x_{t-1}|x_t, x_0) \parallel p_\theta(x_{t-1}|x_t)) - \log p_\theta(x_0|x_1)]$$

Equation (29) uses KL divergence to directly compare $p_\theta(x_{t-1}|x_t)$ against forward process posteriors, which are tractable when conditioned on x_0 :

$$q(x_{t-1}|x_t, x_0) = \mathcal{N}(\mathbf{x}_{t-1}; \tilde{\mu}_t(\mathbf{x}_t, \mathbf{x}_0), \tilde{\beta}_t \mathbf{I}), \quad (32)$$

$$\text{where } \tilde{\mu}_t(\mathbf{x}_t, \mathbf{x}_0) := \frac{\sqrt{\bar{\alpha}_{t-1}}\beta_t}{1 - \bar{\alpha}_t}x_0 + \frac{\sqrt{\bar{\alpha}_t}(1 - \bar{\alpha}_{t-1})}{1 - \bar{\alpha}_t}x_t \quad \text{and} \quad \tilde{\beta}_t := \frac{1 - \bar{\alpha}_{t-1}}{1 - \bar{\alpha}_t}\beta_t \quad (33)$$

Proof of (32) and (33): We already showed that the diffusion process is a Markov Chain, so according to Conditional Independence, we can rewrite (??):

$$\begin{aligned}
q(x_t|x_{t-1}, x_0) &= q(x_t|x_{t-1}) \\
&= \mathcal{N}(x_t; \sqrt{1 - \beta_t}x_{t-1}, \beta_t I) \\
&= \mathcal{N}(x_t; \sqrt{\alpha_t}x_{t-1}, \beta_t I)
\end{aligned}$$

Also, according to (28):

$$q(x_{t-1}|x_0) = \mathcal{N}(x_{t-1}; \sqrt{\bar{\alpha}_{t-1}}x_0, (1 - \bar{\alpha}_{t-1})I)$$

So $q(x_t|x_0) = \mathcal{N}(x_t; \sqrt{\bar{\alpha}_t}x_0, (1 - \bar{\alpha}_t)I)$. Therefore we have:

$$\begin{aligned}
q(x_{t-1}|x_t, x_0) &= q(x_t|x_{t-1}, x_0) \frac{q(x_{t-1}|x_0)}{q(x_t|x_0)} \\
&\propto \exp\left(-\frac{1}{2}\left(\frac{(x_t - \sqrt{\alpha_t}x_{t-1})^2}{\beta_t} + \frac{(x_{t-1} - \sqrt{\bar{\alpha}_{t-1}}x_0)^2}{1 - \bar{\alpha}_{t-1}} - \frac{(x_t - \sqrt{\bar{\alpha}_t}x_0)^2}{1 - \bar{\alpha}_t}\right)\right) \\
&= \exp\left(-\frac{1}{2}\left(\frac{x_t^2 - 2\sqrt{\alpha_t}x_{t-1}x_t + \alpha_t x_{t-1}^2}{\beta_t} + \frac{x_{t-1}^2 - 2\sqrt{\bar{\alpha}_{t-1}}x_0x_{t-1} + \bar{\alpha}_{t-1}x_0^2}{1 - \bar{\alpha}_{t-1}}\right.\right. \\
&\quad \left.\left. - \frac{x_t^2 - 2\sqrt{\bar{\alpha}_t}x_0x_t + \bar{\alpha}_t x_0^2}{1 - \bar{\alpha}_t}\right)\right) \\
&\quad \quad \quad := C(x_t, x_0), \text{ not based on } x_{t-1} \\
&= \exp\left(\frac{1}{2}\left(\underbrace{\left(\frac{\alpha_t}{\beta_t} + \frac{1}{1 - \bar{\alpha}_{t-1}}\right)}_{:=A} x_{t-1}^2 - \underbrace{\left(\frac{2\sqrt{\alpha_t}x_t}{\beta_t} + \frac{2\sqrt{\bar{\alpha}_{t-1}}x_0}{1 - \bar{\alpha}_{t-1}}\right)}_{:=B} x_{t-1} + C(x_t, x_0)\right)\right)
\end{aligned}$$

Considering the basic property of normal distribution, we have:

$$\tilde{\beta}_t = \frac{1}{A}, \quad 2\tilde{\mu}_t = \frac{B}{A}$$

5.2.2 DENOISING AUTOENCODERS

The diffusion model is recognized as an advanced form of the Hierarchical Variational Autoencoder (VAE), offering extensive adaptability in its setup. Important decisions in its configuration include the variances β_t of the forward process, the overall structure of the model, and the Gaussian parameters used for modeling the reverse process. This framework facilitates a newly defined explicit relationship between diffusion models and denoising score matching, leading to the establishment of the *Denoising Diffusion Probabilistic Model (DDPM)*, as presented by Ho et al. (2020).

Now we will examine our specifications for $p_\theta(x_{t-1}|x_t) = \mathcal{N}(x_{t-1}; \mu_\theta(x_t, t), \Sigma_\theta(x_t, t))$ for $1 < t \leq T$. To begin, we assign $\Sigma_\theta(x_t, t) = \sigma_t^2 \mathbf{I}$, treating these as unlearned, time-sensitive constants. We contemplate two contrasting approaches that serve as the extremities for the entropy of the reverse process for data with unit variance: setting $\sigma_t^2 = \beta_t$ and $\alpha_t^2 = \tilde{\beta}_t = \frac{1-\bar{\alpha}_{t-1}}{1-\bar{\alpha}_t}$. The first approach is best suited for when $x_0 \sim \mathcal{N}(0, I)$, and the latter is preferred if x_0 is deterministically fixed at a single value.

Furthermore, in defining $\mu_\theta(x_t, t)$, we introduce a targeted parameterization informed by our analysis of L_t . Given that $p_\theta(x_{t-1}|x_t) = \mathcal{N}(x_{t-1}; \mu_\theta(x_t, t), \sigma_t^2 \mathbf{I})$, the expression is as follows:

$$L_{t-1} = \mathbb{E}_q \left[\frac{1}{2\sigma_t^2} \|\tilde{\mu}_t(x_t, x_0) - \mu_\theta(x_t, t)\|^2 \right] + C \quad (34)$$

where C represents a constant that does not depend on θ .

Proof of (34): We know that for two Gaussian distribution p, q with single variable, KL divergence is:

$$KL(p||q) = \log \frac{\sigma_q}{\sigma_p}$$

Therefore, if we let $\sigma_t^2 = \tilde{\beta}_t$:

$$\begin{aligned} L_{t-1} &= \mathbb{E}_q[D_{KL}(q(x_{t-1}|x_t, x_0)||p_\theta(x_{t-1}|x_t))] \\ &= \mathbb{E}_q[D_{KL}(\mathcal{N}(x_{t-1}; \tilde{\mu}_t(x_t, x_0), \tilde{\beta}_t \mathbf{I})||\mathcal{N}(x_{t-1}; \mu_\theta(x_t, t), \sigma_t^2 \mathbf{I})))] \\ &= \mathbb{E}_q[\log 1 + \frac{\sigma_t^2 + (\tilde{\mu}_t(x_t, x_0) - \mu_\theta(x_t, t))^2}{2\sigma_t^2} - \frac{1}{2}] \\ &= \mathbb{E}_q[\frac{1}{2\sigma_t^2} \|\tilde{\mu}_t(x_t, x_0) - \mu_\theta(x_t, t)\|^2] \end{aligned}$$

Here we have $C = 0$, and if we let $\alpha_t^2 = \tilde{\beta}_t = \frac{1-\bar{\alpha}_{t-1}}{1-\bar{\alpha}_t}$, we have C nonzero.

So, we can see that the most straightforward parameterization of μ_θ is a model that predicts $\tilde{\mu}_t$, the forward process posterior mean. However, we can expand Eq(34) further by parameterizing Eq(28) as $x_t(x_0, \epsilon) = \sqrt{\bar{\alpha}_t}x_0 + \sqrt{1-\bar{\alpha}_t}\epsilon$ for $\epsilon \sim \mathcal{N}(0, \mathbf{I})$ and applying the forward process posterior formula (33):

$$L_{t-1} - C = \mathbb{E}_{x_0, \epsilon}[\frac{1}{2\sigma_t^2} \|\tilde{\mu}_t(x_t(x_0, \epsilon), \frac{1}{\sqrt{\bar{\alpha}_t}}(x_t(x_0, \epsilon) - \sqrt{1-\bar{\alpha}_t}\epsilon)) - \mu_\theta(x_t(x_0, \epsilon), t)\|^2] \quad (35)$$

$$= \mathbb{E}_{x_0, \epsilon}[\frac{1}{2\sigma_t^2} \|\frac{1}{\sqrt{\bar{\alpha}_t}}(x_t(x_0, \epsilon) - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}}\epsilon) - \mu_\theta(x_t(x_0, \epsilon), t)\|^2] \quad (36)$$

Proof of eq(35) and eq(36): Our idea is to rewrite x_t w.r.t x_0 and t :

$$\mathcal{N}(\cdot, \mu, \sigma) : x \mapsto \mathcal{N}(x, \mu, \sigma)$$

$$p(x_T) = \mathcal{N}(x_T; \mu, \sigma) \rightarrow \text{sample: } x_T \sim \mathcal{N}(\mu_\theta, \sigma_\theta), \quad \underbrace{\epsilon \sim \mathcal{N}(0, 1)}_{\delta_T = \mu_\theta + \sqrt{\sigma_\theta} \epsilon, x_T \text{ and } \delta_T \text{ have the same law}}$$

Then we can plug in $x_0 = \frac{x_t(x_0, \epsilon) - \sqrt{1 - \bar{\alpha}_t} \epsilon}{\sqrt{\bar{\alpha}_t}}$ and simplify the equation. Remember $\frac{\sqrt{\bar{\alpha}_{t-1}}}{\sqrt{\bar{\alpha}_t}} = \sqrt{\alpha_t}$ based on the definition of $\bar{\alpha}_t$. This step enables us to do the reparameterization from learning mean to learning noise in order to reduce cost.

Equation (36) reviews that μ_θ must predict $\frac{1}{\sqrt{\alpha_t}}(x_t - \frac{\beta_t}{\sqrt{1 - \bar{\alpha}_t}} \epsilon)$ given x_t . Since x_t is available as input to the model, we may choose the parameterization:

$$\mu_\theta(x_t, t) = \tilde{\mu}_t(x_t, \frac{1}{\bar{\alpha}_t}(x_t - \sqrt{1 - \bar{\alpha}_t} \epsilon_\theta(x_t))) = \frac{1}{\sqrt{\alpha_t}}(x_t - \frac{\beta_t}{\sqrt{1 - \bar{\alpha}_t}} \epsilon_\theta(x_t, t)) \quad (37)$$

where ϵ_θ is a function approximator intended to predict ϵ from x_t . To sample $x_{t-1} \sim p_\theta(x_{t-1}|x_t)$ is to compute $x_{t-1} = \frac{1}{\sqrt{\alpha_t}}(x_t - \frac{\beta_t}{\sqrt{1 - \bar{\alpha}_t}} \epsilon_\theta(x_t, t) + \sigma_t z)$, where $z \sim \mathcal{N}(0, I)$. It is clear that the complete sampling procedure resembles *Langevin Dynamics* that we have talked about in the SMLD section, with ϵ_θ as a learned gradient of the data density. Furthermore, with the parameterization (37), Eq(36) simplifies to:

$$\mathbb{E}_{x_0, \epsilon} \left[\frac{\beta_t^2}{2\sigma^2 \alpha_t (1 - \bar{\alpha}_t)} \|\epsilon - \epsilon_\theta(\sqrt{\bar{\alpha}_t} x_0 + \sqrt{1 - \bar{\alpha}_t} \epsilon, t)\|^2 \right] \quad (38)$$

Proof of (38): Plug in Eq(37 into Eq(36), we have:

$$\begin{aligned} L_{t-1} - C &= \mathbb{E}_{x_0, \epsilon} \left[\frac{1}{2\sigma_t^2} \left\| \frac{1}{\sqrt{\alpha_t}} (x_t(x_0, \epsilon) - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}} \epsilon) - \frac{1}{\sqrt{\alpha_t}} (x_t(x_0, \epsilon) - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}} \epsilon_\theta(x_t, t)) \right\|^2 \right] \\ &= \mathbb{E}_{x_0, \epsilon} \left[\frac{\beta_t^2}{2\sigma_t^2 \cdot \sqrt{\alpha_t} \cdot \sqrt{1-\bar{\alpha}_t}} \|\epsilon - \epsilon_\theta(x_t, t)\|^2 \right] \end{aligned}$$

Since $x_t = \sqrt{\bar{\alpha}_t}x_0 + \sqrt{1-\bar{\alpha}_t}\epsilon$, we can get Eq(38).

The equation Eq(38) corresponds to the variational limit of the reverse process similar to Langevin (37). This indicates that optimizing a goal of denoising score matching (24) aligns with employing variational inference to model the finite-time marginal distribution of a sampling chain that mimics Langevin Dynamics.

5.2.3 SIMULATION FOR DDPM

This section describes a numerical experiment conducted using the DDPM model to generate samples from the MNIST dataset.

Algorithm 1 Training	Algorithm 2 Sampling
1: repeat 2: $\mathbf{x}_0 \sim q(\mathbf{x}_0)$ 3: $t \sim \text{Uniform}(\{1, \dots, T\})$ 4: $\epsilon \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ 5: Take gradient descent step on $\nabla_\theta \ \epsilon - \epsilon_\theta(\sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\epsilon, t)\ ^2$ 6: until converged	1: $\mathbf{x}_T \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ 2: for $t = T, \dots, 1$ do 3: $\mathbf{z} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ if $t > 1$, else $\mathbf{z} = \mathbf{0}$ 4: $\mathbf{x}_{t-1} = \frac{1}{\sqrt{\alpha_t}} \left(\mathbf{x}_t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}} \epsilon_\theta(\mathbf{x}_t, t) \right) + \sigma_t \mathbf{z}$ 5: end for 6: return \mathbf{x}_0

Figure 17: Algorithm 1 Training and Algorithm 2 Sampling from Ho et al. (2020)

Based on the implemented Algorithms 1 and 2 based on Ho et al. (2020), we know that during the training process, the model samples x_0 from the data distribution and a random time step t . It then computes the noise ϵ and performs a gradient descent step to minimize the error between the predicted and actual noise. During the sampling

process, the model begins with a Gaussian noise sample x_T and iteratively denoises it through the reverse process to reconstruct the original data. The reverse process is approximated with:

$$x_{t-1} = \frac{1}{\sqrt{\alpha_t}} \left(x_t - \frac{1 - \alpha_t}{\sqrt{1 - \bar{\alpha}_t}} \epsilon_\theta(x_t, t) \right) + \sigma_t z$$

where z is Gaussian noise if $t > 1$, otherwise it is zero.

We attempt to reproduce the DDPM simulation process using the relatively simple dataset "MNIST". For this simulation, we utilized a UNet architecture introduced by Ronneberger et al. (2015)—a convolutional neural network known for its efficiency in image-to-image tasks—to predict the noise ϵ_θ . This architecture was chosen for its ability to capture multi-scale information through both downsampling and upsampling paths.

The simulation results show good validity. Take the generated result of Figure 18 as an example. From timestep 0 to timestep 190, the gradual refinement of the noisy input can be observed, revealing the digit '2' in increasing clarity and detail. This transformation illustrates the model's ability to effectively denoise and reconstruct coherent images from highly corrupted data, confirming the model's capability in capturing and learning the true distribution of the dataset.

Moreover, figure 19 provides an insightful visualization of the convergence behavior of the training process, showing the loss values over epochs across five runs, each with varying random seeds. The mean loss, depicted by the solid blue line, demonstrates a consistent downward trend, indicating effective learning and convergence of the model during training. The shaded blue region represents the confidence interval, encompassing the standard deviation of the loss across the different runs. Notably, this

shaded area narrows as the training progresses, highlighting reduced variability and increased consistency in performance. This graph validates the model's generation capability, confirming that the training process leads to a stable and reliable model, thereby supporting the validity of the results generated.

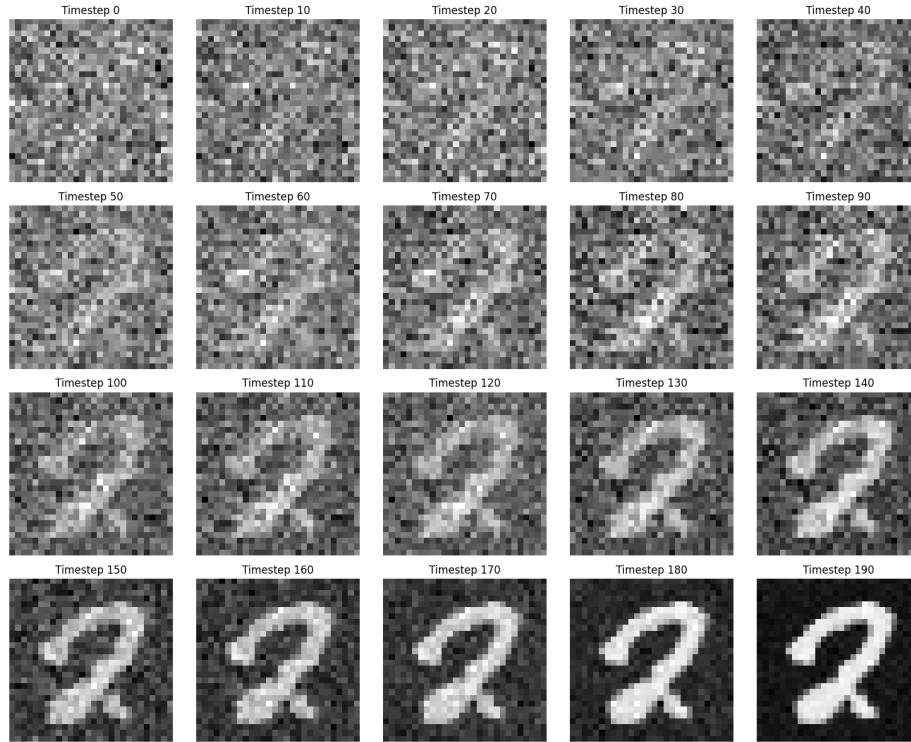


Figure 18: The denoised process of a random image from the MNIST dataset

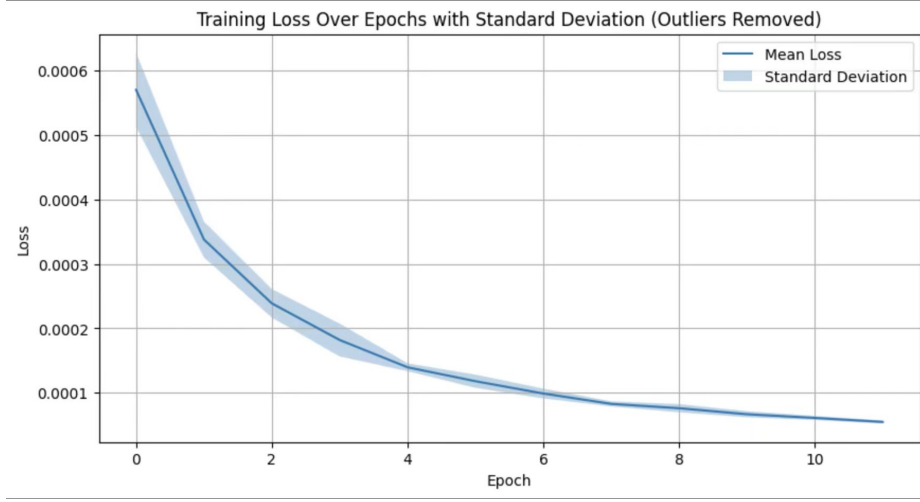


Figure 19: The convergent average loss for 5 runs

5.3 DDPM and SMLD with SDE

In this section, we introduce Song et al. (2020)’s contribution to combining SDE with the diffusion model. We will interpret and unify the SMLD and DDPM models from the perspective of SDE by adding detailed explanations and mathematical deduction to Song et al. (2020)’s paper.

5.3.1 FORWARD AND REVERSE PROCESS WITH SDE

From the previous sections, we know that the forward and reverse process of the diffusion model is a type of stochastic process. We will intuitively think of the tool for delineating the stochastic processes: Stochastic Differential Equations. However, the diffusion model we introduced is discrete, i.e. $x_0 \rightarrow x_1 \rightarrow \dots \rightarrow x_T$. By contrast, SDE is a continuous process, with the continuous expression of the forward process(similar to SMLD) as follows:

$$dx = f_t(x)dt + g_tdw \quad (39)$$

where the dx can be expressed in the following form:

$$dx = \lim_{\Delta t \rightarrow 0} (x_{t+\Delta t} - x_t)$$

Therefore, we can rewrite the previous discrete forward process($x_t \rightarrow x_{t+1}$) into the continuous form($x_t \rightarrow x_{t+\Delta t}$):

$$x_{t+\Delta t} = x_t + \underbrace{f_t(x_t)\Delta t}_{\text{drift term}} + \underbrace{g_t\sqrt{\Delta t}\epsilon}_{\text{diffusion term}}, \quad \epsilon \sim \mathcal{N}(0, I) \quad (40)$$

Similar to the deductions made in the DDPM section, we now have a continuous forward process and are intuitively seeking a continuous reverse process in the form of SDE that closely approximates the true distribution, which is:

$$dx = [f_t(x) - g_t^2 \nabla_x \log p_t(x)]dt + g_t dw \quad (41)$$

Proof of (41): We use the conditional probability to express (40):

$$\begin{aligned} p(x_{t+\Delta t}|x_t) &= \mathcal{N}(x_{t+\Delta t}; x_t + f_t(x_t)\Delta t, g_t^2 \Delta t I) \\ &\propto \exp\left(-\frac{\|x_{t+\Delta t} - x_t - f_t(x_t)\Delta t\|^2}{2g_t^2 \Delta t}\right) \end{aligned}$$

Using Bayes Theorem:

$$\begin{aligned} p(x_t|x_{t+\Delta t}) &= \frac{p(x_{t+\Delta t}|x_t)p(x_t)}{p(x_{t+\Delta t})} = p(x_{t+\Delta t}|x_t) \exp(\log p(x_t) - \log p(x_{t+\Delta t})) \\ &\propto \exp\left(-\frac{\|x_{t+\Delta t} - x_t - f_t(x_t)\Delta t\|^2}{2g_t^2 \Delta t} + \log p(x_t) - \log p(x_{t+\Delta t})\right) \end{aligned}$$

Applying Taylor expansion to $\log p(x_{t+\Delta t})$, where $p(x_t)$ is a bivariate function of (x_t, t) :

$$\log p(x_{t+\Delta t}) \approx \log p(x_t) + (x_{t+\Delta t} - x_t) \cdot \nabla_{x_t} \log p(x_t) + \Delta t \frac{\partial}{\partial t} \log p(x_t)$$

Plugging the Taylor expansion expression into the probability expression, we have:

$$p(x_t|x_{t+\Delta t}) \propto \exp \left(-\frac{\|x_{t+\Delta t} - x_t - [f_t(x_t) - g_t^2 \nabla_{x_t} \log p(x_t)] \Delta t\|^2}{2g_t^2 \Delta t} + \mathcal{O}(\Delta t) \right)$$

Since $\Delta t \rightarrow 0$, with infinitesimal time steps, $\mathcal{O}(\Delta t) \rightarrow 0$, $t \rightarrow t + \Delta t$.

Replace the unknown t with $t + \Delta t$:

$$p(x_t|x_{t+\Delta t}) \approx \exp \left(-\frac{\|x_t - x_{t+\Delta t} + [f_{t+\Delta t}(x_{t+\Delta t}) - g_{t+\Delta t}^2 \nabla_{x_{t+\Delta t}} \log p(x_{t+\Delta t})] \Delta t\|^2}{2g_{t+\Delta t}^2 \Delta t} \right)$$

So finally we have:

$$p(x_t|x_{t+\Delta t}) = \mathcal{N}(x_t; x_{t+\Delta t} - [f_{t+\Delta t}(x_{t+\Delta t}) - g_{t+\Delta t}^2 \nabla_{x_{t+\Delta t}} \log p(x_{t+\Delta t})] \Delta t, g_{t+\Delta t}^2 \Delta t I)$$

As $\Delta t \rightarrow 0$ and $dx = \lim_{\Delta t \rightarrow 0} (x_{t+\Delta t} - x_t)$, we have:

$$dx = [f_t(x) - g_t^2 \nabla_x \log p_t(x)] dt + g_t dW$$

Finally, we have the forward and reverse process of SDE, which aligns with the picture given in Song's paper:

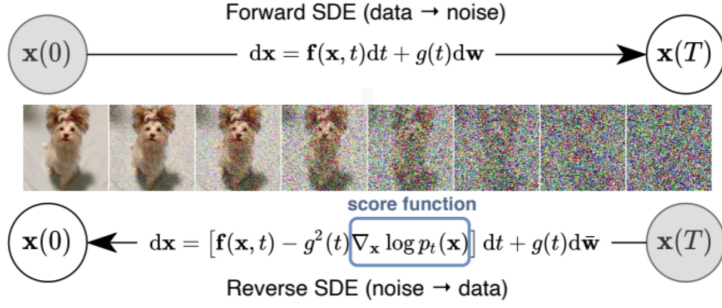


Figure 1: **Solving a reverse-time SDE yields a score-based generative model.** Transforming data to a simple noise distribution can be accomplished with a continuous-time SDE. This SDE can be reversed if we know the score of the distribution at each intermediate time step, $\nabla_{\mathbf{x}} \log p_t(\mathbf{x})$.

Figure 20: Figure 1 from Song’s Paper Score-Based Generative Modeling through SDE.

5.3.2 CONNECTION BETWEEN DDPM AND SMLD WITH SDE

Based on DDPM, $f(x, t)$ and $g(t)$ in the forward process are defined, so the only unknown term in Eq.(40) is $\nabla_x \log p_t(x)$, which is indeed the *score*. We have already explained how to estimate the score in the previous section of SMLD.

Song noticed that the DDPM and SMLD models both align with the framework of SDE by setting different $f(x, t)$ and $g(t)$.

For SMLD, based on Eq.(25), our diffusion formula is:

$$x_{t+1} = x_t + \epsilon \nabla_{x_t} \log p(x_t) + \sqrt{2\epsilon} z_t \quad (42)$$

$$= x_t + \epsilon s_{\theta}^*(x_t) + \sqrt{2\epsilon} z_t \quad (43)$$

To interpret the formula, we are relying on the $s_{\theta}^*(x_t)$ which should be big enough to let x_t be a Gaussian noise dominated by ϵ . We call this *Variance Exploding (VE)*.

The single step of the SMLD forward diffusion process can be expressed as:

$$x_{t+\Delta t} = x_t + \sqrt{s_\theta^*(x_{t+\Delta t})^2 - s_\theta^*(x_t)^2} \epsilon \quad (44)$$

$$= x_t + \sqrt{\frac{s_\theta^*(x_{t+\Delta t})^2 - s_\theta^*(x_t)^2}{\Delta t}} \sqrt{\Delta t} \epsilon \quad (45)$$

$$= x_t + \frac{\sqrt{\Delta s_\theta^*(x_t^2)}}{\Delta t} \sqrt{\Delta t} \epsilon \quad (46)$$

Comparing Eq.(46) with Eq.(40) to find the coefficients, we see that for SMLD with SDE, what we called *Variance Exploding (VE) SDE*, the coefficients follow:

$$\begin{cases} f(x_t, t) = 0 \\ g(t) = \frac{d}{dt} s_\theta^*(x_t)^2 \end{cases}$$

For DDPM, based on Eq.(28), our diffusion formula is:

$$x_t = \sqrt{\alpha_t} x_0 + \sqrt{1 - \alpha_t} \epsilon \quad (47)$$

$$= \sqrt{\alpha_t} x_{t-1} + \sqrt{1 - \alpha_t} \epsilon_t \quad (48)$$

We are using the $\sqrt{\alpha_t}$ which is small enough to constrain x_{t-1} , when $\sqrt{1 - \alpha_t}$ is relatively not big. So we call this *Variance Preserving (VP)*.

Rewrite Eq.(48) with $\alpha_t = 1 - \beta_t$ and a series of noise levels $\{\bar{\beta}_i := T\beta_i\}_{i=1}^T$:

$$x_{t+1} = \sqrt{1 - \beta_{t+1}} x_t + \sqrt{\beta_{t+1}} \epsilon \quad (49)$$

$$= \sqrt{1 - \frac{\bar{\beta}_{t+1}}{T}} x_t + \sqrt{\frac{\bar{\beta}_{t+1}}{T}} \epsilon \quad (50)$$

When $T \rightarrow \infty$, $\{\bar{\beta}_i\}_{i=1}^T \rightarrow \beta(t)$, where $t \in [0, 1]$, $\beta(\frac{i}{T}) = \bar{\beta}_i$, $\Delta t = \frac{1}{T}$. Also, when $x \rightarrow 0$, we have $(1 - x)^\alpha \approx 1 - \alpha x$. Therefore, we can rewrite the discrete formula to

a continuous one:

$$x_{t+\Delta t} = \sqrt{1 - \beta(t + \Delta t)\Delta t}x_t + \sqrt{\beta(t + \Delta t)\Delta t}\epsilon \quad (51)$$

$$\approx (1 - \frac{1}{2}\beta(t + \Delta t)\Delta t)x_t + \sqrt{\beta(t + \Delta t)}\sqrt{\Delta t}\epsilon \quad (52)$$

$$\approx (1 - \frac{1}{2}\beta(t)\Delta t)x_t + \sqrt{\beta(t)}\sqrt{\Delta t}\epsilon \quad (53)$$

Compare Eq.(53) with Eq.(40) to find the coefficients, we see that for DDPM with SDE, what we called *Variance Preserving(VP) SDE*, the coefficients follow:

$$\begin{cases} f(x_t, t) = -\frac{1}{2}\beta(t)x_t \\ g(t) = \sqrt{\beta(t)} \end{cases}$$

5.3.3 CONNECTION BETWEEN NOISE AND SCORE

In the end, we would like to figure out the relationship between noise and score. In the SMLD model, the score $s_\theta(x_t, t)$ is estimated, while in the DDPM model, the noise $\epsilon_\theta(x_t, t)$ is predicted. If the correlation between score and noise can be found, we can train the DDPM model under the framework of SDE by estimating the score.

By Eq.(28), we know that:

$$\epsilon(x_t, t) = \frac{x_t - \sqrt{\bar{\alpha}_t}x_0}{1 - \bar{\alpha}_t} \quad (54)$$

Also, by Eq.(12) we have:

$$s_\theta(x_t, t) = \nabla_{x_t} \log(x_t) \quad (55)$$

$$= -\frac{x_t - \mu}{\sigma^2} \quad (56)$$

$$= -\frac{x_t - \sqrt{\bar{\alpha}_t}x_0}{1 - \bar{\alpha}_t} \quad (57)$$

since $p(x_t)$ is a Gaussian distribution where the mean and variance correspond to Eq.(47).

We have deduced the correlation between noise and score:

$$s_\theta(x_t, t) = -\frac{1}{\sqrt{1 - \bar{\alpha}_t}}\epsilon_\theta(x_t, t) \quad (58)$$

6 Conclusion

In the first part of the thesis, we provided a comprehensive analysis of the theoretical connection between SGD and SDE. By modeling the stochastic behavior of SGD through SDE, we gained valuable insights into the convergence properties of SGD. The discrete SGD updates can be understood as approximations of continuous SDE, offering a framework to study the influence of noise on convergence rates and setting up conditions under which SGD can achieve optimal performance.

In the second part, we explored diffusion models in detail, linking them to the SDE framework with a focus on DDPM and SMLD. The DDPM model was implemented using the MNIST dataset, demonstrating the capacity of diffusion processes to generate samples through progressive denoising. UNet architecture, a convolutional neural network capable of capturing intricate image details via downsampling and upsampling paths, was used to predict the noise effectively. The results confirmed the model’s validity in reconstructing coherent images from noise. However, the limited number of epochs and timesteps constrained the full exploration of DDPM’s potential. Additionally, we did not conduct numerical experiments on the SMLD models due to time constraints, leaving this as an area for future research. Examining SMLD simulations within the SDE framework could provide further insights into their practical applications.

In summary, while this thesis offered a comprehensive overview of SDE’s application in machine learning, its main limitations are the lack of comprehensive simulations to substantiate the theoretical insights and the relatively small scale of the DDPM experiments. Future research should focus on conducting extensive numerical experiments on both SGD and diffusion models to validate the theoretical findings presented here. Further exploration into the application of SDE in other machine learning contexts,

such as reinforcement learning and neural architecture search, could also yield novel insights.

7 Acknowledgement

Many thanks to my supervisor Prof. Mathieu Lauriere; my previous research mentors Loucas Pillaud-Vivien, Roberto Fernandez, Jiding Zhang; and all the math professors at NYU Shanghai and NYU Courant, for their consistent guidance, kindness, and wisdom. Many thanks to my mom, dad, grandma, boyfriend, and my cat for their love, accompany, and money. Many thanks to the past, present, and future me, for everything.

I would like to keep the ending part short, as short as the ending of my 4 years of college time; but it is long enough, as long as I am grateful.

8 Appendix A: Introduction to SDE

This section establishes the essential concepts of Stochastic Differential Equations (SDE) to explore their applications in machine learning. We begin with outlining the fundamental principles of SDE, highlighting their importance in modeling systems influenced by random variations. Attention is then given to numerical methods for SDE, which are crucial for deriving solutions when analytical methods are insufficient. We also discuss Empirical Risk Minimization (ERM), illustrating its application in stochastic settings. Additionally, Stochastic Modified Equations are introduced, shedding light on their underlying logic. The section concludes with a focus on the Ornstein-Uhlenbeck process, providing a practical instance of SDE, readying the reader for its further exploration in subsequent sections. By laying this groundwork, we not only frame SDE within machine learning but also prepare the foundation for their application in the more complex generative models discussed in most of this paper.

8.1 What is SDE(Stochastic Differential Equations)

For stochastic analysis, a primary object of interest is the *Continuous-time Stochastic Process*, defined as a set of random variables X_t indexed by real numbers $t \geq 0$. Each realization of the stochastic process is a choice from the random variable X_t for each t , and is, therefore, a function of t . Any deterministic function $f(t)$ can be trivially considered as a stochastic process, with variance $V(f(t)) = 0$.

A fundamental construct in this framework is the *Wiener process* W_t , known as the mathematical abstraction of Brownian motion, first introduced by Oksendal (2003). It defines the scaling limit of random walks as the step size and time interval between steps both go to zero. Usually used to represent random, external influences on an

otherwise deterministic system. The definition of the Wiener process is based on the three constraints:

- For each t , the random variable W_t is normally distributed with **mean 0** and **variance t** . i.e. $W_t - W_s \sim N(0, t - s)$.
- For each $t_1 < t_2$, the normal random variable $W_{t_2} - W_{t_1}$ is **independent** of the random variable W_{t_1} , and in fact **independent** of all W_t , $0 \leq t \leq t_1$.
- The Wiener process W_t can be represented by **continuous** paths.

Central to the study of SDE is the concept of a *Diffusion process*, characterized by the coexistence of deterministic drift terms and stochastic diffusion terms, represented as:

$$dX = a(t, X)dt + b(t, X)dW_t, \quad (59)$$

- $a(t, X)dt$: drift term because it captures the average or expected rate of change of the process X if no randomness was involved.
- $b(t, X)dW_t$: diffusion term because it scales the magnitude of the randomness by the increment of W .

This is what we call an SDE equation. It can be expressed in integral form as

$$X(t) = X(0) + \int_0^t a(s, X)ds + \int_0^t b(s, X)dW_s.$$

To solve such an equation, one employs *Ito's formula*, which acts as the stochastic analog of the classical chain rule. For a differentiable function $f(t, X)$, Ito's formula expands to include a second-order term that accounts for the quadratic variation of

the Wiener process:

$$df(t, X_t) = \left(\frac{\partial f}{\partial t} + a(t, X) \frac{\partial f}{\partial x} + \frac{1}{2} b^2(t, X) \frac{\partial^2 f}{\partial x^2} \right) dt + b(t, X) \frac{\partial f}{\partial x} dW_t.$$

Integral expressions in stochastic calculus diverge from the traditional *Riemann integral*, with Ito's integral defined through limiting processes involving the left endpoint of partition intervals, following the paper of Ito (1965). Such integrals are intrinsic to the modeling of stochastic systems and often yield solutions that are a synthesis of predictable and random components.

- **[Riemann integral]**: $\int_c^d f(x)dx = \lim_{\Delta t \rightarrow 0} \sum_{i=1}^n f(t'_i) \Delta t_i$, where $\Delta t_i = t_i - t_{i-1}$ and $t_{i-1} \leq t'_i \leq t_i$. Here t'_i may be chosen at any point in the interval (t_{i-1}, t_i) .
- **[Ito's integral]**: $\int_c^d f(t)dW_t = \lim_{\Delta t \rightarrow 0} \sum_{i=1}^n f(t_{i-1}) \Delta W_i$, where $\Delta W_i = W_{t_i} - W_{t_{i-1}}$. Here t'_i must be the left endpoint of (t_{i-1}, t_i) . Since f and W_t are random variables, so is the Ito's integral $I = \int_c^d f(t)dW_t$, thus:

$$I = \int_c^d f dW_t$$

is expressed in differential form as

$$dI = f dW_t$$

where dW_t is called white noise. A typical solution is a combination of drift and the diffusion of Brownian motion.

A notable application is the *Black-Scholes Differential Equation* introduced by Black and Scholes (1973), modeling financial derivatives through a geometric Brownian

motion:

$$\begin{cases} dX = \mu X dt + \sigma X dW_t, \\ X(0) = X_0, \end{cases} \quad (60)$$

where μ represents the drift coefficient, and σ the volatility. The solution to this SDE is:

$$X(t) = X_0 e^{(\mu - \frac{1}{2}\sigma^2)t + \sigma W_t}. \quad (61)$$

Proof of (61): Write $X = f(t, Y) = X_0 e^Y$, where $Y = (\mu - \frac{1}{2}\sigma^2)t + \sigma W_t$. By Ito's formula:

$$dX = X_0 e^Y dY + \frac{1}{2} e^Y dY dY$$

Where $dY = (\mu - \frac{1}{2}\sigma^2)dt + \sigma dW_t$. Using the differential identities from the Ito formula,

$$dY dY = \sigma^2 dt$$

and therefore

$$\begin{aligned} dX &= X_0 e^Y (\mu - \frac{1}{2}\sigma^2)dt + X_0 e^Y \sigma dW_t + \frac{1}{2} \sigma^2 e^Y dt \\ &= X_0 e^Y \mu dt + X_0 e^Y \sigma dW_t \\ &= \mu X dt + \sigma X dW_t \end{aligned}$$

Verification via Ito's formula confirms that the dynamics of $X(t)$ align with the original SDE, reinforcing the interplay between deterministic growth and stochastic perturbations in the evolution of financial prices.

8.2 Numerical Methods for SDE

The approximation of solutions to Stochastic Differential Equations (SDE) over the interval $[c, d]$ can be approached by methods such as the *Euler-Maruyama method*, which we will explain here in detail. We consider a partition of the interval with points $c = t_0 < t_1 < \dots < t_n = d$ and try to approximate the values of $X(t)$ at these points, starting with the initial condition $X(c) = X_c$. The Euler-Maruyama method introduced by Oksendal (2003) provides an iterative procedure to approximate the solution of the SDE

$$\begin{cases} dX(t) = a(t, X)dt + b(t, X)dW_t, \\ X(c) = X_c. \end{cases}$$

This is done by iteratively computing

$$\begin{aligned} \omega_0 &= X_0, \\ \omega_{i+1} &= \omega_i + a(t_i, \omega_i)\Delta t_{i+1} + b(t_i, \omega_i)\Delta W_{i+1}, \end{aligned}$$

where $\Delta t_{i+1} = t_{i+1} - t_i$ and ΔW_{i+1} is the increment of the Wiener process between t_i and t_{i+1} , modeled as $\Delta W_i = z_i\sqrt{\Delta t_i}$ with z_i drawn from a standard normal distribution $\mathcal{N}(0, 1)$.

For instance, applying the Euler-Maruyama method to the Black-Scholes SDE, we obtain the discrete-time approximation

$$\begin{aligned} \omega_0 &= X_0, \\ \omega_{i+1} &= \omega_i + \mu\omega_i\Delta t_i + \sigma\omega_i\Delta W_i. \end{aligned}$$

As another illustration, the *Langevin equation*, given by

$$dX(t) = -\mu X(t)dt + \sigma dW_t,$$

with positive constants μ and σ , leads to the *Ornstein-Uhlenbeck process* through its Euler-Maruyama approximation:

$$\begin{aligned}\omega_0 &= X_0, \\ \omega_{i+1} &= \omega_i - \mu\omega_i\Delta t_i + \sigma\Delta W_i,\end{aligned}$$

for $i = 1, \dots, n$. These examples showcase the Euler-Maruyama method's application to classical SDE in financial mathematics and physical systems.

8.3 Empirical Risk Minimization (ERM)

Empirical Risk Minimization (ERM) is a widely used principle in machine learning where the goal is to find the parameter vector that minimizes the expected loss. The problem can be described as:

$$\min_{x \in \mathbb{R}^d} f(x) := \mathbb{E}_\gamma [f_\gamma(x)],$$

where the family of functions $\{f_\gamma : \gamma \in \Gamma\}$ maps \mathbb{R}^d to \mathbb{R} , and γ denotes a Γ -valued random variable over which the expectation is computed.

To reach optimization in the context of ERM, Stochastic Gradient Algorithms (SGA) are frequently applied. Consider the optimization via standard Gradient Descent (GD), which uses the gradient of the function f at a point $x = (x_1, \dots, x_d) \in \mathbb{R}^d$.

The gradient is given by:

$$\nabla f(x) = \begin{pmatrix} \partial_{x_1} f(x_1, \dots, x_d) \\ \vdots \\ \partial_{x_d} f(x_1, \dots, x_d) \end{pmatrix} \in \mathbb{R}^d.$$

The GD update rule is then expressed recursively as:

$$x_{k+1} = x_k - \eta \nabla f(x_k) = x_k - \eta \nabla \mathbb{E}_\gamma [f_\gamma(x_k)], \quad (62)$$

where $k \geq 0$ represents the iteration index and η is the learning rate, which is a small positive scalar that determines the step size of each iteration.

Stochastic Gradient Descent (SGD), a variant of GD, simplifies the computational burden by approximating the true gradient with a sample at each iteration. In SGD, the update takes the form:

$$x_{k+1} = x_k - \eta \nabla f_{\gamma_k}(x_k), \quad (63)$$

with γ_k representing an independent and identically distributed (i.i.d.) sample from the same distribution as γ . This substitution relies on the unbiasedness of the sampled gradient, where $\mathbb{E}[\nabla f_{\gamma_k}(x_k) | x_k] = \nabla \mathbb{E} f(x_k)$, thus ensuring that the SGD iteration is an unbiased estimate of the GD iteration.

8.4 Stochastic Modified Equations(SME)

The evolution of a system in the presence of a gradient of a potential function f can often be described by the deterministic equation

$$\frac{dx}{dt} = -\nabla f(x).$$

8.4.1 HEURISTIC MOTIVATIONS

The heuristic underpinning of stochastic modified equations stems from the examination of Stochastic Gradient Algorithms (SGA). To begin, one may express the iteration of the SGA by rewriting Eq(63) as:

$$\begin{aligned} x_{k+1} &= x_k - \eta \nabla f_{\gamma_k}(x_k) \\ &= x_k - \eta \nabla f_{\gamma_k}(x_k) + \eta \nabla f(x_k) - \eta \nabla f(x_k) \\ &= x_k - \eta \nabla f(x_k) + \underbrace{\eta (\mathbb{E}_{\gamma} [f_{\gamma_k}(x_k)] - \nabla f_{\gamma_k}(x_k))}_{\sqrt{\eta} V_k} \end{aligned}$$

where we introduce the term $\sqrt{\eta} V_k(x_k, \gamma_k)$ to capture the fluctuations around the gradient of f . A straightforward calculation shows that:

$$\begin{aligned} \mathbb{E}[V_k|x_k] &= 0 \\ \text{Cov}[V_k|x_k] &= \eta \Sigma(x_k) := \mathbb{E}[(\nabla f_{\gamma_k}(x_k) - \nabla f(x_k))(\nabla f_{\gamma_k}(x_k) - \nabla f(x_k))^T|x_k] \end{aligned}$$

i.e. conditional on x_k , $V_k(x_k)$ has 0 mean and covariance $\eta \Sigma(x_k)$. Here Σ is simply the conditional covariance of the stochastic gradient approximation ∇f_{γ} or ∇f .

We can consider this iteration scheme in the context of a time-homogeneous Ito

stochastic differential equation (SDE), which is of the form

$$dX_t = b(X_t)dt + \sqrt{\eta}\sigma(X_t)dW_t,$$

where b denotes the drift term and σ is the diffusion matrix. By discretizing this SDE using the Euler-Maruyama method with a step-size of η and setting $b = -\nabla f$ and $\sigma(x) = \Sigma(x)^{\frac{1}{2}}$, we align the first and second conditional moments with those of the SGA iteration, yielding the approximation

$$dX_t = -\nabla f(X_t)dt + (\eta\Sigma(X_t))^{\frac{1}{2}}dW_t.$$

This correspondence between the SGA and the SDE justifies the inclusion of the $\sqrt{\eta}$ factor on the diffusion term, highlighting that as the learning rate η decreases, the influence of stochastic fluctuations in the algorithm's iterations should correspondingly diminish.

8.5 Ornstein-Uhlenbeck Process

8.5.1 DEFINITION

We will now explain the Ornstein-Uhlenbeck process, originally formulated by Uhlenbeck and Ornstein (1930). This stochastic process follows the following Stochastic Differential Equation (SDE):

$$dX_t = \kappa(\theta - X_t) dt + \sigma dW_t \tag{64}$$

Here, W_t represents a standard Brownian motion over the interval $t \in [0, \infty)$. The fixed parameters are defined as:

- $\kappa > 0$, representing the mean reversion rate;
- θ , the long-term mean or equilibrium level of the process;
- $\sigma > 0$, indicating the volatility or intensity of the random deviations described by Brownian motion.

8.5.2 PROPERTIES OF MEAN REVERSION

Disregarding the stochastic disturbances due to dW_t , the path of X_t generally trends towards its mean θ . The value of X_t gravitates back to this mean at an exponential rate κ , proportionally to the deviation from θ .

Examining the corresponding ordinary differential equation $dx_t = \kappa(\theta - x) dt$ yields the solution:

$$\frac{\theta - x_t}{\theta - x_0} = e^{-\kappa(t-t_0)}, \text{ or } x_t = \theta + (x_0 - \theta)e^{-\kappa(t-t_0)}$$

This property designates the Ornstein-Uhlenbeck process as a mean-reverting process.

8.5.3 ANALYTICAL SOLUTION

Next, we introduce the analytical solution to the SDE described in (64). The definition of the Ornstein-Uhlenbeck process for any $0 \leq s \leq t$ is given by:

$$X_t = \theta + (X_s - \theta)e^{-\kappa(t-s)} + \sigma \int_s^t e^{-\kappa(t-u)} dW_u$$

where the integral is evaluated using Itô's calculus.

For a fixed s and t , the random variable X_t , given X_s , follows a normal distribution with:

$$\text{mean} = \theta + (X_s - \theta)e^{-\kappa(t-s)}$$

$$\text{variance} = \frac{\sigma^2}{2\kappa}(1 - e^{-2\kappa(t-s)})$$

The mean value of X_t aligns with the heuristic solution derived from the ODE, emphasizing the Ornstein-Uhlenbeck process as a time-homogeneous Itô diffusion.

9 Appendix B: Python code for simulations

For section 4.3, 3.3, 5.2.3, the python code can be found here: https://github.com/SelenaGe/DDPM_Simulation. The file names are:





 SelenaGe	Add files via upload
 grq_ddpm_simulation_mnist.py	Add files via upload
 ou_process_simulation.py	Add files via upload
 sgd_simulation.py	Add files via upload

Figure 21: Github file names for numerical experiments

References

- Alnur Ali, Edgar Dobriban, and Ryan Tibshirani. The implicit regularization of stochastic gradient flow for least squares. In *International conference on machine learning*, pages 233–244. PMLR, 2020.
- F. Bach and E. Moulines. Non-asymptotic analysis of stochastic approximation algorithms for machine learning. In *Advances in Neural Information Processing Systems*, pages 451–459, 2011.
- Michel Benaïm. Dynamics of stochastic approximation algorithms. In *Seminaire de probabilités XXXIII*, pages 1–68. Springer, 2006.
- Fischer Black and Myron S. Scholes. The pricing of options and corporate liabilities. *Journal of Political Economy*, 81(3):637–654, 1973.
- Léon Bottou, Frank E Curtis, and Jorge Nocedal. Optimization methods for large-scale machine learning. *SIAM Review*, 60(2):223–311, 2018.
- V. Cevher and Bng Công Vũ. On the linear convergence of the stochastic gradient method with constant step-size. *Optimization Letters*, 13(5):1177–1187, 2019.
- Chen-Hao Chao, Wei-Fang Sun, Bo-Wun Cheng, Yi-Chen Lo, Chia-Che Chang, Yu-Lun Liu, Yu-Lin Chang, Chia-Ping Chen, and Chun-Yi Lee. Denoising likelihood score matching for conditional score-based data generation. *arXiv preprint arXiv:2203.14206*, 2022.
- Valentin De Bortoli, Michael Hutchinson, Peter Wirsberger, and Arnaud Doucet. Target score matching. *arXiv preprint arXiv:2402.08667*, 2024.
- J Harold, G Kushner, and George Yin. Stochastic approximation and recursive algorithm and applications. *Application of Mathematics*, 35, 1997.

- Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *arXiv preprint arXiv:2006.11239v2*, pages 1–4, 13–14, 2020.
- Aapo Hyvärinen and Peter Dayan. Estimation of non-normalized statistical models by score matching. *Journal of Machine Learning Research*, 6(4):695–709, 2005.
- Kiyosi Ito. *Calculus of Variations*. Springer, 1965.
- Rafail Khasminskii. *Stochastic stability of differential equations*, volume 66. Springer Science & Business Media, 2011.
- Qianxiao Li, Cheng Tai, and E Weinan. Stochastic modified equations and dynamics of stochastic gradient algorithms i: Mathematical foundations. *The Journal of Machine Learning Research*, 20(1):1474–1520, 2019.
- Bernt Oksendal. *Stochastic differential equations: an introduction with applications*, volume 3. Springer, 2003.
- Chirag Pabbaraju, Dhruv Rohatgi, Anish Sevekari, Holden Lee, Ankur Moitra, and Andrej Risteski. Provable benefits of score matching. *arXiv preprint arXiv:2306.01993*, 2023.
- Scott Pesme, Loucas Pillaud-Vivien, and Nicolas Flammarion. Implicit bias of sgd for diagonal linear networks: a provable benefit of stochasticity. *Advances in Neural Information Processing Systems*, 34:29218–29230, 2021.
- H. Robbins and S. Monro. A stochastic approximation method. *Ann. Math. Statistics*, 22:400–407, 1951.
- Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015*, pages 234–241. Springer, 2015.

- O. Shamir and T. Zhang. Stochastic gradient descent for non-smooth optimization: convergence results and optimal averaging schemes. In *International Conference on Machine Learning*, pages 71–79, 2013.
- Yang Song and Stefano Ermon. Generative modeling by estimating gradients of the data distribution. *arXiv preprint arXiv:1907.05600*, 2019. NeurIPS 2019 (Oral).
- 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:2011.13456v2*, 2020.
- George E. Uhlenbeck and Leonard S. Ornstein. On the theory of the brownian motion. *Physical Review*, 36(5):823, 1930.
- Richard S Varga. *Geršgorin and his circles*, volume 36. Springer Science & Business Media, 2010.