AS.110.653 : Stochastic Differential Equations

Applying Stochastic Differential Equation on the Analysis of Stochastic Gradient Descent

James Guo (sguo45)

May 11, 2025

I Introduction

I.1 Machine Learning Objectives

In the basic case of machine learning, we consider the task of learning the function $f : \mathcal{X} \to \mathcal{Y}$, where $\mathcal{X}$ is the class of features and $\mathcal{Y}$ is the class of response. Note that we do not know the function f and we want to learn it. For the first part of the report, the constructions are mainly from Chapter 7 of [2].

In the typical setup, there exists a set of data $\mathcal{D} \subset \mathcal{X} \times \mathcal{Y}$ for us to learn containing $\big(x, f(x)\big)$ or following certain distributions, and our goal is to learn the function $\tilde{f} : \mathcal{X} \to \mathcal{Y}$ to our best capabilities.

While learning models, we need to find how much we are off, hence we shall introduce a loss function.

Definition I.1.1. Loss Function.

Let $\mathcal{L} : F \to \mathbb{R}_{\geq 0}$ be the loss function, where F is the class of all functions such that $\tilde{f} : \mathcal{X} \to \mathcal{Y}$. $\mathcal{L}$ evaluates how much $\tilde{f}$ is off from the actual model f. ⌟

In reality, we do not have access to f, so we we consider $\mathcal{D} = \{(x_k, y_k)\}_{k=1}^n$ as a data set, and evaluate the function we learned based on these data.

Example I.1.2. Examples of Loss Functions.

In general, there are various kinds of loss functions, corresponding to different class of $\mathcal{Y}$:

- When $\mathcal{Y} = \mathbb{R}^n$, we often use the **Least-Squares Loss**:

$$\mathcal{L}(\tilde{f}; \mathcal{D}) = \frac{1}{n} \sum_{k=1}^{n} \big(y_n - \tilde{f}(x_n)\big)^2.$$

- When $\mathcal{Y} = \{0, 1\}$, we often use the **Cross-Entropy Loss**:

$$\mathcal{L}(\tilde{f}; \mathcal{D}) = -\sum_{k=1}^{N} \big[y_k \log\big(f(x_k)\big) + (1 - y_k) \log\big(1 - f(x_k)\big)\big].$$

There are also many other examples of loss functions defined on specific class of $\mathcal{Y}$. ⌟

The goal of machine learning is to minimize $\mathcal{L}$ with our selection of F. In most cases, we have F as a fixed type of function with different d-dimensional parameters $\theta \in \mathbb{R}^d$, often denoted $\tilde{f}(x;\theta)$, in which you can consider $\tilde{f}(\theta) : \mathcal{X} \to \mathcal{Y}$ to be a function, whose loss is denoted $\mathcal{L}(\theta;\mathcal{D})$.

I.2 Gradient Descent

Now, our goal is effectively solve the optimization problem:

$$\min_{\theta\in\mathbb{R}^d} \mathcal{L}\big(\tilde{f}(\theta);\mathcal{D}\big).$$

An effective approach is the **gradient descent** method, based on Section 3 of [3].

To find the minimum parameter θ, we can use the gradient descent method for $\theta^{(t)}$ with discrete time that:

$$\theta^{(t+1)} = \theta^{(t)} - \eta\nabla\mathcal{L}\big(\theta^{(t)};\mathcal{D}\big), \tag{1}$$

where $0 < \eta \ll 1$ is a small learning rate.

Remark I.2.1. We can recall *Euler's method* as an numerical method from ODEs, so we can think of the gradient descent process as the discretization of continuous ODE initial value problem:

$$\begin{cases} \dfrac{d\theta}{dt} = -\nabla\mathcal{L}(\theta;\mathcal{D}), \\ \theta(0) = \theta_0, \end{cases}$$

where θ_0 is the initialized parameters for the learning network. ⌟

Note that the **gradient descent** method is deterministic, so we can anticipate the **same** result in each run despite the choice of the initial parameters. However, it still exhibits some issue.

Remark I.2.2. Computation Efficiency.

Note that in (1), for each step of the iteration, the gradient descent needs to compute the gradient of the loss of this parameter evaluated at all data points in $\mathcal{D}$. When $|\mathcal{D}|$ is large, the learning is not efficient. ⌟

I.3 Stochastic Gradient Descent

To get around this large number of data, computer scientists decided to introduce an additional method, namely the **stochastic gradient descent**, in which they have a subset of the data $\mathcal{B} \subset \mathcal{D}$ called a **data batch** used for training, which is replacing (1) by:

$$\theta^{(t+1)} = \theta^{(t)} - \eta\nabla\mathcal{L}\big(\theta^{(t)};\mathcal{B}\big), \tag{2}$$

and having $0 < \eta \ll 1$ as the small learning rate.

II Preliminaries

II.1 Formalizing Stochastic Gradient Descent

To closely investigate the case here, we want to formally define the concept of batch using mathematical language here, corresponding to Section 3.1 and 3.2 in [3].

Definition II.1.1. Batch Set.

Here, we consider a general probability space as $(\Omega, \mathcal{F}, \mathbb{P})$ and the σ-algebra pair $(\Gamma, \mathcal{F}_\Gamma)$ as the *index space* of the data set. Consider $\gamma : \Omega \to \Gamma$ as a random variables and $(r, \theta) \mapsto \mathcal{L}_r(\theta) = \mathcal{L}(\theta; \mathcal{D}_r)$ as a measurable map from $\Gamma \times \mathbb{R}^d \to \mathbb{R}$, so for each fixed x, $\mathcal{L}_\gamma(x)$ is a random variable. ⌟

Moreover, we assume the following three properties of the $\mathcal{L}_\gamma(\theta)$ function (Assumption 1 in [3]), which is true when Γ is finite:

1. $\mathcal{L}_\gamma(\theta) \in L^1(\Omega)$ for all $\theta \in \mathbb{R}^d$.
2. $\mathcal{L}_\gamma(\theta)$ is continuously differentiable in θ almost surely, and for all $R > 0$, there exists a random variable $M_{R,\gamma}$ such that:
$$\max_{|\theta| \le R} |\nabla \mathcal{L}_\gamma(\theta)| \le M_{R,\gamma}$$
almost surely and $\mathbb{E}|M_{R,\gamma}| < +\infty$.
3. $\nabla \mathcal{L}_\gamma(\theta) \in L^2(\Omega)$ for all $\theta \in \mathbb{R}^d$.

Here, by condition 1, we can have:

$$\mathcal{L}(\theta) := \mathbb{E}\mathcal{L}_\gamma(\theta) = \int_\Omega \mathcal{L}_{\gamma(\omega)}(\theta) d\mathbb{P}(\omega).$$

By condition 2 and the dominated convergence theorem, we have:

$$\mathbb{E}[\nabla \mathcal{L}_\gamma] = \mathbb{E}[\nabla \mathcal{L}_\gamma] = \nabla \mathcal{L}.$$

Then, we want to model the gradient descent (SGD) process with stochastic differential equations.

II.2 Euler-Marayama Approximation

Just like Euler's method in ODEs, we can use the similar idea to approximate the SDEs as well.

Definition II.2.1. Euler-Marayama Approximation (Section 9.1 in [4]).

Let $\{X_t\}_{0 \le t \le T}$ be an Itô process satisfying the SDE:

$$\begin{cases} dX_t = b(t, X_t)dt + \sigma(t, X_t)dB_t, & \text{for } 0 \le t \le T, \\ X_0 = x. \end{cases}$$

Then, we can approximate via a discrete time steps $0 = \tau_0 < \tau_1 < \cdots < \tau_N = T$, so approximation for $\{Y_{\tau_i}\}_{i=0}^N$ through the recursive definition:

$$\begin{cases} Y_0 = x, \\ Y_{\tau_{n+1}} = Y_{\tau_n} + b(\tau_n, Y_n)(\tau_{n+1} - \tau_n) + \sigma(\tau_n, Y_n)(B_{\tau_{n+1}} - B_{\tau_n}). \end{cases}$$ ⌟

The **Euler-Marayama Approximation** is very similar to the Euler's method, they are both numerical approximations, but the Euler-Marayama Approximation has a Brownian motion term in the equation.

Remark II.2.2. Goal of Connection between SGD with SDE.

The goal of this derivation is to consider some stochastic process X_t for $t \geq 0$ modeling the change in parameters in the stochastic gradient descent (SGD) process, and find the Itô diffusion:

$$dX_t = \underbrace{b(X_t, \eta)}_{\text{drift}} dt + \sqrt{\eta} \underbrace{\sigma(X_t, \eta)}_{\text{diffusion}} dB_t, \qquad X_0 = \theta_0,$$

where B_t for $t \geq 0$ is the d-dimensional Brownian motion independent of γ_k for all $k \geq 0$. ⌟

In particular, we want to model X_t as the parameters and think of it as an Itô diffusion such that the diffusion part is caused by the randomness of selecting the batch.

Remark II.2.3. Furthermore, we assume that the SDE satisfies the **linear growth** and **Lipschitz condition** from existence and uniqueness theorem (Theorem 5.2.1 in [1]). ⌟

II.3 Weak Approximation

In developing our relation between the Stochastic process X_t and the actual parameter learning, we would need to establish a weaker convergence than the typical converges, such as L^2, pointwise, or uniform convergence. Such convergence would be called **weak convergence** over a class of functions, according to Section 3.2 of [3].

First of all, we consider a class of functions that is weaker than the linear growth condition.

Definition II.3.1. Polynomial Growth Functions (Definition 1 in [3]).

$G \subset \mathcal{C}(\mathbb{R}^d, \mathbb{R})$ is the class of polynomial growth functions if for all $g \in G$, there exists positive integers $C_1, C_2 > 0$ such that:

$$|g(x)| \leq C_1(1 + |x|^{2C_2}) \text{ for all } x \in \mathbb{R}^d.$$

Moreover, for any integer $\alpha \geq 1$, we let $G^\alpha \subset \mathcal{C}^\alpha(\mathbb{R}^d, \mathbb{R})$ be the class such that all $g \in \mathcal{G}^\alpha$ satisfy:

$$\partial_a g(x) \in G \text{ for all multi-indices } a \text{ such that } |a| \leq \alpha.$$

⌟

Then, we consider another type of "convergence" to evaluate how close our SDE corresponds to the SGD process.

Definition II.3.2. Weak Approximation.

Let $T > 0$ be fixed, $\eta \in (0, 1 \wedge T)$, and $\alpha \geq 1$ be an integer. Let $N = \lfloor T/\eta \rfloor$, and consider a continuous stochastic process $\{X_t : t \in [0, T]\}$ and a discrete stochastic process $\{\theta_k : k = 0, \cdots, N\}$. $\{X_t\}$ is an order α weak approximation of $\{\theta_k\}$ if for every $g \in \mathcal{G}^{\alpha+1}$, there exists a positive constant C, independent of η, such that:

$$\max_{0 \leq k \leq N} \left| \mathbb{E}[g(\theta_k) - \mathbb{E}[g(X_{k\eta})] \right| \leq C\eta^{\alpha}.$$ ⌟

Here, we eventually want to demonstrate that the continuous stochastic process $\{X_t\}_{t \geq 0}$ is an **weak approximation** of $\{\theta_k\}_{k \in \mathbb{N}}$ in (2), which was the parameters of the actual SGD process.

Remark II.3.3. Notations in Derivations.

For simplicity of notation, we adopt the similar conventions as [3]:

- To get rid of the redundant learning rate (η) factor in the subscript, so:

$$\tilde{X}_k := X_{k\eta}.$$

- For both stochastic processes $\{X_t\}$ and $\{\theta_t\}$, we assume the initial condition $X_0 = \theta_0$. ⌟

III Review of the Papers

III.1 Connecting SDG with SDEs

The main result in [3] is to verify the **weak convergence** of SDEs towards SDG process.

Definition III.1.1. SDE Approximation of SGD.

We define the stochastic process $\{X_t\}_{0 \leq t \leq T}$ satisfying the following SDE:

$$\begin{cases} dX_t = -\nabla \mathcal{L}(X_t)dt + \sqrt{\eta}\Sigma(X_t)^{\frac{1}{2}}dB_t, \\ X_0 = x_0, \end{cases} \tag{3}$$

where $\Sigma(x) = \mathbb{E}\left[\left(\nabla \mathcal{L}_\gamma(x) - \nabla \mathcal{L}(x)\right)\left(\nabla \mathcal{L}_\gamma(x) - \nabla \mathcal{L}(x)\right)^{\intercal}\right]$ and x_0 is the initialized parameter. ⌟

Note that the SDE has the drift part exactly as the gradient of the loss function and the diffusion is about a variance matrix and also correlated with the learning rate.

Theorem III.1.2. Weak Convergence of SDE system (Corollary 10 in [3]).

Let $T > 0$ and $0 < \eta \ll 1 \wedge T$ and let $N = \lfloor T/\eta \rfloor$. Consider $\{\theta_t\}_{t=0}^{T}$ be as defined in (2). Also suppose that:

- $\mathcal{L} \equiv \mathbb{E}[\mathcal{L}_\gamma]$ is continuously differentiable, and $\mathcal{L} \in G^3$.
- $\nabla \mathcal{L}_\gamma$ satisfies the Lipschitz condition:

$$|\nabla \mathcal{L}_\gamma(x) - \nabla \mathcal{L}_\gamma(y)| \leq L_\gamma |x - y|,$$

  for all $x, y \in \mathbb{R}^n$ and L_γ is a random variable positive almost surely and has finite expectation.

Then $\{X_t\}_{0 \leq t \leq T}$ in Definition III.1.1 is an order 1 weak approximation of the SGD.

Remark III.1.3. (Remark 11 and 12 in [3]). The important constraint is on the Lipschitz condition on $\nabla \mathcal{L}_\gamma$ to ensure the existence of a strong solution. Also, the model is aiming for a generalized case, so the estimates are not tight for some coefficients, such as the constant factor in the weak convergence above. ⌟

Here, we provide an example given in Section 5.1 of [3].

Example III.1.4. A Linear Loss Function.

Suppose $H \in \mathbb{R}^{d \times d}$. Consider the loss function:

$$\mathcal{L}_\gamma(\theta) := \frac{1}{2}(x - \gamma)^\intercal H (x - \gamma) - \frac{1}{2}\operatorname{Tr}(H) \text{ and } \gamma \sim \mathcal{N}(0, \mathrm{Id}).$$

In particular, it has $\mathcal{L}(\theta) \equiv \mathbb{E}[\mathcal{L}_\gamma(\theta)] = \frac{1}{2} x^\intercal H x$.
Here, we have the SDE model as:

$$dX_t = -HX_t dt + \sqrt{\eta} H dB_t,$$

which is a Ornstein-Uhlenbeck (OU) process. ⌟

III.2 The Scaling Rule and Non-Gaussian Noise

In the papers [5] and [6], an important concept is about the **scaling rule** for SDEs.

Definition III.2.1. Linear Scaling Rule (Definition 2.1 in [5] and Section 2.2 in [6]).

When the batch size ($|\gamma|$) is scaled by $\kappa > 0$, the learning rate (η) should also be scaled by κ. ⌟

Based on [5], the paper suggests possible non-Gaussian noise in the SGD process. It starts with (3), where we have the higher-dimensional Brownian motion satisfying that:

$$B_t \sim \mathcal{N}(0, \mathrm{Id}).$$

The paper proposes in Section 2.1 of [5] that the validity of SDE approximation is stronger condition (sufficient but not necessary) then Linear scaling rule (LSR) to apply.

In **Example B.1** in [5], the model uses a **Poisson process**, which turns out to be incompatible with the Brownian motion since the gap is non-Gaussian.

Later on in **Section 4** of [5], the paper presents a new algorithm SVAG, which modifies the learning into:

$$\theta^{(t+1)} = \theta^{(k)} = \frac{\eta}{l}\nabla\mathcal{L}^l(\theta^{(t)}; \mathcal{B}_1, \mathcal{B}_2),$$

with some $l \in \mathbb{N}^+$, and we have $\mathcal{B}_1, \mathcal{B}_2$ sampled independently and the loss turns out to be:

$$\mathcal{L}^l(\theta^{(t)}; \mathcal{B}_1, \mathcal{B}_2) = \frac{1+\sqrt{2l-1}}{2}\mathcal{L}(\theta^{(t)}; \mathcal{B}_1) + \frac{1-\sqrt{2l-1}}{2}\mathcal{L}(\theta^{(t)}; \mathcal{B}_2),$$

which is equivalent to doing SGD with two independent samples.

Then, we can also get to the conclusion based on weak convergence.

Theorem III.2.2. Weak Convergence of SVAG (Theorem 4.3 in [5]).

Suppose that the loss function satisfies that:

- $\mathcal{L}$ is $\mathcal{C}^\infty$-smooth and is in G^4.
- $|\nabla\mathcal{L}(x;\mathcal{B}) - \nabla\mathcal{L}(y,\mathcal{B})| \leq L_\mathcal{B}|x-y|$ for all $x, y \in \mathbb{R}^d$, where $L_\mathcal{B}$ is a positive random variable whose expectation of k-th power is bounded.
- $\Sigma^{1/2}(X)$ is $\mathcal{C}^\infty$-smooth.

For fixed $T > 0$ and the hyperparameter l, consider the SDE approximation:

$$dX_t = -\nabla\mathcal{L}(X_t)dt + \left(\eta\Sigma(X_t)\right)^{1/2}dB_t,$$

and the parameters $\{(\theta^{(t)})^{\eta/l}\}_{k=0}^{\langle lT/\eta\rangle}$ for SVAG with $\theta_0 = X_0$, we have SVAG as an order-1 weak approximation of the SDE.

IV Discussion

IV.1 Connection between SDG and SDEs

Through the paper [3], we can see that the SDG process and its variations are, in fact, legit in terms of the machine learning problem. In general, the SDG allows faster computations, but still maintains a relatively good learning expectation over the learning problems. This is an example where mathematicians can verify the empirical results in computer science.

In particular, the SGD conceptually corresponds with the randomness of Brownian motion, and we have seen that both convergences uses the order 1 weak convergence. Especially in modern machine learning and deep learning tasks, there have been more and more versions of SGD and its variations, such as SGD with momentum or ATOM (in [3]), or similar adaption with SVAG (in [5]), the SDEs and stochastic systems turn out to be effective ways of validating these models theoretically when the computer science develops focuses more on performances in practice.

Also, as always to note, the mathematical representation in SDEs is a ideal version. As always for mathematicians, convergence is a good enough thing, but in practice, the concern is also about the rate of convergence since there is no real infinity in actual implementation. Hence, the theoretical proofs would be at best accompanied by some experimental results.

IV.2 Gaussian or Other Types of Noise

From the paper [5], a good question is about randomness. Throughout the SDEs course, the exploration of randomness is by Brownian motion, and in the end, the Giasnov theorem allows a measure conversion between different motions into Brownian motion, but the paper is suggesting some new ways to model, such as using a Poisson process.

In practice, when we are trying to propose more models, there could be a wider class then SDEs to learn about the learning procedure such as the LSR in [5], and it could also be an innovation for the future training models and techniques for machine learning.

References

[1] B. Øksendal, *Stochastic Differential Equations: An Introduction with Applications*, 6th ed. Springer, 2003.

[2] C. S. O. Marc Peter, Deisenroth A. Aldo Faisal, *Mathematics for Machine Learning*. Cambridge University Press, 2020.

[3] W. E. Qianxiao Li, Cheng Tai, "Stochastic modified equations and dynamics of stochastic gradient algorithms i: Mathematical foundations," *Journal of Machine Learning Research*, vol. 20, 2019.

[4] E. P. Peter E. Kleoden, *Numerical Solution of Stochastic Differential Equations*, 3rd ed. Springer, 1999.

[5] Z. Li, S. Malladi, and S. Arora, "On the validity of modeling sgd with stochastic differential equations (sdes)," *Advances in Neural Information Processing Systems*, vol. 34, pp. 12 712–12 725, 2021.

[6] S. Malladi, K. Lyu, A. Panigrahi, and S. Arora, "On the sdes and scaling rules for adaptive gradient algorithms," *Advances in Neural Information Processing Systems*, vol. 35, pp. 7697–7711, 2022.