Chapter 6

Stochastic Approximation

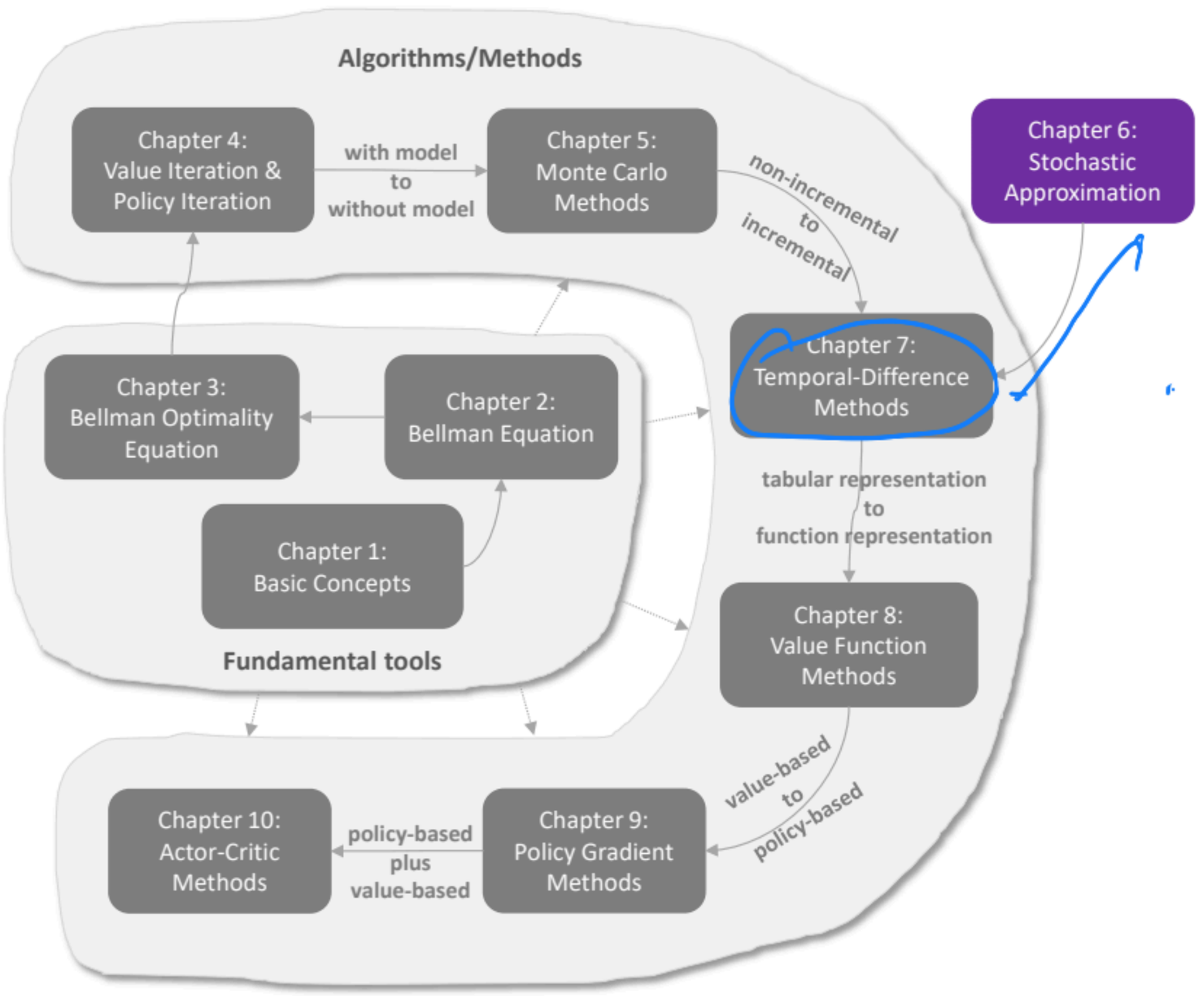

Figure 6.1: Where we are in this book.

Chapter 5 introduced the first class of model-free reinforcement learning algorithms based on Monte Carlo estimation. In the next chapter (Chapter 7), we will introduce another class of model-free reinforcement learning algorithms: temporal-difference learning. However, before proceeding to the next chapter, we need to press the pause button to better prepare ourselves. This is because temporal-difference algorithms are very different from the algorithms that we have studied so far. Many readers who see the temporal-difference algorithms for the first time often wonder how these algorithms were designed in the first place and why they can work effectively. In fact, there is a *knowledge gap* between the previous and subsequent chapters: the algorithms we have studied so far are

non-incremental, but the algorithms that we will study in the subsequent chapters are *incremental*.

We use the present chapter to fill this knowledge gap by introducing the basics of stochastic approximation. Although this chapter does not introduce any specific reinforcement learning algorithms, it lays the necessary foundations for studying subsequent chapters. We will see in Chapter 7 that the temporal-difference algorithms can be viewed as special stochastic approximation algorithms. The well-known stochastic gradient descent algorithms widely used in machine learning are also introduced in the present chapter.

6.1 Motivating example: Mean estimation

We next demonstrate how to convert a non-incremental algorithm to an incremental one by examining the mean estimation problem.

Consider a random variable X that takes values from a finite set $\mathcal{X}$. Our goal is to estimate $\mathbb{E}[X]$. Suppose that we have a sequence of i.i.d. samples $\{x_i\}_{i=1}^n$. The expected value of X can be approximated by

$$\mathbb{E}[X] \approx \bar{x} \doteq \frac{1}{n}\sum_{i=1}^{n} x_i. \tag{6.1}$$

The approximation in (6.1) is the basic idea of Monte Carlo estimation, as introduced in Chapter 5. We know that $\bar{x} \to \mathbb{E}[X]$ as $n \to \infty$ according to the law of large numbers.

We next show that two methods can be used to calculate $\bar{x}$ in (6.1). The first *non-incremental* method collects all the samples first and then calculates the average. The drawback of such a method is that, if the number of samples is large, we may have to wait for a long time until all of the samples are collected. The second method can avoid this drawback because it calculates the average in an *incremental* manner. Specifically, suppose that

$$w_{k+1} \doteq \frac{1}{k}\sum_{i=1}^{k} x_i, \quad k = 1, 2, \ldots$$

and hence

$$w_k = \frac{1}{k-1}\sum_{i=1}^{k-1} x_i, \quad k = 2, 3, \ldots$$

Then, w_{k+1} can be expressed in terms of w_k as

$$w_{k+1} = \frac{1}{k}\sum_{i=1}^{k} x_i = \frac{1}{k}\left(\sum_{i=1}^{k-1} x_i + x_k\right) = \frac{1}{k}((k-1)w_k + x_k) = w_k - \frac{1}{k}(w_k - x_k).$$

Therefore, we obtain the following incremental algorithm:

$$w_{k+1} = w_k - \frac{1}{k}(w_k - x_k). \tag{6.2}$$

This algorithm can be used to calculate the mean $\bar{x}$ in an incremental manner. It can be verified that

$$\begin{aligned}
w_1 &= x_1, \\
w_2 &= w_1 - \frac{1}{1}(w_1 - x_1) = x_1, \\
w_3 &= w_2 - \frac{1}{2}(w_2 - x_2) = x_1 - \frac{1}{2}(x_1 - x_2) = \frac{1}{2}(x_1 + x_2), \\
w_4 &= w_3 - \frac{1}{3}(w_3 - x_3) = \frac{1}{3}(x_1 + x_2 + x_3), \\
&\vdots \\
w_{k+1} &= \frac{1}{k}\sum_{i=1}^{k} x_i.
\end{aligned} \tag{6.3}$$

The advantage of (6.2) is that the average can be immediately calculated every time we receive a sample. This average can be used to approximate $\bar{x}$ and hence $\mathbb{E}[X]$. Notably, the approximation may not be accurate at the beginning due to insufficient samples. However, it is better than nothing. As more samples are obtained, the estimation accuracy can be gradually improved according to the law of large numbers. In addition, one can also define $w_{k+1} = \frac{1}{1+k}\sum_{i=1}^{k+1} x_i$ and $w_k = \frac{1}{k}\sum_{i=1}^{k} x_i$. Doing so would not make any significant difference. In this case, the corresponding iterative algorithm is $w_{k+1} = w_k - \frac{1}{1+k}(w_k - x_{k+1})$.

Furthermore, consider an algorithm with a more general expression:

$$w_{k+1} = w_k - \alpha_k(w_k - x_k). \tag{6.4}$$

This algorithm is important and frequently used in this chapter. It is the same as (6.2) except that the coefficient $1/k$ is replaced by $\alpha_k > 0$. Since the expression of α_k is not given, we are not able to obtain the explicit expression of w_k as in (6.3). However, we will show in the next section that, if $\{\alpha_k\}$ satisfies some mild conditions, $w_k \to \mathbb{E}[X]$ as $k \to \infty$. In Chapter 7, we will see that temporal-difference algorithms have similar (but more complex) expressions.

6.2 Robbins-Monro algorithm

Stochastic approximation refers to a broad class of stochastic iterative algorithms for solving root-finding or optimization problems [24]. Compared to many other root-finding

algorithms such as gradient-based ones, stochastic approximation is powerful in the sense that it does not require the expression of the objective function or its derivative.

The Robbins-Monro (RM) algorithm is a pioneering work in the field of stochastic approximation [24–27]. The famous stochastic gradient descent algorithm is a special form of the RM algorithm, as shown in Section 6.4. We next introduce the details of the RM algorithm.

Suppose that we would like to find the root of the equation

$$g(w) = 0,$$

where $w \in \mathbb{R}$ is the unknown variable and $g : \mathbb{R} \to \mathbb{R}$ is a function. Many problems can be formulated as root-finding problems. For example, if $J(w)$ is an objective function to be optimized, this optimization problem can be converted to solving $g(w) \doteq \nabla_w J(w) = 0$. In addition, an equation such as $g(w) = c$, where c is a constant, can also be converted to the above equation by rewriting $g(w) - c$ as a new function.

If the expression of g or its derivative is known, there are many numerical algorithms that can be used. However, the problem we are facing is that the expression of the function g is unknown. For example, the function may be represented by an artificial neural network whose structure and parameters are unknown. Moreover, we can only obtain a noisy observation of $g(w)$:

$$\tilde{g}(w, \eta) = g(w) + \eta,$$

where $\eta \in \mathbb{R}$ is the observation error, which may or may not be Gaussian. In summary, it is a black-box system where only the input w and the noisy output $\tilde{g}(w, \eta)$ are known (see Figure 6.2). Our aim is to solve $g(w) = 0$ using w and $\tilde{g}$.

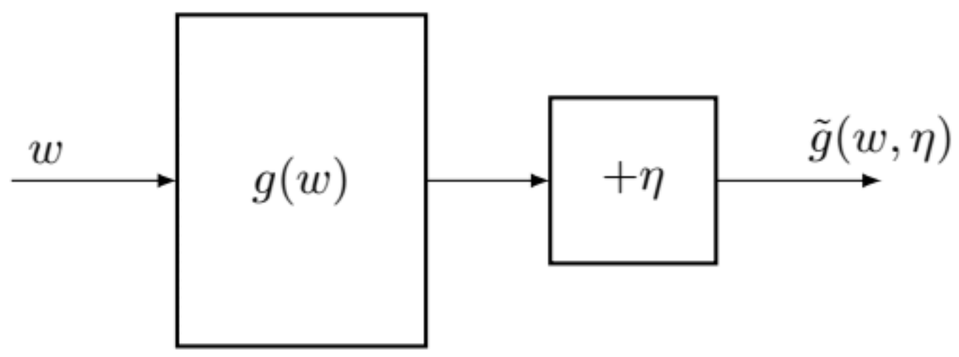

Figure 6.2: An illustration of the problem of solving $g(w) = 0$ from w and $\tilde{g}$.

The RM algorithm that can solve $g(w) = 0$ is

$$w_{k+1} = w_k - a_k \tilde{g}(w_k, \eta_k), \qquad k = 1, 2, 3, \ldots \tag{6.5}$$

where w_k is the kth estimate of the root, $\tilde{g}(w_k, \eta_k)$ is the kth noisy observation, and a_k is a positive coefficient. As can be seen, the RM algorithm does not require any information about the function. It only requires the input and output.

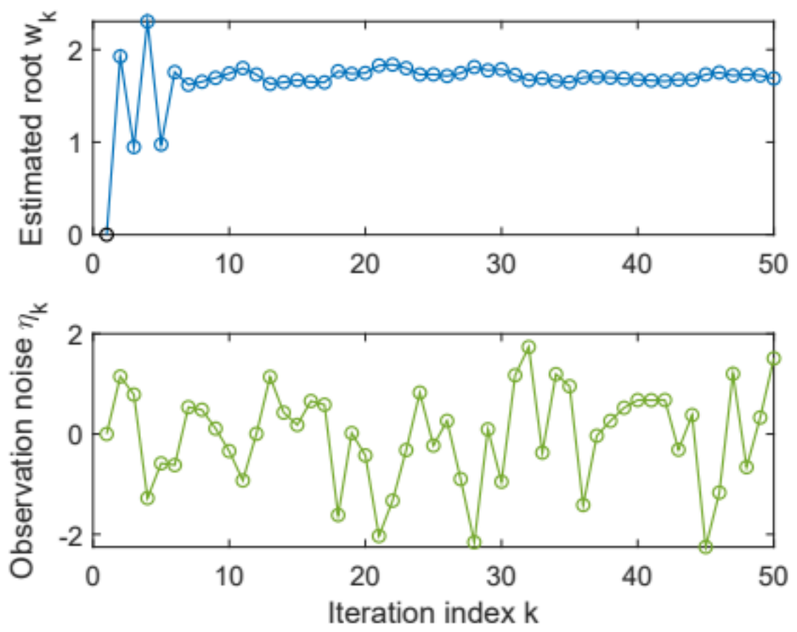

Figure 6.3: An illustrative example of the RM algorithm.

To illustrate the RM algorithm, consider an example in which $g(w) = w^3 - 5$. The true root is $5^{1/3} \approx 1.71$. Now, suppose that we can only observe the input w and the output $\tilde{g}(w) = g(w) + \eta$, where η is i.i.d. and obeys a standard normal distribution with a zero mean and a standard deviation of 1. The initial guess is $w_1 = 0$, and the coefficient is $a_k = 1/k$. The evolution process of w_k is shown in Figure 6.3. Even though the observation is corrupted by noise η_k, the estimate w_k can still converge to the true root. Note that the initial guess w_1 must be properly selected to ensure convergence for the specific function of $g(w) = w^3 - 5$. In the following subsection, we present the conditions under which the RM algorithm converges for any initial guesses.

6.2.1 Convergence properties

Why can the RM algorithm in (6.5) find the root of $g(w) = 0$? We next illustrate the idea with an example and then provide a rigorous convergence analysis.

Consider the example shown in Figure 6.4. In this example, $g(w) = \tanh(w-1)$. The true root of $g(w) = 0$ is $w^* = 1$. We apply the RM algorithm with $w_1 = 3$ and $a_k = 1/k$. To better illustrate the reason for convergence, we simply set $\eta_k \equiv 0$, and consequently, $\tilde{g}(w_k, \eta_k) = g(w_k)$. The RM algorithm in this case is $w_{k+1} = w_k - a_k g(w_k)$. The resulting $\{w_k\}$ generated by the RM algorithm is shown in Figure 6.4. It can be seen that w_k converges to the true root $w^* = 1$.

This simple example can illustrate why the RM algorithm converges.

- ◇ When $w_k > w^*$, we have $g(w_k) > 0$. Then, $w_{k+1} = w_k - a_k g(w_k) < w_k$. If $a_k g(w_k)$ is sufficiently small, we have $w^* < w_{k+1} < w_k$. As a result, w_{k+1} is closer to w^* than w_k.
- ◇ When $w_k < w^*$, we have $g(w_k) < 0$. Then, $w_{k+1} = w_k - a_k g(w_k) > w_k$. If $|a_k g(w_k)|$ is sufficiently small, we have $w^* > w_{k+1} > w_k$. As a result, w_{k+1} is closer to w^* than w_k.

In either case, w_{k+1} is closer to w^* than w_k. Therefore, it is intuitive that w_k converges to w^*.

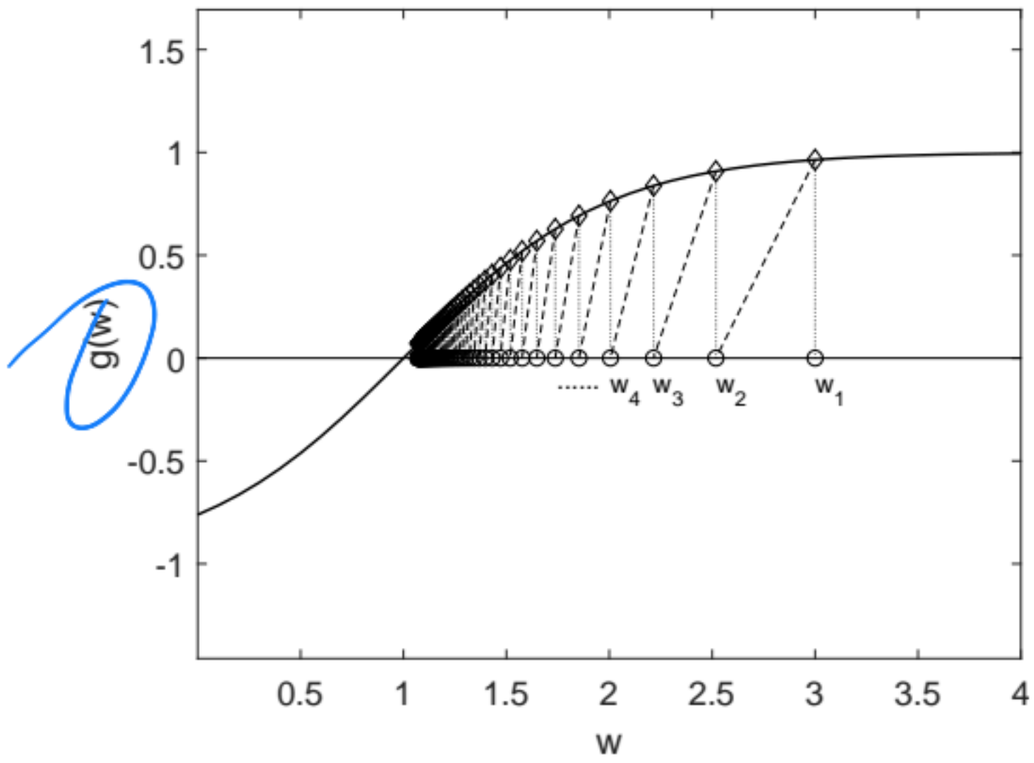

Figure 6.4: An example for illustrating the convergence of the RM algorithm.

The above example is simple since the observation error is assumed to be zero. It would be nontrivial to analyze the convergence in the presence of stochastic observation errors. A rigorous convergence result is given below.

Theorem 6.1 (Robbins-Monro theorem)**.** *In the Robbins-Monro algorithm in* (6.5)*, if*

(a) $0 < c_1 \leq \nabla_w g(w) \leq c_2$ *for all* w*;*

(b) $\sum_{k=1}^{\infty} a_k = \infty$ *and* $\sum_{k=1}^{\infty} a_k^2 < \infty$*;*

(c) $\mathbb{E}[\eta_k | \mathcal{H}_k] = 0$ *and* $\mathbb{E}[\eta_k^2 | \mathcal{H}_k] < \infty$*;*

where $\mathcal{H}_k = \{w_k, w_{k-1}, \dots\}$*, then* w_k *almost surely converges to the root* w^* *satisfying* $g(w^*) = 0$*.*

We postpone the proof of this theorem to Section 6.3.3. This theorem relies on the notion of *almost sure* convergence, which is introduced in Appendix B.

The three conditions in Theorem 6.1 are explained as follows.

- In the first condition, $0 < c_1 \leq \nabla_w g(w)$ indicates that $g(w)$ is a monotonically increasing function. This condition ensures that the root of $g(w) = 0$ exists and is unique. If $g(w)$ is monotonically decreasing, we can simply treat $-g(w)$ as a new function that is monotonically increasing.

  As an application, we can formulate an optimization problem in which the objective function is $J(w)$ as a root-finding problem: $g(w) \doteq \nabla_w J(w) = 0$. In this case, the condition that $g(w)$ is monotonically increasing indicates that $J(w)$ is *convex*, which is a commonly adopted assumption in optimization problems.

  The inequality $\nabla_w g(w) \leq c_2$ indicates that the gradient of $g(w)$ is bounded from above. For example, $g(w) = \tanh(w - 1)$ satisfies this condition, but $g(w) = w^3 - 5$ does not.

◇ The second condition about $\{a_k\}$ is interesting. We often see conditions like this in reinforcement learning algorithms. In particular, the condition $\sum_{k=1}^{\infty} a_k^2 < \infty$ means that $\lim_{n\to\infty} \sum_{k=1}^{n} a_k^2$ is bounded from above. It requires that a_k converges to zero as $k \to \infty$. The condition $\sum_{k=1}^{\infty} a_k = \infty$ means that $\lim_{n\to\infty} \sum_{k=1}^{n} a_k$ is infinitely large. It requires that a_k should not converge to zero too fast. These conditions have interesting properties, which will be analyzed in detail shortly.

◇ The third condition is mild. It does not require the observation error η_k to be Gaussian. An important special case is that $\{\eta_k\}$ is an i.i.d. stochastic sequence satisfying $\mathbb{E}[\eta_k] = 0$ and $\mathbb{E}[\eta_k^2] < \infty$. In this case, the third condition is valid because η_k is independent of $\mathcal{H}_k$ and hence we have $\mathbb{E}[\eta_k|\mathcal{H}_k] = \mathbb{E}[\eta_k] = 0$ and $\mathbb{E}[\eta_k^2|\mathcal{H}_k] = \mathbb{E}[\eta_k^2]$.

We next examine the second condition about the coefficients $\{a_k\}$ more closely.

◇ Why is the second condition important for the convergence of the RM algorithm?

This question can naturally be answered when we present a rigorous proof of the above theorem later. Here, we would like to provide some insightful intuition.

First, $\sum_{k=1}^{\infty} a_k^2 < \infty$ indicates that $a_k \to 0$ as $k \to \infty$. Why is this condition important? Suppose that the observation $\tilde{g}(w_k, \eta_k)$ is always bounded. Since

$$w_{k+1} - w_k = -a_k \tilde{g}(w_k, \eta_k),$$

if $a_k \to 0$, then $a_k \tilde{g}(w_k, \eta_k) \to 0$ and hence $w_{k+1} - w_k \to 0$, indicating that w_{k+1} and w_k approach each other when $k \to \infty$. Otherwise, if a_k does not converge, then w_k may still fluctuate when $k \to \infty$.

Second, $\sum_{k=1}^{\infty} a_k = \infty$ indicates that a_k should not converge to zero too fast. Why is this condition important? Summarizing both sides of the equations of $w_2 - w_1 = -a_1 \tilde{g}(w_1, \eta_1)$, $w_3 - w_2 = -a_2 \tilde{g}(w_2, \eta_2)$, $w_4 - w_3 = -a_3 \tilde{g}(w_3, \eta_3)$, ... gives

$$w_1 - w_\infty = \sum_{k=1}^{\infty} a_k \tilde{g}(w_k, \eta_k).$$

If $\sum_{k=1}^{\infty} a_k < \infty$, then $|\sum_{k=1}^{\infty} a_k \tilde{g}(w_k, \eta_k)|$ is also bounded. Let b denote the finite upper bound such that

$$|w_1 - w_\infty| = \left| \sum_{k=1}^{\infty} a_k \tilde{g}(w_k, \eta_k) \right| \leq b. \tag{6.6}$$

If the initial guess w_1 is selected far away from w^* so that $|w_1 - w^*| > b$, then it is impossible to have $w_\infty = w^*$ according to (6.6). This suggests that the RM algorithm cannot find the true solution w^* in this case. Therefore, the condition $\sum_{k=1}^{\infty} a_k = \infty$ is necessary to ensure convergency given an *arbitrary* initial guess.

◇ What kinds of sequences satisfy $\sum_{k=1}^{\infty} a_k = \infty$ and $\sum_{k=1}^{\infty} a_k^2 < \infty$?

One typical sequence is

$$a_k = \frac{1}{k}.$$

On the one hand, it holds that

$$\lim_{n\to\infty} \left(\sum_{k=1}^{n} \frac{1}{k} - \ln n \right) = \kappa,$$

where $\kappa \approx 0.577$ is called the Euler-Mascheroni constant (or Euler's constant) [28]. Since $\ln n \to \infty$ as $n \to \infty$, we have

$$\sum_{k=1}^{\infty} \frac{1}{k} = \infty.$$

In fact, $H_n = \sum_{k=1}^{n} \frac{1}{k}$ is called the harmonic number in number theory [29]. On the other hand, it holds that

$$\sum_{k=1}^{\infty} \frac{1}{k^2} = \frac{\pi^2}{6} < \infty.$$

Finding the value of $\sum_{k=1}^{\infty} \frac{1}{k^2}$ is known as the Basel problem [30].

In summary, the sequence $\{a_k = 1/k\}$ satisfies the second condition in Theorem 6.1. Notably, a slight modification, such as $a_k = 1/(k+1)$ or $a_k = c_k/k$ where c_k is bounded, also preserves this condition.

In the RM algorithm, a_k is often selected as a sufficiently small *constant* in many applications. Although the second condition is not satisfied anymore in this case because $\sum_{k=1}^{\infty} a_k^2 = \infty$ rather than $\sum_{k=1}^{\infty} a_k^2 < \infty$, the algorithm can still converge in a certain sense [24, Section 1.5]. In addition, $g(x) = x^3 - 5$ in the example shown in Figure 6.3 does not satisfy the first condition, but the RM algorithm can still find the root if the initial guess is adequately (not arbitrarily) selected.

6.2.2 Application to mean estimation

We next apply the Robbins-Monro theorem to analyze the mean estimation problem, which has been discussed in Section 6.1. Recall that

$$w_{k+1} = w_k + \alpha_k(x_k - w_k)$$

is the mean estimation algorithm in (6.4). When $\alpha_k = 1/k$, we can obtain the analytical expression of w_{k+1} as $w_{k+1} = 1/k \sum_{i=1}^{k} x_i$. However, we would not be able to obtain an analytical expression when given general values of α_k. In this case, the convergence analysis is nontrivial. We can show that the algorithm in this case is a special RM

algorithm and hence its convergence naturally follows.

In particular, define a function as

$$g(w) \doteq w - \mathbb{E}[X].$$

The original problem is to obtain the value of $\mathbb{E}[X]$. This problem is formulated as a root-finding problem to solve $g(w) = 0$. Given a value of w, the noisy observation that we can obtain is $\tilde{g} \doteq w - x$, where x is a sample of X. Note that $\tilde{g}$ can be written as

$$\begin{aligned}\tilde{g}(w, \eta) &= w - x \\ &= w - x + \mathbb{E}[X] - \mathbb{E}[X] \\ &= (w - \mathbb{E}[X]) + (\mathbb{E}[X] - x) \doteq g(w) + \eta,\end{aligned}$$

where $\eta \doteq \mathbb{E}[X] - x$.

The RM algorithm for solving this problem is

$$w_{k+1} = w_k - \alpha_k \tilde{g}(w_k, \eta_k) = w_k - \alpha_k(w_k - x_k),$$

which is exactly the algorithm in (6.4). As a result, it is guaranteed by Theorem 6.1 that w_k converges to $\mathbb{E}[X]$ almost surely if $\sum_{k=1}^{\infty} \alpha_k = \infty$, $\sum_{k=1}^{\infty} \alpha_k^2 < \infty$, and $\{x_k\}$ is i.i.d. It is worth mentioning that the convergence property does not rely on any assumption regarding the distribution of X.

6.3 Dvoretzky's convergence theorem

Until now, the convergence of the RM algorithm has not yet been proven. To do that, we next introduce Dvoretzky's theorem [31, 32], which is a classic result in the field of stochastic approximation. This theorem can be used to analyze the convergence of the RM algorithm and many reinforcement learning algorithms.

This section is slightly mathematically intensive. Readers who are interested in the convergence analyses of stochastic algorithms are recommended to study this section. Otherwise, this section can be skipped.

Theorem 6.2 (Dvoretzky's theorem)**.** *Consider a stochastic process*

$$\Delta_{k+1} = (1 - \alpha_k)\Delta_k + \beta_k \eta_k,$$

where $\{\alpha_k\}_{k=1}^{\infty}, \{\beta_k\}_{k=1}^{\infty}, \{\eta_k\}_{k=1}^{\infty}$ *are stochastic sequences. Here* $\alpha_k \geq 0, \beta_k \geq 0$ *for all* k*. Then,* Δ_k *converges to zero almost surely if the following conditions are satisfied:*

(a) $\sum_{k=1}^{\infty} \alpha_k = \infty$, $\sum_{k=1}^{\infty} \alpha_k^2 < \infty$, *and* $\sum_{k=1}^{\infty} \beta_k^2 < \infty$ *uniformly almost surely;*

(b) $\mathbb{E}[\eta_k|\mathcal{H}_k] = 0$ *and* $\mathbb{E}[\eta_k^2|\mathcal{H}_k] \leq C$ *almost surely;*

where $\mathcal{H}_k = \{\Delta_k, \Delta_{k-1}, \ldots, \eta_{k-1}, \ldots, \alpha_{k-1}, \ldots, \beta_{k-1}, \ldots\}$.

Before presenting the proof of this theorem, we first clarify some issues.

- ◇ In the RM algorithm, the coefficient sequence $\{\alpha_k\}$ is deterministic. However, Dvoretzky's theorem allows $\{\alpha_k\}, \{\beta_k\}$ to be random variables that depend on $\mathcal{H}_k$. Thus, it is more useful in cases where α_k or β_k is a function of Δ_k.
- ◇ In the first condition, it is stated as "uniformly almost surely". This is because α_k and β_k may be random variables and hence the definition of their limits must be in the stochastic sense. In the second condition, it is also stated as "almost surely". This is because $\mathcal{H}_k$ is a sequence of random variables rather than specific values. As a result, $\mathbb{E}[\eta_k|\mathcal{H}_k]$ and $\mathbb{E}[\eta_k^2|\mathcal{H}_k]$ are random variables. The definition of the conditional expectation in this case is in the "almost sure" sense (Appendix B).
- ◇ The statement of Theorem 6.2 is slightly different from [32] in the sense that Theorem 6.2 does not require $\sum_{k=1}^{\infty} \beta_k = \infty$ in the first condition. When $\sum_{k=1}^{\infty} \beta_k < \infty$, especially in the extreme case where $\beta_k = 0$ for all k, the sequence can still converge.

6.3.1 Proof of Dvoretzky's theorem

The original proof of Dvoretzky's theorem was given in 1956 [31]. There are also other proofs. We next present a proof based on quasimartingales. With the convergence results of quasimartingales, the proof of Dvoretzky's theorem is straightforward. More information about quasimartingales can be found in Appendix C.

Proof of Dvoretzky's theorem. Let $h_k \doteq \Delta_k^2$. Then,

$$\begin{aligned} h_{k+1} - h_k &= \Delta_{k+1}^2 - \Delta_k^2 \\ &= (\Delta_{k+1} - \Delta_k)(\Delta_{k+1} + \Delta_k) \\ &= (-\alpha_k \Delta_k + \beta_k \eta_k)[(2 - \alpha_k)\Delta_k + \beta_k \eta_k] \\ &= -\alpha_k(2 - \alpha_k)\Delta_k^2 + \beta_k^2 \eta_k^2 + 2(1 - \alpha_k)\beta_k \eta_k \Delta_k. \end{aligned}$$

Taking expectations on both sides of the above equation yields

$$\mathbb{E}[h_{k+1} - h_k|\mathcal{H}_k] = \mathbb{E}[-\alpha_k(2 - \alpha_k)\Delta_k^2|\mathcal{H}_k] + \mathbb{E}[\beta_k^2 \eta_k^2|\mathcal{H}_k] + \mathbb{E}[2(1 - \alpha_k)\beta_k \eta_k \Delta_k|\mathcal{H}_k]. \tag{6.7}$$

First, since Δ_k is included and hence determined by $\mathcal{H}_k$, it can be taken out from the expectation (see property (e) in Lemma B.1). Second, consider the simple case

where α_k, β_k is determined by $\mathcal{H}_k$. This case is valid when, for example, $\{\alpha_k\}$ and $\{\beta_k\}$ are functions of Δ_k or deterministic sequences. Then, they can also be taken out of the expectation. Therefore, (6.7) becomes

$$\mathbb{E}[h_{k+1} - h_k|\mathcal{H}_k] = -\alpha_k(2-\alpha_k)\Delta_k^2 + \beta_k^2\mathbb{E}[\eta_k^2|\mathcal{H}_k] + 2(1-\alpha_k)\beta_k\Delta_k\mathbb{E}[\eta_k|\mathcal{H}_k]. \quad (6.8)$$

For the first term, since $\sum_{k=1}^{\infty}\alpha_k^2 < \infty$ implies $\alpha_k \to 0$ almost surely, there exists a finite n such that $\alpha_k \leq 1$ almost surely for all $k \geq n$. Without loss of generality, we next merely consider the case of $\alpha_k \leq 1$. Then, $-\alpha_k(2-\alpha_k)\Delta_k^2 \leq 0$. For the second term, we have $\beta_k^2\mathbb{E}[\eta_k^2|\mathcal{H}_k] \leq \beta_k^2 C$ as assumed. The third term equals zero because $\mathbb{E}[\eta_k|\mathcal{H}_k] = 0$ as assumed. Therefore, (6.8) becomes

$$\mathbb{E}[h_{k+1} - h_k|\mathcal{H}_k] = -\alpha_k(2-\alpha_k)\Delta_k^2 + \beta_k^2\mathbb{E}[\eta_k^2|\mathcal{H}_k] \leq \beta_k^2 C, \quad (6.9)$$

and hence

$$\sum_{k=1}^{\infty}\mathbb{E}[h_{k+1} - h_k|\mathcal{H}_k] \leq \sum_{k=1}^{\infty}\beta_k^2 C < \infty.$$

The last inequality is due to the condition $\sum_{k=1}^{\infty}\beta_k^2 < \infty$. Then, based on the quasimartingale convergence theorem in Appendix C, we conclude that h_k converges almost surely.

We next determine what value Δ_k converges to. It follows from (6.9) that

$$\sum_{k=1}^{\infty}\alpha_k(2-\alpha_k)\Delta_k^2 = \sum_{k=1}^{\infty}\beta_k^2\mathbb{E}[\eta_k^2|\mathcal{H}_k] - \sum_{k=1}^{\infty}\mathbb{E}[h_{k+1} - h_k|\mathcal{H}_k].$$

The first term on the right-hand side is bounded as assumed. The second term is also bounded because h_k converges and hence $h_{k+1} - h_k$ is summable. Thus, $\sum_{k=1}^{\infty}\alpha_k(2-\alpha_k)\Delta_k^2$ on the left-hand side is also bounded. Since we consider the case of $\alpha_k \leq 1$, we have

$$\infty > \sum_{k=1}^{\infty}\alpha_k(2-\alpha_k)\Delta_k^2 \geq \sum_{k=1}^{\infty}\alpha_k\Delta_k^2 \geq 0.$$

Therefore, $\sum_{k=1}^{\infty}\alpha_k\Delta_k^2$ is bounded. Since $\sum_{k=1}^{\infty}\alpha_k = \infty$, we must have $\Delta_k \to 0$ almost surely. □

6.3.2 Application to mean estimation

While the mean estimation algorithm, $w_{k+1} = w_k + \alpha_k(x_k - w_k)$, has been analyzed using the RM theorem, we next show that its convergence can also be directly proven by Dvoretzky's theorem.

Proof. Let $w^* = \mathbb{E}[X]$. The mean estimation algorithm $w_{k+1} = w_k + \alpha_k(x_k - w_k)$ can be rewritten as

$$w_{k+1} - w^* = w_k - w^* + \alpha_k(x_k - w^* + w^* - w_k).$$

Let $\Delta \doteq w - w^*$. Then, we have

$$\begin{aligned}\Delta_{k+1} &= \Delta_k + \alpha_k(x_k - w^* - \Delta_k) \\ &= (1 - \alpha_k)\Delta_k + \alpha_k \underbrace{(x_k - w^*)}_{\eta_k}.\end{aligned}$$

Since $\{x_k\}$ is i.i.d., we have $\mathbb{E}[x_k|\mathcal{H}_k] = \mathbb{E}[x_k] = w^*$. As a result, $\mathbb{E}[\eta_k|\mathcal{H}_k] = \mathbb{E}[x_k - w^*|\mathcal{H}_k] = 0$ and $\mathbb{E}[\eta_k^2|\mathcal{H}_k] = \mathbb{E}[x_k^2|\mathcal{H}_k] - (w^*)^2 = \mathbb{E}[x_k^2] - (w^*)^2$ are bounded if the variance of x_k is finite. Following Dvoretzky's theorem, we conclude that Δ_k converges to zero and hence w_k converges to $w^* = \mathbb{E}[X]$ almost surely. □

6.3.3 Application to the Robbins-Monro theorem

We are now ready to prove the Robbins-Monro theorem using Dvoretzky's theorem.

Proof of the Robbins-Monro theorem. The RM algorithm aims to find the root of $g(w) = 0$. Suppose that the root is w^* such that $g(w^*) = 0$. The RM algorithm is

$$\begin{aligned}w_{k+1} &= w_k - a_k\tilde{g}(w_k, \eta_k) \\ &= w_k - a_k[g(w_k) + \eta_k].\end{aligned}$$

Then, we have

$$w_{k+1} - w^* = w_k - w^* - a_k[g(w_k) - g(w^*) + \eta_k].$$

Due to the mean value theorem [7, 8], we have $g(w_k) - g(w^*) = \nabla_w g(w'_k)(w_k - w^*)$,

where $w_k' \in [w_k, w^*]$. Let $\Delta_k \doteq w_k - w^*$. The above equation becomes

$$\begin{aligned}\Delta_{k+1} &= \Delta_k - a_k[\nabla_w g(w_k')(w_k - w^*) + \eta_k] \\ &= \Delta_k - a_k \nabla_w g(w_k')\Delta_k + a_k(-\eta_k) \\ &= [1 - \underbrace{a_k \nabla_w g(w_k')}_{\alpha_k}]\Delta_k + a_k(-\eta_k).\end{aligned}$$

Note that $\nabla_w g(w)$ is bounded as $0 < c_1 \leq \nabla_w g(w) \leq c_2$ as assumed. Since $\sum_{k=1}^{\infty} a_k = \infty$ and $\sum_{k=1}^{\infty} a_k^2 < \infty$ as assumed, we know $\sum_{k=1}^{\infty} \alpha_k = \infty$ and $\sum_{k=1}^{\infty} \alpha_k^2 < \infty$. Thus, all the conditions in Dvoretzky's theorem are satisfied and hence Δ_k converges to zero almost surely. □

The proof of the RM theorem demonstrates the power of Dvoretzky's theorem. In particular, α_k in the proof is a stochastic sequence depending on w_k rather than a deterministic sequence. In this case, Dvoretzky's theorem is still applicable.

6.3.4 An extension of Dvoretzky's theorem

We next extend Dvoretzky's theorem to a more general theorem that can handle *multiple* variables. This general theorem, proposed by [32], can be used to analyze the convergence of stochastic iterative algorithms such as Q-learning.

Theorem 6.3. *Consider a finite set $\mathcal{S}$ of real numbers. For the stochastic process*

$$\Delta_{k+1}(s) = (1 - \alpha_k(s))\Delta_k(s) + \beta_k(s)\eta_k(s),$$

it holds that $\Delta_k(s)$ converges to zero almost surely for every $s \in \mathcal{S}$ if the following conditions are satisfied for $s \in \mathcal{S}$:

(a) $\sum_k \alpha_k(s) = \infty$, $\sum_k \alpha_k^2(s) < \infty$, $\sum_k \beta_k^2(s) < \infty$, *and* $\mathbb{E}[\beta_k(s)|\mathcal{H}_k] \leq \mathbb{E}[\alpha_k(s)|\mathcal{H}_k]$ *uniformly almost surely;*

(b) $\|\mathbb{E}[\eta_k(s)|\mathcal{H}_k]\|_\infty \leq \gamma \|\Delta_k\|_\infty$, *where* $\gamma \in (0, 1)$;

(c) $\mathrm{var}[\eta_k(s)|\mathcal{H}_k] \leq C(1 + \|\Delta_k(s)\|_\infty)^2$, *where C is a constant.*

Here, $\mathcal{H}_k = \{\Delta_k, \Delta_{k-1}, \ldots, \eta_{k-1}, \ldots, \alpha_{k-1}, \ldots, \beta_{k-1}, \ldots\}$ represents the historical information. The term $\|\cdot\|_\infty$ refers to the maximum norm.

Proof. As an extension, this theorem can be proven based on Dvoretzky's theorem. Details can be found in [32] and are omitted here. □

Some remarks about this theorem are given below.

- ◇ We first clarify some notations in the theorem. The variable s can be viewed as an index. In the context of reinforcement learning, it indicates a state or a state-action pair. The *maximum norm* $\|\cdot\|_\infty$ is defined over a set. It is similar but different from the L^∞ norm of vectors. In particular, $\|\mathbb{E}[\eta_k(s)|\mathcal{H}_k]\|_\infty \doteq \max_{s\in\mathcal{S}} |\mathbb{E}[\eta_k(s)|\mathcal{H}_k]|$ and $\|\Delta_k(s)\|_\infty \doteq \max_{s\in\mathcal{S}} |\Delta_k(s)|$.
- ◇ This theorem is more general than Dvoretzky's theorem. First, it can handle the case of multiple variables due to the maximum norm operations. This is important for a reinforcement learning problem where there are multiple states. Second, while Dvoretzky's theorem requires $\mathbb{E}[\eta_k(s)|\mathcal{H}_k] = 0$ and $\text{var}[\eta_k(s)|\mathcal{H}_k] \leq C$, this theorem only requires that the expectation and variance are bounded by the error Δ_k.
- ◇ It should be noted that the convergence of $\Delta(s)$ for all $s \in \mathcal{S}$ requires that the conditions are valid for every $s \in \mathcal{S}$. Therefore, when applying this theorem to prove the convergence of reinforcement learning algorithms, we need to show that the conditions are valid for every state (or state-action pair).

6.4 Stochastic gradient descent

This section introduces stochastic gradient descent (SGD) algorithms, which are widely used in the field of machine learning. We will see that SGD is a special RM algorithm, and the mean estimation algorithm is a special SGD algorithm.

Consider the following optimization problem:

$$\min_w J(w) = \mathbb{E}[f(w, X)], \tag{6.10}$$

where w is the parameter to be optimized, and X is a random variable. The expectation is calculated with respect to X. Here, w and X can be either scalars or vectors. The function $f(\cdot)$ is a scalar.

A straightforward method for solving (6.10) is *gradient descent*. In particular, the gradient of $\mathbb{E}[f(w, X)]$ is $\nabla_w \mathbb{E}[f(w, X)] = \mathbb{E}[\nabla_w f(w, X)]$. Then, the gradient descent algorithm is

$$w_{k+1} = w_k - \alpha_k \nabla_w J(w_k) = w_k - \alpha_k \mathbb{E}[\nabla_w f(w_k, X)]. \tag{6.11}$$

This gradient descent algorithm can find the optimal solution w^* under some mild conditions such as the convexity of f. Preliminaries about gradient descent algorithms can be found in Appendix D.

The gradient descent algorithm requires the expected value $\mathbb{E}[\nabla_w f(w_k, X)]$. One way to obtain the expected value is based on the probability distribution of X. The

distribution is, however, often unknown in practice. Another way is to collect a large number of i.i.d. samples $\{x_i\}_{i=1}^n$ of X so that the expected value can be approximated as

$$\mathbb{E}[\nabla_w f(w_k, X)] \approx \frac{1}{n}\sum_{i=1}^{n} \nabla_w f(w_k, x_i).$$

Then, (6.11) becomes

$$w_{k+1} = w_k - \frac{\alpha_k}{n}\sum_{i=1}^{n} \nabla_w f(w_k, x_i). \tag{6.12}$$

One problem of the algorithm in (6.12) is that it requires all the samples in each iteration. In practice, if the samples are collected one by one, then it is favorable to update w every time a sample is collected. To that end, we can use the following algorithm:

$$w_{k+1} = w_k - \alpha_k \nabla_w f(w_k, x_k), \tag{6.13}$$

where x_k is the sample collected at time step k. This is the well-known *stochastic gradient descent* algorithm. This algorithm is called "stochastic" because it relies on stochastic samples $\{x_k\}$.

Compared to the gradient descent algorithm in (6.11), SGD replaces the true gradient $\mathbb{E}[\nabla_w f(w, X)]$ with the *stochastic gradient* $\nabla_w f(w_k, x_k)$. Since $\nabla_w f(w_k, x_k) \neq \mathbb{E}[\nabla_w f(w, X)]$, can such a replacement still ensure $w_k \to w^*$ as $k \to \infty$? The answer is yes. We next present an intuitive explanation and postpone the rigorous proof of the convergence to Section 6.4.5.

In particular, since

$$\begin{aligned}\nabla_w f(w_k, x_k) &= \mathbb{E}[\nabla_w f(w_k, X)] + \Big(\nabla_w f(w_k, x_k) - \mathbb{E}[\nabla_w f(w_k, X)]\Big) \\ &\doteq \mathbb{E}[\nabla_w f(w_k, X)] + \eta_k,\end{aligned}$$

the SGD algorithm in (6.13) can be rewritten as

$$w_{k+1} = w_k - \alpha_k \mathbb{E}[\nabla_w f(w_k, X)] - \alpha_k \eta_k.$$

Therefore, the SGD algorithm is the same as the regular gradient descent algorithm except that it has a perturbation term $\alpha_k \eta_k$. Since $\{x_k\}$ is i.i.d., we have $\mathbb{E}_{x_k}[\nabla_w f(w_k, x_k)] = \mathbb{E}_X[\nabla_w f(w_k, X)]$. As a result,

$$\mathbb{E}[\eta_k] = \mathbb{E}\Big[\nabla_w f(w_k, x_k) - \mathbb{E}[\nabla_w f(w_k, X)]\Big] = \mathbb{E}_{x_k}[\nabla_w f(w_k, x_k)] - \mathbb{E}_X[\nabla_w f(w_k, X)] = 0.$$

Therefore, the perturbation term η_k has a zero mean, which intuitively suggests that it may not jeopardize the convergence property. A rigorous proof of the convergence of

SGD is given in Section 6.4.5.

6.4.1 Application to mean estimation

We next apply SGD to analyze the mean estimation problem and show that the mean estimation algorithm in (6.4) is a special SGD algorithm. To that end, we formulate the mean estimation problem as an optimization problem:

$$\min_w J(w) = \mathbb{E}\left[\frac{1}{2}\|w - X\|^2\right] \doteq \mathbb{E}[f(w, X)], \tag{6.14}$$

where $f(w, X) = \|w - X\|^2/2$ and the gradient is $\nabla_w f(w, X) = w - X$. It can be verified that the optimal solution is $w^* = \mathbb{E}[X]$ by solving $\nabla_w J(w) = 0$. Therefore, this optimization problem is equivalent to the mean estimation problem.

◇ The gradient descent algorithm for solving (6.14) is

$$\begin{aligned} w_{k+1} &= w_k - \alpha_k \nabla_w J(w_k) \\ &= w_k - \alpha_k \mathbb{E}[\nabla_w f(w_k, X)] \\ &= w_k - \alpha_k \mathbb{E}[w_k - X]. \end{aligned}$$

This gradient descent algorithm is not applicable since $\mathbb{E}[w_k - X]$ or $\mathbb{E}[X]$ on the right-hand side is unknown (in fact, it is what we need to solve).

◇ The SGD algorithm for solving (6.14) is

$$w_{k+1} = w_k - \alpha_k \nabla_w f(w_k, x_k) = w_k - \alpha_k (w_k - x_k),$$

where x_k is a sample obtained at time step k. Notably, this SGD algorithm is the same as the iterative mean estimation algorithm in (6.4). Therefore, (6.4) is an SGD algorithm designed specifically for solving the mean estimation problem.

6.4.2 Convergence pattern of SGD

The idea of the SGD algorithm is to replace the true gradient with a stochastic gradient. However, since the stochastic gradient is random, one may ask whether the convergence speed of SGD is slow or random. Fortunately, SGD can converge efficiently in general. An interesting *convergence pattern* is that it behaves similarly to the regular gradient descent algorithm when the estimate w_k is far from the optimal solution w^*. Only when w_k is close to w^*, does the convergence of SGD exhibit more randomness.

An analysis of this pattern and an illustrative example are given below.

◇ Analysis: The *relative error* between the stochastic and true gradients is

$$\delta_k \doteq \frac{|\nabla_w f(w_k, x_k) - \mathbb{E}[\nabla_w f(w_k, X)]|}{|\mathbb{E}[\nabla_w f(w_k, X)]|}.$$

For the sake of simplicity, we consider the case where w and $\nabla_w f(w, x)$ are both *scalars*. Since w^* is the optimal solution, it holds that $\mathbb{E}[\nabla_w f(w^*, X)] = 0$. Then, the relative error can be rewritten as

$$\delta_k = \frac{|\nabla_w f(w_k, x_k) - \mathbb{E}[\nabla_w f(w_k, X)]|}{|\mathbb{E}[\nabla_w f(w_k, X)] - \mathbb{E}[\nabla_w f(w^*, X)]|} = \frac{|\nabla_w f(w_k, x_k) - \mathbb{E}[\nabla_w f(w_k, X)]|}{|\mathbb{E}[\nabla_w^2 f(\tilde{w}_k, X)(w_k - w^*)]|}, \quad (6.15)$$

where the last equality is due to the mean value theorem [7, 8] and $\tilde{w}_k \in [w_k, w^*]$. Suppose that f is strictly convex such that $\nabla_w^2 f \geq c > 0$ for all w, X. Then, the denominator in (6.15) becomes

$$\begin{aligned} \big|\mathbb{E}[\nabla_w^2 f(\tilde{w}_k, X)(w_k - w^*)]\big| &= \big|\mathbb{E}[\nabla_w^2 f(\tilde{w}_k, X)]\big|\big|(w_k - w^*)\big| \\ &\geq c|w_k - w^*|. \end{aligned}$$

Substituting the above inequality into (6.15) yields

$$\delta_k \leq \frac{\big|\overbrace{\nabla_w f(w_k, x_k)}^{\text{stochastic gradient}} - \overbrace{\mathbb{E}[\nabla_w f(w_k, X)]}^{\text{true gradient}}\big|}{\underbrace{c|w_k - w^*|}_{\text{distance to the optimal solution}}}.$$

The above inequality suggests an interesting convergence pattern of SGD: the relative error δ_k is inversely proportional to $|w_k - w^*|$. As a result, when $|w_k - w^*|$ is large, δ_k is small. In this case, the SGD algorithm behaves like the gradient descent algorithm and hence w_k quickly converges to w^*. When w_k is close to w^*, the relative error δ_k may be large, and the convergence exhibits more randomness.

◇ Example: A good example for demonstrating the above analysis is the mean estimation problem. Consider the mean estimation problem in (6.14). When w and X are both scalar, we have $f(w, X) = |w - X|^2/2$ and hence

$$\begin{aligned} \nabla_w f(w, x_k) &= w - x_k, \\ \mathbb{E}[\nabla_w f(w, x_k)] &= w - \mathbb{E}[X] = w - w^*. \end{aligned}$$

Thus, the relative error is

$$\delta_k = \frac{|\nabla_w f(w_k, x_k) - \mathbb{E}[\nabla_w f(w_k, X)]|}{|\mathbb{E}[\nabla_w f(w_k, X)]|} = \frac{|(w_k - x_k) - (w_k - \mathbb{E}[X])|}{|w_k - w^*|} = \frac{|\mathbb{E}[X] - x_k|}{|w_k - w^*|}.$$

The expression of the relative error clearly shows that δ_k is *inversely proportional* to

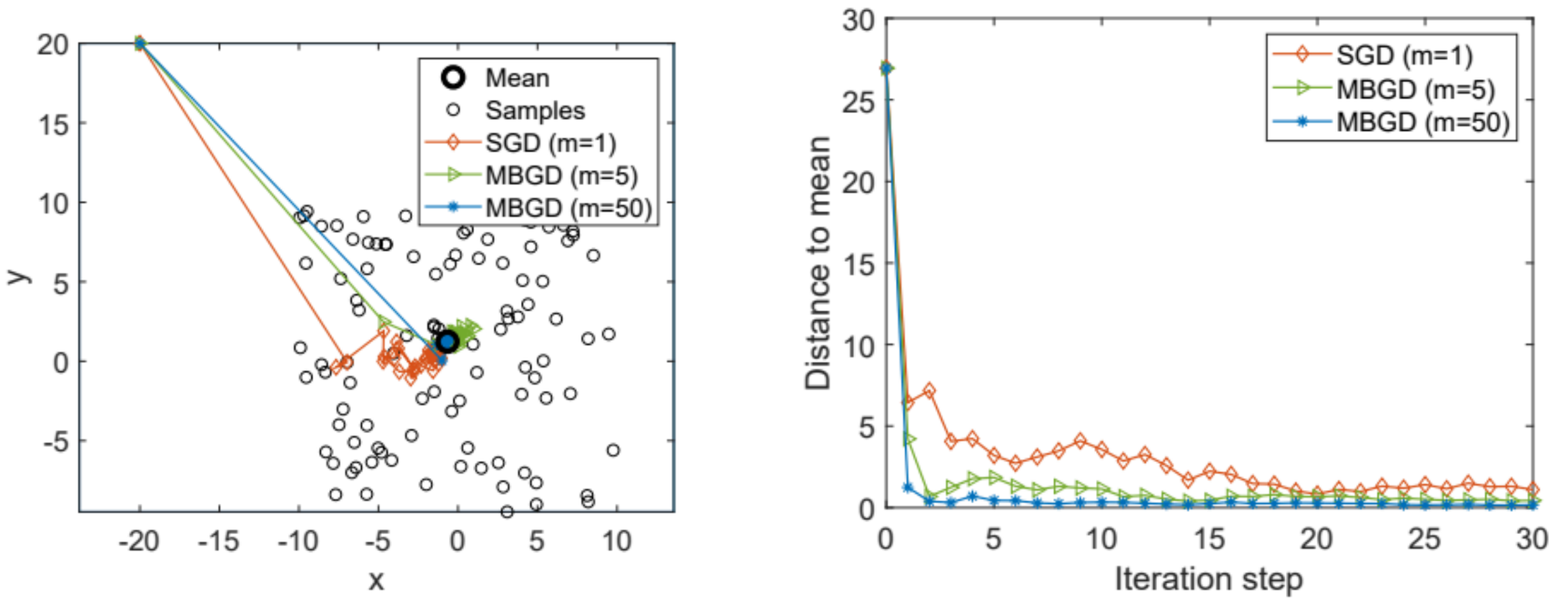

Figure 6.5: An example for demonstrating stochastic and mini-batch gradient descent algorithms. The distribution of $X \in \mathbb{R}^2$ is uniform in the square area centered at the origin with a side length as 20. The mean is $\mathbb{E}[X] = 0$. The mean estimation is based on 100 i.i.d. samples.

$|w_k - w^*|$. As a result, when w_k is far from w^*, the relative error is small, and SGD behaves like gradient descent. In addition, since δ_k is proportional to $|\mathbb{E}[X] - x_k|$, the mean of δ_k is proportional to the variance of X.

The simulation results are shown in Figure 6.5. Here, $X \in \mathbb{R}^2$ represents a random position in the plane. Its distribution is uniform in the square area centered at the origin and $\mathbb{E}[X] = 0$. The mean estimation is based on 100 i.i.d. samples. Although the initial guess of the mean is far away from the true value, it can be seen that the SGD estimate quickly approaches the neighborhood of the origin. When the estimate is close to the origin, the convergence process exhibits certain randomness.

6.4.3 A deterministic formulation of SGD

The formulation of SGD in (6.13) involves random variables. One may often encounter a deterministic formulation of SGD without involving any random variables.

In particular, consider a set of real numbers $\{x_i\}_{i=1}^n$, where x_i does not have to be a sample of any random variable. The optimization problem to be solved is to minimize the average:

$$\min_w J(w) = \frac{1}{n} \sum_{i=1}^{n} f(w, x_i),$$

where $f(w, x_i)$ is a parameterized function, and w is the parameter to be optimized. The gradient descent algorithm for solving this problem is

$$w_{k+1} = w_k - \alpha_k \nabla_w J(w_k) = w_k - \alpha_k \frac{1}{n} \sum_{i=1}^{n} \nabla_w f(w_k, x_i).$$

Suppose that the set $\{x_i\}_{i=1}^n$ is large and we can only fetch a single number each time.

In this case, it is favorable to update w_k in an incremental manner:

$$w_{k+1} = w_k - \alpha_k \nabla_w f(w_k, x_k). \tag{6.16}$$

It must be noted that x_k here is the number fetched at time step k instead of the kth element in the set $\{x_i\}_{i=1}^n$.

The algorithm in (6.16) is very similar to SGD, but its problem formulation is subtly different because it does not involve any random variables or expected values. Then, many questions arise. For example, is this algorithm SGD? How should we use the finite set of numbers $\{x_i\}_{i=1}^n$? Should we sort these numbers in a certain order and then use them one by one, or should we randomly sample a number from the set?

A quick answer to the above questions is that, although no random variables are involved in the above formulation, we can convert the *deterministic formulation* to the *stochastic formulation* by introducing a random variable. In particular, let X be a random variable defined on the set $\{x_i\}_{i=1}^n$. Suppose that its probability distribution is uniform such that $p(X = x_i) = 1/n$. Then, the deterministic optimization problem becomes a stochastic one:

$$\min_w J(w) = \frac{1}{n} \sum_{i=1}^{n} f(w, x_i) = \mathbb{E}[f(w, X)].$$

The last equality in the above equation is strict instead of approximate. Therefore, the algorithm in (6.16) is SGD, and the estimate converges if x_k is *uniformly* and independently sampled from $\{x_i\}_{i=1}^n$. Note that x_k may repeatedly take the same number in $\{x_i\}_{i=1}^n$ since it is sampled randomly.

6.4.4 BGD, SGD, and mini-batch GD

While SGD uses a single sample in every iteration, we next introduce *mini-batch gradient descent* (MBGD), which uses a few more samples in every iteration. When all samples are used in every iteration, the algorithm is called *batch gradient descent* (BGD).

In particular, suppose that we would like to find the optimal solution that can minimize $J(w) = \mathbb{E}[f(w, X)]$ given a set of random samples $\{x_i\}_{i=1}^n$ of X. The BGD, SGD, and MBGD algorithms for solving this problem are, respectively,

$$w_{k+1} = w_k - \alpha_k \frac{1}{n} \sum_{i=1}^{n} \nabla_w f(w_k, x_i), \qquad \text{(BGD)}$$

$$w_{k+1} = w_k - \alpha_k \frac{1}{m} \sum_{j \in \mathcal{I}_k} \nabla_w f(w_k, x_j), \qquad \text{(MBGD)}$$

$$w_{k+1} = w_k - \alpha_k \nabla_w f(w_k, x_k). \qquad \text{(SGD)}$$

In the BGD algorithm, all the samples are used in every iteration. When n is large, $(1/n) \sum_{i=1}^n \nabla_w f(w_k, x_i)$ is close to the true gradient $\mathbb{E}[\nabla_w f(w_k, X)]$. In the MBGD al-

gorithm, $\mathcal{I}_k$ is a subset of $\{1, \ldots, n\}$ obtained at time k. The size of the set is $|\mathcal{I}_k| = m$. The samples in $\mathcal{I}_k$ are also assumed to be i.i.d. In the SGD algorithm, x_k is randomly sampled from $\{x_i\}_{i=1}^n$ at time k.

MBGD can be viewed as an intermediate version between SGD and BGD. Compared to SGD, MBGD has less randomness because it uses more samples instead of just one as in SGD. Compared to BGD, MBGD does not require using all the samples in every iteration, making it more flexible. If $m = 1$, then MBGD becomes SGD. However, if $m = n$, MBGD may *not* become BGD. This is because MBGD uses n randomly fetched samples, whereas BGD uses all n numbers. These n randomly fetched samples may contain the same number multiple times and hence may not cover all n numbers in $\{x_i\}_{i=1}^n$.

The convergence speed of MBGD is faster than that of SGD in general. This is because SGD uses $\nabla_w f(w_k, x_k)$ to approximate the true gradient, whereas MBGD uses $(1/m)\sum_{j\in\mathcal{I}_k} \nabla_w f(w_k, x_j)$, which is closer to the true gradient because the randomness is averaged out. The convergence of the MBGD algorithm can be proven similarly to the SGD case.

A good example for demonstrating the above analysis is the mean estimation problem. In particular, given some numbers $\{x_i\}_{i=1}^n$, our goal is to calculate the mean $\bar{x} = \sum_{i=1}^n x_i/n$. This problem can be equivalently stated as the following optimization problem:

$$\min_w J(w) = \frac{1}{2n}\sum_{i=1}^n \|w - x_i\|^2,$$

whose optimal solution is $w^* = \bar{x}$. The three algorithms for solving this problem are, respectively,

$$w_{k+1} = w_k - \alpha_k \frac{1}{n}\sum_{i=1}^n (w_k - x_i) = w_k - \alpha_k(w_k - \bar{x}), \qquad \text{(BGD)}$$

$$w_{k+1} = w_k - \alpha_k \frac{1}{m}\sum_{j\in\mathcal{I}_k} (w_k - x_j) = w_k - \alpha_k\left(w_k - \bar{x}_k^{(m)}\right), \qquad \text{(MBGD)}$$

$$w_{k+1} = w_k - \alpha_k(w_k - x_k), \qquad \text{(SGD)}$$

where $\bar{x}_k^{(m)} = \sum_{j\in\mathcal{I}_k} x_j/m$. Furthermore, if $\alpha_k = 1/k$, the above equations can be solved

as follows:

$$w_{k+1} = \frac{1}{k}\sum_{j=1}^{k} \bar{x} = \bar{x}, \qquad \text{(BGD)}$$

$$w_{k+1} = \frac{1}{k}\sum_{j=1}^{k} \bar{x}_j^{(m)}, \qquad \text{(MBGD)}$$

$$w_{k+1} = \frac{1}{k}\sum_{j=1}^{k} x_j. \qquad \text{(SGD)}$$

The derivation of the above equations is similar to that of (6.3) and is omitted here. It can be seen that the estimate given by BGD at each step is exactly the optimal solution $w^* = \bar{x}$. MBGD converges to the mean faster than SGD because $\bar{x}_k^{(m)}$ is already an average.

A simulation example is given in Figure 6.5 to demonstrate the convergence of MBGD. Let $\alpha_k = 1/k$. It is shown that all MBGD algorithms with different mini-batch sizes can converge to the mean. The case with $m = 50$ converges the fastest, while SGD with $m = 1$ is the slowest. This is consistent with the above analysis. Nevertheless, the convergence rate of SGD is still fast, especially when w_k is far from w^*.

6.4.5 Convergence of SGD

The rigorous proof of the convergence of SGD is given as follows.

Theorem 6.4 (Convergence of SGD)**.** *For the SGD algorithm in* (6.13)*, if the following conditions are satisfied, then w_k converges to the root of $\nabla_w \mathbb{E}[f(w, X)] = 0$ almost surely.*

(a) $0 < c_1 \leq \nabla_w^2 f(w, X) \leq c_2$;

(b) $\sum_{k=1}^{\infty} a_k = \infty$ *and* $\sum_{k=1}^{\infty} a_k^2 < \infty$;

(c) $\{x_k\}_{k=1}^{\infty}$ *are i.i.d.*

The three conditions in Theorem 6.4 are discussed below.

- ◇ Condition (a) is about the convexity of f. It requires the curvature of f to be bounded from above and below. Here, w is a scalar, and so is $\nabla_w^2 f(w, X)$. This condition can be generalized to the vector case. When w is a vector, $\nabla_w^2 f(w, X)$ is the well-known Hessian matrix.
- ◇ Condition (b) is similar to that of the RM algorithm. In fact, the SGD algorithm is a special RM algorithm (as shown in the proof in Box 6.1). In practice, a_k is often selected as a sufficiently small *constant*. Although condition (b) is not satisfied in this case, the algorithm can still converge in a certain sense [24, Section 1.5].
- ◇ Condition (c) is a common requirement.

Box 6.1: Proof of Theorem 6.4

We next show that the SGD algorithm is a special RM algorithm. Then, the convergence of SGD naturally follows from the RM theorem.

The problem to be solved by SGD is to minimize $J(w) = \mathbb{E}[f(w, X)]$. This problem can be converted to a root-finding problem. That is, finding the root of $\nabla_w J(w) = \mathbb{E}[\nabla_w f(w, X)] = 0$. Let

$$g(w) = \nabla_w J(w) = \mathbb{E}[\nabla_w f(w, X)].$$

Then, SGD aims to find the root of $g(w) = 0$. This is exactly the problem solved by the RM algorithm. The quantity that we can measure is $\tilde{g} = \nabla_w f(w, x)$, where x is a sample of X. Note that $\tilde{g}$ can be rewritten as

$$\begin{aligned}\tilde{g}(w, \eta) &= \nabla_w f(w, x) \\ &= \mathbb{E}[\nabla_w f(w, X)] + \underbrace{\nabla_w f(w, x) - \mathbb{E}[\nabla_w f(w, X)]}_{\eta}.\end{aligned}$$

Then, the RM algorithm for solving $g(w) = 0$ is

$$w_{k+1} = w_k - a_k \tilde{g}(w_k, \eta_k) = w_k - a_k \nabla_w f(w_k, x_k),$$

which is the same as the SGD algorithm in (6.13). As a result, the SGD algorithm is a special RM algorithm. We next show that the three conditions in Theorem 6.1 are satisfied. Then, the convergence of SGD naturally follows from Theorem 6.1.

- ◇ Since $\nabla_w g(w) = \nabla_w \mathbb{E}[\nabla_w f(w, X)] = \mathbb{E}[\nabla_w^2 f(w, X)]$, it follows from $c_1 \leq \nabla_w^2 f(w, X) \leq c_2$ that $c_1 \leq \nabla_w g(w) \leq c_2$. Thus, the first condition in Theorem 6.1 is satisfied.
- ◇ The second condition in Theorem 6.1 is the same as the second condition in this theorem.
- ◇ The third condition in Theorem 6.1 requires $\mathbb{E}[\eta_k | \mathcal{H}_k] = 0$ and $\mathbb{E}[\eta_k^2 | \mathcal{H}_k] < \infty$. Since $\{x_k\}$ is i.i.d., we have $\mathbb{E}_{x_k}[\nabla_w f(w, x_k)] = \mathbb{E}[\nabla_w f(w, X)]$ for all k. Therefore,

$$\mathbb{E}[\eta_k | \mathcal{H}_k] = \mathbb{E}[\nabla_w f(w_k, x_k) - \mathbb{E}[\nabla_w f(w_k, X)] | \mathcal{H}_k].$$

Since $\mathcal{H}_k = \{w_k, w_{k-1}, \dots\}$ and x_k is independent of $\mathcal{H}_k$, the first term on the right-hand side becomes $\mathbb{E}[\nabla_w f(w_k, x_k) | \mathcal{H}_k] = \mathbb{E}_{x_k}[\nabla_w f(w_k, x_k)]$. The second term becomes $\mathbb{E}[\mathbb{E}[\nabla_w f(w_k, X)] | \mathcal{H}_k] = \mathbb{E}[\nabla_w f(w_k, X)]$ because $\mathbb{E}[\nabla_w f(w_k, X)]$ is

a function of w_k. Therefore,

$$\mathbb{E}[\eta_k|\mathcal{H}_k] = \mathbb{E}_{x_k}[\nabla_w f(w_k, x_k)] - \mathbb{E}[\nabla_w f(w_k, X)] = 0.$$

Similarly, it can be proven that $\mathbb{E}[\eta_k^2|\mathcal{H}_k] < \infty$ if $|\nabla_w f(w, x)| < \infty$ for all w given any x.

Since the three conditions in Theorem 6.1 are satisfied, the convergence of the SGD algorithm follows.

6.5 Summary

Instead of introducing new reinforcement learning algorithms, this chapter introduced the preliminaries of stochastic approximation such as the RM and SGD algorithms. Compared to many other root-finding algorithms, the RM algorithm does not require the expression of the objective function or its derivative. It has been shown that the SGD algorithm is a special RM algorithm. Moreover, an important problem frequently discussed throughout this chapter is mean estimation. The mean estimation algorithm (6.4) is the first stochastic iterative algorithm we have ever introduced in this book. We showed that it is a special SGD algorithm. We will see in Chapter 7 that temporal-difference learning algorithms have similar expressions. Finally, the name "stochastic approximation" was first used by Robbins and Monro in 1951 [25]. More information about stochastic approximation can be found in [24].

6.6 Q&A

- Q: What is stochastic approximation?

  A: Stochastic approximation refers to a broad class of stochastic iterative algorithms for solving root-finding or optimization problems.

- Q: Why do we need to study stochastic approximation?

  A: This is because the temporal-difference reinforcement learning algorithms that will be introduced in Chapter 7 can be viewed as stochastic approximation algorithms. With the knowledge introduced in this chapter, we can be better prepared, and it will not be abrupt for us to see these algorithms for the first time.

- Q: Why do we frequently discuss the mean estimation problem in this chapter?

  A: This is because the state and action values are defined as the means of random variables. The temporal-difference learning algorithms introduced in Chapter 7 are similar to stochastic approximation algorithms for mean estimation.

◇ Q: What is the advantage of the RM algorithm over other root-finding algorithms?

A: Compared to many other root-finding algorithms, the RM algorithm is powerful in the sense that it does not require the expression of the objective function or its derivative. As a result, it is a black-box technique that only requires the input and output of the objective function. The famous SGD algorithm is a special form of the RM algorithm.

◇ Q: What is the basic idea of SGD?

A: SGD aims to solve optimization problems involving random variables. When the probability distributions of the given random variables are not known, SGD can solve the optimization problems merely by using samples. Mathematically, the SGD algorithm can be obtained by replacing the true gradient expressed as an expectation in the gradient descent algorithm with a stochastic gradient.

◇ Q: Can SGD converge quickly?

A: SGD has an interesting convergence pattern. That is, if the estimate is far from the optimal solution, then the convergence process is fast. When the estimate is close to the solution, the randomness of the stochastic gradient becomes influential, and the convergence rate decreases.

◇ Q: What is MBGD? What are its advantages over SGD and BGD?

A: MBGD can be viewed as an intermediate version between SGD and BGD. Compared to SGD, it has less randomness because it uses more samples instead of just one as in SGD. Compared to BGD, it does not require the use of all the samples, making it more flexible.