Introduction to Stochastic Search and Optimization: Estimation, Simulation, and Control. James C. Spall

ISBN: 0-471-33052-3

CHAPTER 4

STOCHASTIC APPROXIMATION FOR NONLINEAR ROOT-FINDING

We now broaden our horizons considerably from the linear setting of Chapter 3 and consider nonlinear systems. This is the first of four chapters that focus on a core approach in nonlinear stochastic search and optimization—*stochastic approximation* (SA). As might be expected, the problem of nonlinear estimation is usually more challenging than estimation in linear models (where the loss function is quadratic—or equivalently, the loss gradient is linear—in the elements of θ). Stochastic search and optimization for nonlinear models is used in a large number of areas. These include (to name just a few) the estimation of connection weights in artificial neural networks, system design for discrete-event dynamic (queuing) systems, image restoration from blurred image data, dose response analysis for drugs, and stochastic adaptive control.

Section 4.1 introduces the basic root-finding (Robbins–Monro) SA algorithm form and Section 4.2 presents several motivating examples. Sections 4.3 and 4.4 discuss some of the theoretical properties related to convergence and asymptotic distributions. Section 4.5 summarizes four extensions to the basic algorithm form of this chapter and Section 4.6 provides some concluding remarks.

4.1 INTRODUCTION

This chapter introduces the method of *stochastic approximation* (SA) for solving nonlinear root-finding problems in the presence of noisy measurements. The basic approach is sometimes referred to as the Robbins–Monro algorithm in honor of the two people who introduced the modern general setting (Robbins and Monro, 1951). Root-finding SA is a cornerstone of stochastic search and optimization as a generalization of the well-known deterministic algorithms in Section 1.4 (steepest descent and Newton–Raphson).

Root-finding SA provides a general framework for convergence analysis of many algorithms that may not appear directly as root-finding methods. These include the recursive-least-squares (RLS) and least-mean-squares (LMS) algorithms of Chapter 3, nonlinear parameter estimation methods of Chapter 5 (e.g., neural network backpropagation), gradient-free SA algorithms of Chapters

6 and 7, simulated annealing and related algorithms of Chapter 8, reinforcement (temporal difference) learning in Chapter 11, simulation-based optimization of Chapters 14 and 15, and some optimal experimental design methods in Chapter 17.

In the notation of Section 1.1, the focus in this chapter is to find at least one root $\boldsymbol{\theta}^* \in \Theta^* \subseteq \Theta \subseteq \mathbb{R}^p$ to

$$\boldsymbol{g}(\boldsymbol{\theta}) = \mathbf{0} \tag{4.1}$$

based on noisy measurements of $\boldsymbol{g}(\boldsymbol{\theta})$. This root-finding problem was introduced in Section 1.1. Note that $\boldsymbol{g}(\boldsymbol{\theta}) \in \mathbb{R}^p$. So the problem is the classic "p equations and p unknowns" with, in general, the significant complications of nonlinear $\boldsymbol{g}(\boldsymbol{\theta})$ and noisy input information. A very important special case is finding a root to $\boldsymbol{g}(\boldsymbol{\theta}) = \partial L/\partial \boldsymbol{\theta} = \mathbf{0}$ when faced with a problem of minimizing $L(\boldsymbol{\theta})$. Given the importance of this special case, a full chapter—Chapter 5—is devoted to the stochastic gradient problem of optimization when only noisy measurements of $\partial L/\partial \boldsymbol{\theta}$ are available.

This chapter focuses on root-finding per se, without dwelling on the special case of optimization ($\partial L/\partial \boldsymbol{\theta} = \mathbf{0}$). Root-finding via SA was introduced in modern form in Robbins and Monro (1951), with important generalizations and extensions following close behind as given in Kiefer and Wolfowitz (1952), Chung (1954), and Blum (1954a, b). Some classic books covering SA are Albert and Gardner (1967), Nevel'son and Has'minskii (1973), and Kushner and Clark (1978). A more recent thorough mathematical treatment is Kushner and Yin (1997).

A central aspect of SA is the allowance for noisy input information in the algorithm. In fact, as we will see, the SA methods in this and the next three chapters are often better at coping with noisy input information than other search methods in this book. Moreover, the theoretical foundation for SA is deeper than the theory for other stochastic search methods with noisy measurements. In the case of root-finding SA, the noise manifests itself in the measurements of $\boldsymbol{g}(\boldsymbol{\theta})$ used in the search as $\boldsymbol{\theta}$ varies. More specifically, as in Section 1.1, suppose that measurements of $\boldsymbol{g}(\boldsymbol{\theta})$ at any $\boldsymbol{\theta}$ are available as

$$\boldsymbol{Y}_k(\boldsymbol{\theta}) = \boldsymbol{g}(\boldsymbol{\theta}) + \boldsymbol{e}_k(\boldsymbol{\theta}), \quad k = 0, 1, 2, \ldots, \tag{4.2}$$

where $\boldsymbol{e}_k(\boldsymbol{\theta})$ is assumed to be some noise term of dimension p.

Closely related to (4.2) is the case where there are input measurements $\boldsymbol{x}_k$ as part of $\boldsymbol{Y}_k(\boldsymbol{\theta})$. These inputs may represent random or deterministic terms (e.g., randomly generated or user-specified target values in a target-tracking problem). So, for a specified $\boldsymbol{\theta}$ *and* $\boldsymbol{x}_k$, a noisy measurement $\boldsymbol{Y}_k(\boldsymbol{\theta})$ is returned. The measurements in this setting are assumed to come from

$$\boldsymbol{Y}_k(\boldsymbol{\theta}) = \tilde{\boldsymbol{g}}_k(\boldsymbol{\theta}, \boldsymbol{x}_k) + \tilde{\boldsymbol{e}}_k(\boldsymbol{\theta}, \boldsymbol{x}_k), \tag{4.3}$$

where $\tilde{g}_k(\theta, x_k)$ and $\tilde{e}_k(\theta, x_k)$ represent function and noise terms analogous to $g(\theta)$ and $e_k(\theta)$ in (4.2). We can reexpress (4.3) in the conventional form of (4.2) by defining

$$e_k(\theta) \equiv \tilde{g}_k(\theta, x_k) - g(\theta) + \tilde{e}_k(\theta, x_k). \tag{4.4}$$

Substituting this noise term in (4.2) yields a measurement the same as (4.3). The introduction of inputs x_k does not fundamentally alter the basic root-finding problem of (4.1). That is, there exists a $g(\theta)$ and $e_k(\theta)$ as in (4.4) such that measurements of form (4.3) yield a solution to (4.1). For example, if the inputs x_k are random and $E[e_k(\theta)] = E[\tilde{e}_k(\theta, x_k)] = 0$, then $g(\theta) = E[\tilde{g}_k(\theta, x_k)]$.

The mechanics of the algorithm are essentially the same in either measurement setting, (4.2) or (4.3). In fact, in much of the literature on SA, the concept of inputs as in (4.3) is suppressed. That is, the problem is usually stated as in (4.1) and (4.2) while assuming (implicitly, at least) a noise structure as in (4.4). In a like manner, to avoid having to continuously distinguish the two settings, we generally just discuss problems in the context of $g(\theta)$ without specific reference to inputs x_k.

In cases where an average $g(\theta)$ does not exist (i.e., $g(\theta) \neq E[\tilde{g}_k(\theta, x_k)]$), there may be an inherent time-varying root-finding problem where $g(\theta)$ is replaced by $g_k(\theta) \equiv \tilde{g}_k(\theta, x_k)$. Some of the theory and methodology of SA extends to the time-varying $g_k(\theta)$ case, as discussed in Subsections 4.5.1 and 4.5.4.

In root-finding SA, the measurements at $\theta = \hat{\theta}_k$ are:

$$Y_k(\hat{\theta}_k) = g(\hat{\theta}_k) + e_k(\hat{\theta}_k), \tag{4.5}$$

where $e_k = e_k(\hat{\theta}_k)$ is the general error term in (4.2) or (4.4). Although e_k may have general statistical properties (e.g., dependence across k and/or nonidentical distributions), an important special case is where $\{e_k\}$ is an independent, identically distributed (i.i.d.) sequence of mean-zero random vectors.

Recalling the basic steepest descent algorithm in Section 1.4 (i.e., $\hat{\theta}_{k+1} = \hat{\theta}_k - a_k g(\hat{\theta}_k)$), an obvious implementation with noisy measurements of $g(\theta)$ is to average $Y_k(\theta)$ values at $\theta = \hat{\theta}_k$. Such averaging is used to approximate $g(\hat{\theta}_k)$ from multiple values of $Y_k(\hat{\theta}_k)$. A significant innovation of Robbins and Monro (1951) was the recognition that this is a wasteful use of the measurements. Recall that $g(\hat{\theta}_k)$ is merely an intermediate calculation towards the ultimate goal of trying to find a root θ^*. There is little interest in $g(\hat{\theta}_k)$ per se. So, the main innovation in SA is to do a form of averaging *across iterations*. At first thought, this type of averaging may seem dubious, since the underlying evaluation point θ is changing across iterations. But, as suggested by Robbins and Monro (1951), this across-iteration averaging can lead to a more effective use of the input information than expending a large amount of resources in getting accurate estimates for $g(\theta)$ at each iteration.

In implementing the across-iteration averaging, the core root-finding SA algorithms are given below (unconstrained and constrained versions). Let the scalar a_k be a nonnegative "gain" value, $\hat{\boldsymbol{\theta}}_0$ be the initial condition, and $\Psi_\Theta[\cdot]$ be a user-defined mapping that projects any point not in the constraint domain Θ to a new point inside Θ.

Basic Root-Finding (Robbins–Monro) SA Algorithms

Unconstrained:
$$\hat{\boldsymbol{\theta}}_{k+1} = \hat{\boldsymbol{\theta}}_k - a_k \boldsymbol{Y}_k(\hat{\boldsymbol{\theta}}_k) \tag{4.6}$$

Constrained:
$$\hat{\boldsymbol{\theta}}_{k+1} = \Psi_\Theta[\hat{\boldsymbol{\theta}}_k - a_k \boldsymbol{Y}_k(\hat{\boldsymbol{\theta}}_k)] \tag{4.7}$$

The above root-finding algorithms are clearly motivated by the steepest descent algorithm with the noisy measurement $\boldsymbol{Y}_k(\hat{\boldsymbol{\theta}}_k)$ replacing the exact root-finding function $\boldsymbol{g}(\hat{\boldsymbol{\theta}}_k)$. A major innovation in the algorithm is the specification of precise conditions on the gain coefficients a_k to ensure that the process in (4.6) or (4.7) properly invokes the across-iteration averaging and converges to a root $\boldsymbol{\theta}^*$. As expected, these conditions generally differ from those in the easier deterministic steepest descent setting. (Of course, these conditions also apply in steepest descent because that is a special case of SA.) We discuss these conditions in Sections 4.3 and 4.4. Because of the minus sign on the right-hand side of (4.6) and (4.7), each component of $\boldsymbol{g}(\boldsymbol{\theta})$ should be positive when the corresponding component of $\boldsymbol{\theta}$ is greater than the corresponding component of $\boldsymbol{\theta}^*$ and negative for the opposite case (this applies in the typical case where $\boldsymbol{e}_k$ has mean zero) (why?). For example, a trivial scalar problem of finding θ such that $1 - \theta = 0$ should be changed to the equivalent $g(\theta) = \theta - 1 = 0$. More formal statements of this sign requirement are given in Section 4.3.

While (4.7) provides a nice conceptual framework for constrained search, the vast majority of practical theoretical results pertain to the unconstrained version (4.6). In applications, it is often difficult to implement (4.7) unless the constraints are fairly benign (such as a hypercube constraint on $\boldsymbol{\theta}$ where each component of $\boldsymbol{\theta}$ has a distinct lower and upper bound). As discussed in Subsection 1.2.1, this issue of difficult analysis and implementation for constrained algorithms is not unique to SA. It affects *all* search and optimization methods, and is especially challenging in stochastic methods as considered in this book.

4.2 POTPOURRI OF STOCHASTIC APPROXIMATION EXAMPLES

Four examples of root-finding SA are given below. The first shows that the classical sample mean of a sequence of random vectors is a special case of SA. The second is a problem of estimating a *quantile*, a point on the real line such that a process will have an outcome below this point with a specified probability.

This problem appears in many contexts. For example, in pharmacology there is interest in determining the required dose such that (say) 90 percent of a population achieves a desired therapeutic response to a treatment at that dose. A bivariate extension of the quantile idea appears in the third example. The goal is to find the radius of a circle about a target such that a projectile directed towards the target is likely to land in this circle with specified probability. If the probability is set to 0.5, this radius is referred to as the *circular error probable* (CEP), which is the single most important measure of accuracy for many military weapon systems. The fourth example is from microeconomics, the aim being to estimate a *production function* relating labor and capital inputs to production outputs for firms in a sector of the economy. This example compares SA with the method of maximum likelihood.

Example 4.1—Sample mean as an SA algorithm. Suppose that independent measurements $\boldsymbol{X}_i$ are available, where the measurements share a common mean $\boldsymbol{\mu}$ (i.e., $E(\boldsymbol{X}_i) = \boldsymbol{\mu}$ for all i). The goal is to estimate $\boldsymbol{\mu}$. The sample mean of the $\boldsymbol{X}_i$ represents the most common estimator:

$$\begin{aligned}\bar{\boldsymbol{X}}_{k+1} &\equiv \frac{1}{k+1}\sum_{i=1}^{k+1}\boldsymbol{X}_i \\ &= \frac{k}{k+1}\bar{\boldsymbol{X}}_k + \frac{1}{k+1}\boldsymbol{X}_{k+1} \\ &= \bar{\boldsymbol{X}}_k - \frac{1}{k+1}\left(\bar{\boldsymbol{X}}_k - \boldsymbol{X}_{k+1}\right),\end{aligned}$$

where $\bar{\boldsymbol{X}}_0 = \mathbf{0}$ in the recursive representation in the second and third lines. Letting $\hat{\boldsymbol{\theta}}_k = \bar{\boldsymbol{X}}_k$, $a_k = 1/(k+1)$, and $\boldsymbol{Y}_k(\hat{\boldsymbol{\theta}}_k) = \bar{\boldsymbol{X}}_k - \boldsymbol{X}_{k+1} = \hat{\boldsymbol{\theta}}_k - \boldsymbol{X}_{k+1}$ puts this recursion for the sample mean in the framework of SA recursion (4.6). Further, connections to (4.2) (and thus (4.5)) are apparent by letting $\boldsymbol{g}(\boldsymbol{\theta}) = \boldsymbol{\theta} - \boldsymbol{\mu}$ and $\boldsymbol{e}_k = \boldsymbol{\mu} - \boldsymbol{X}_{k+1}$ (i.e., $\boldsymbol{e}_k$ is independent of $\boldsymbol{\theta}$ and has mean zero). Hence, the simple sample mean calculation for a sequence of random vectors represents a special case of an SA algorithm. (Exercise 4.3 considers the problem of estimating the mean using a different gain a_k.) ❑

Example 4.2—LD_{50} quantile. Consider the following problem in quantile estimation, similar to that described in Robbins and Monro (1951). Let $F_X(x) = P(X \le x)$ be an unknown distribution function for a scalar random variable X. Suppose that a researcher wants to estimate the scalar quantile θ such that $F_X(\theta) = 0.5$, i.e., find the root of the equation $g(\theta) = F_X(\theta) - 0.5 = 0$. In clinical trials, the indicated probability 0.5 has special importance in determining a quantity called LD_{50}—*lethal dosage 50*—the dosage that is lethal to 50 percent of a population of organisms.

Suppose that the researcher is not allowed to know the value of X in an experiment, but is allowed to know whether X is above or below a specified threshold. In particular, in running a sequence of experiments generating outcomes $X_0, X_1, \ldots$ according to the distribution $F_X(x)$, the researcher is free to specify a value $\hat{\theta}_k$ such that information about whether X_k is smaller or larger than $\hat{\theta}_k$ (a "success" or "nonsuccess," respectively) is returned. Formally, let us introduce the success variable:

$$s_k(\hat{\theta}_k) = \begin{cases} 1 \text{ if } X_k \le \hat{\theta}_k \text{ (success)}, \\ 0 \text{ otherwise (nonsuccess)}. \end{cases}$$

With $\hat{\theta}_0$ as the best guess of the quantile θ such that $F_X(\theta) = 0.5$, the root-finding SA recursion has the form

$$\hat{\theta}_{k+1} = \hat{\theta}_k - a_k\left(s_k(\hat{\theta}_k) - 0.5\right).$$

So, $Y_k(\hat{\theta}_k) = s_k(\hat{\theta}_k) - 0.5$. Note that $E[Y_k(\hat{\theta}_k)\,|\,\hat{\theta}_k] = F_X(\hat{\theta}_k) - 0.5 = g(\hat{\theta}_k)$, so that the noise e_k has mean zero. Intuitively, it is apparent that if $\hat{\theta}_k$ is too small, then $s_k(\hat{\theta}_k) = 0$ is more likely than $s_k(\hat{\theta}_k) = 1$, leading to $Y_k(\hat{\theta}_k) = 0 - 0.5 < 0$. This, in turn, tends to make $\hat{\theta}_{k+1}$ larger than $\hat{\theta}_k$ (as desired) according to the update $\hat{\theta}_{k+1} = \hat{\theta}_k + 0.5a_k$. The other possible outcome, $Y_k(\hat{\theta}_k) = 1 - 0.5 > 0$, which is less likely to occur, leads to an update with the same magnitude of change in θ but in the wrong direction of decreasing θ (i.e., $\hat{\theta}_{k+1} = \hat{\theta}_k - 0.5a_k$). Overall, therefore, there is a tendency to make θ larger, as desired. Conversely, if $\hat{\theta}_k$ is too large, the next value $\hat{\theta}_{k+1}$ is likely to be smaller than $\hat{\theta}_k$.

This example points to the important role of a_k. If a_k is too small, the progress of the algorithm to the optimal quantile θ^* will be sluggish because the increment $0.5a_k$ (the magnitude of change in θ value) will be too small. Conversely, too large a value for a_k may cause the new estimate to vastly overshoot the LD_{50} quantile. Exercise 4.4 demonstrates some numerical performance associated with this quantile example. ❑

Example 4.3—Circular error probable (CEP). Let us now consider a numerical example for the CEP problem mentioned above. More detail on this problem is given in Grubbs (1964) and Spall and Maryak (1992). The algorithm processes projectile impact measurements $\boldsymbol{X}_k \in \mathbb{R}^2$ having a (generally unknown) bivariate probability distribution. The first coordinate in $\boldsymbol{X}_k$ represents the crossrange direction and the second coordinate represents the downrange direction. The mean of the bivariate distribution represents the impact bias relative to the target and the covariance matrix represents the dispersion of the impact points in the two coordinates.

Recall that the Euclidean norm $\|\cdot\|$ has the geometric interpretation of distance from the origin to the argument point. Then, following Example 4.2, let

$$s_k(\hat{\theta}_k) = \begin{cases} 1 \text{ if } \|\boldsymbol{X}_k\| \le \hat{\theta}_k \quad \text{(success)}, \\ 0 \text{ otherwise (nonsuccess)}. \end{cases}$$

The SA recursion has a form identical to that in Example 4.2: $\hat{\theta}_{k+1} = \hat{\theta}_k - a_k(s_k(\hat{\theta}_k) - 0.5)$ (implying that $Y_k(\hat{\theta}_k) = s_k(\hat{\theta}_k) - 0.5$). Arguments similar to those in Example 4.2 can be used to show the rationale of this form.

Suppose that the impacts $\boldsymbol{X}_k$ arrive according to a $N([1, 1]^T, \text{diag}[4, 1])$ distribution (the target is the point $[0, 0]^T$). For this statistical model, the true CEP equals 2.17. (This value is found from a combination of deterministic numerical optimization and numerical quadrature applied to the integral of the bivariate normal density function; an alternative approach is the highly accurate closed-form approximation in Grubbs, 1964.) In practice, of course, the data-generating mechanism (true model) is not known exactly and so the exact solution is unavailable for comparison.

Based on simulated data from the assumed model, one replication of the root-finding SA algorithm in (4.7) finds estimates of 1.60, 1.82, and 2.16, respectively, after n = 20, 100, and 1000 experimental impact points $\boldsymbol{X}_k$ (each generating one $Y_k(\cdot)$). These estimates are based on $\hat{\theta}_0 = 0.5$ and $a_k = 1/(k+1)$. A better initial condition or a tuned a_k sequence produces a better SA estimate at a specified n. Figure 4.1 shows a plot of the 1000 impacts and the associated true CEP and (indistinguishable) CEP estimate. A visual inspection shows that the circle contains approximately half the impact points, as expected by the

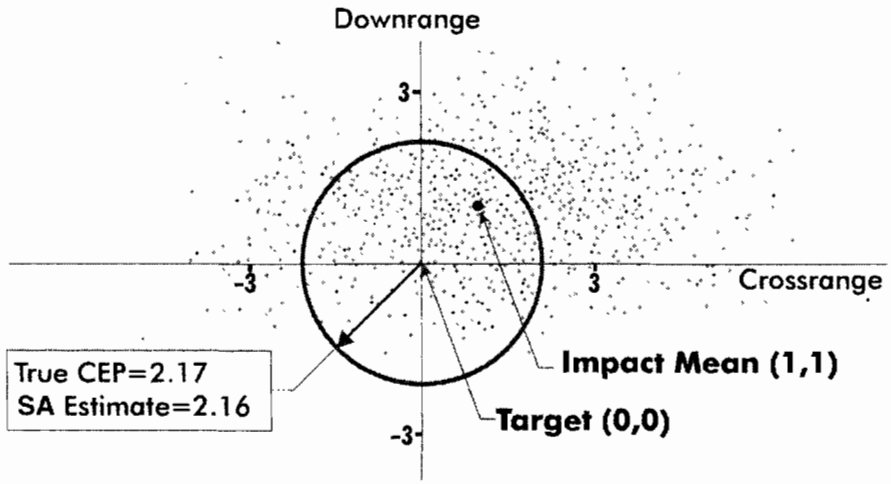

Figure 4.1. 1000 impact points with impact mean differing from target point. The indicated circle is centered at the target with radius equal to the CEP.

definition of CEP. Note the bias in the impacts toward the upper right quadrant and greater spread in the crossrange direction (the standard deviation for crossrange is double that of downrange). Because the CEP radius is relative to the target, not the mean of impacts, the CEP is larger with a nonzero mean than with a zero mean given that the covariance matrix is the same. ❑

Example 4.4—Production function in microeconomics. An important concern in economics is the tradeoff between different types of inputs that can be used in making a product. At its simplest level, this tradeoff may be between work hours (W) and capital equipment (C) in the manufacturing of a product. A common means of analyzing the tradeoff is through the use of a production function $h(\cdot)$ that relates the quantities of labor and capital to the amount of output z_k:

$$z_k = h(\boldsymbol{\theta}, \boldsymbol{x}_k) + v_k,$$

where $\boldsymbol{\theta}$ is a vector of parameters of the function, $\boldsymbol{x}_k$ represents the inputs used in the production of the product for index k (e.g., $\boldsymbol{x}_k = [W_k, C_k]^T$ represent the work hours and capital inputs at the kth time period), and v_k is the random noise. In the numerical results below, W_k and C_k are generated randomly (uniformly) in [1, 10] and [11, 100] at each k, respectively.

An important special case of a production function for two inputs is the Cobb–Douglas form.[1] Let $\boldsymbol{\theta} = [\lambda, \beta]^T$, where $\lambda > 0$ and $0 \le \beta \le 1$ are parameters having economic significance relative to total production capability and the degree of tradeoff possible between the two inputs. The functional form for this production function together with additive noise is

$$z_k = h(\boldsymbol{\theta}, \boldsymbol{x}_k) + v_k = \lambda C_k^{\beta} W_k^{1-\beta} + v_k \tag{4.8}$$

(Kmenta, 1997, pp. 252–253). One well-known approach to estimating $\boldsymbol{\theta}$ from a sequence of n input–output data pairs $\{(\boldsymbol{x}_1, z_1), (\boldsymbol{x}_2, z_2), \ldots, (\boldsymbol{x}_n, z_n)\}$ is the method of maximum likelihood (ML) (e.g., Kmenta, 1997, pp. 582–583). An alternative approach is to apply the SA algorithm to estimate $\boldsymbol{\theta}$ based on the above form for $h(\boldsymbol{x}_k, \boldsymbol{\theta})$. Note that a model similar to (4.8) was considered in Section 3.1 (Example 3.2), but unlike the earlier model, the form in (4.8) cannot be converted to an equivalent linear form via a logarithmic (or other) transformation because the noise is additive (versus multiplicative in Example 3.2).

One of the points illustrated in this example is that there is a fundamental difference between SA and standard approaches such as ML in that SA operates under weaker conditions. In particular, SA does not require assumptions about

[1]This production function was introduced in the 1930s by economists Charles W. Cobb and Paul H. Douglas; Douglas served as a U.S. senator from 1948 to 1966.

the distributional form of the outputs z_k. Although ML has certain optimality properties, it may yield a poor estimate when the actual data have a distribution significantly different from that assumed in forming the likelihood criterion.

The ML estimator is derived by maximizing the joint probability density function of the n data points when viewed as a function of $\boldsymbol{\theta}$. It is common to assume that the data are independent, normally distributed, which leads to an ML criterion based on the normal density function. Reflecting common practice in ML estimation, the results below are based on this criterion, even when the true data are not normally distributed.

The loss function for the SA recursion is the expected squared error

$$L(\boldsymbol{\theta}) = \tfrac{1}{2} E\left\{[z_{k+1} - h(\boldsymbol{\theta}, \boldsymbol{x}_{k+1})]^2\right\},$$

where the expectation is taken with respect to the noise v_k and the randomness in $\boldsymbol{x}_k$. The stochastic gradient (i.e., the derivative of the argument in the above expectation) is

$$\boldsymbol{Y}_k(\boldsymbol{\theta}) = \frac{1}{2} \frac{\partial [z_{k+1} - h(\boldsymbol{\theta}, \boldsymbol{x}_{k+1})]^2}{\partial \boldsymbol{\theta}}.$$

So, the input at step k is calculated as

$$\boldsymbol{Y}_k(\hat{\boldsymbol{\theta}}_k) = [h(\hat{\boldsymbol{\theta}}_k, \boldsymbol{x}_{k+1}) - z_{k+1}] \left. \frac{\partial h(\boldsymbol{\theta}, \boldsymbol{x}_{k+1})}{\partial \boldsymbol{\theta}} \right|_{\boldsymbol{\theta}=\hat{\boldsymbol{\theta}}_k}.$$

We use this gradient in the root-finding algorithm (4.7). Note that SA assumes no knowledge of the distributions generating the data (i.e., the expectation associated with $L(\boldsymbol{\theta})$ is never actually computed).

The results of the analysis are shown on Table 4.1. Data are generated according to the form in (4.8) with $\boldsymbol{\theta}$ taken as $\boldsymbol{\theta}^* = [2.5, 0.70]^T$. The table contrasts the ML estimates with those of SA. The first row of results in Table 4.1 shows the parameter estimates when the added noise matches the assumptions for ML. The second row shows the results when the noise is not normally distributed, but is distributed according to $v_k = h(\boldsymbol{\theta}^*, \boldsymbol{x}_k)(\xi_k - 1)$, where ξ_k is generated from an exponential distribution with a mean of 1.0 (see Appendix D for a brief discussion of the exponential distribution). To test the estimates, 100 new data points z_k are generated for each of the normal and nonnormal cases using $\boldsymbol{\theta} = \boldsymbol{\theta}^*$. Then, for each $\boldsymbol{\theta}$ (corresponding to one of the four estimates in the table), the sample root-mean-squared (RMS) errors are computed according to

$$\text{RMS} = \sqrt{\frac{1}{100} \sum_{k=1}^{100} [z_k - h(\boldsymbol{\theta}, \boldsymbol{x}_k)]^2}.$$

Table 4.1. ML and SA estimates for production function parameters $\boldsymbol{\theta} = [\lambda, \beta]^T$ from one realization of 1000 measurements ($\boldsymbol{\theta}^* = [2.5, 0.70]^T$). RMS errors are determined from prediction errors using measurements independent of the measurements used for estimating $\boldsymbol{\theta}$.

Noise distribution	ML estimate	RMS error	SA estimate	RMS error
Normal	$[2.54, 0.71]^T$	0.129	$[2.47, 0.70]^T$	0.268
Nonnormal	$[2.76, 0.70]^T$	2.822	$[2.48, 0.67]^T$	1.725

(So each summand involves the difference between an actual output z_k and a prediction $h(\boldsymbol{\theta}, x_k)$.)

While the ML estimate does better than the SA estimate when the data are normally distributed (consistent with the assumptions), the opposite is true when the data are not normally distributed. Relative to SA, the ML estimate significantly degrades when the true noise distribution deviates from the assumptions. (Both of the SA and ML estimates degrade in the nonnormal case as a reflection of greater noise variability.) This example is an introduction to the notion of stochastic gradient, covered in more detail in Chapter 5. ❑

4.3 CONVERGENCE OF STOCHASTIC APPROXIMATION

4.3.1 Background

As with any search algorithm, it is of interest to know whether the iterate $\hat{\boldsymbol{\theta}}_k$ converges to a solution $\boldsymbol{\theta}^* \in \Theta^*$ as k gets large. In fact, one of the strongest aspects of SA is the rich convergence theory that has developed over many years. The versatility of SA theory allows it to be used to show convergence of stochastic algorithms that, on the surface, may not look like SA. For example, SA is used to analyze the convergence of neural network backpropagation (see Section 5.2), simulated annealing (Section 8.6), evolutionary computation (Section 10.5), temporal difference learning (Section 11.6), simulation-based optimization (Chapters 14 and 15), Markov chain Monte Carlo (Section 16.6), and sequential experimental design (Section 17.4). This allows researchers and analysts in many fields to establish formal convergence where otherwise that may have remained an open question.

Most of the stated convergence results for SA are in the almost sure (a.s.) sense. Of historical note is that Robbins and Monro (1951) gave conditions for *mean-squared* (m.s.) convergence, which implies "in probability" (pr.) convergence. Blum (1954a,b) was the first to give conditions for a.s.

convergence. As discussed in Appendix C, neither a.s. nor m.s. convergence is implied by the other, but pr. convergence is implied by a.s. convergence.

Many sufficient conditions have been given over the years for a.s. convergence of the SA recursions in (4.6) and (4.7). The convergence results have largely evolved out of two general settings. One focuses on the imposition of statistical conditions on the function $\boldsymbol{g}(\boldsymbol{\theta})$ and noise $\boldsymbol{e}_k(\boldsymbol{\theta})$. Young (1984, pp. 33–41), Ruppert (1991), and Rustagi (1994, Chap. 9), for example, discuss such results, which have largely evolved out of a statistics perspective. The other setting is based on defining an underlying ordinary differential equation (ODE) that roughly emulates the SA algorithm in (4.6) for large k and as the random effects disappear. This approach has been particularly popular in the applied mathematics and engineering literature. Ljung (1977), Kushner and Clark (1978, Chap. 2), Benveniste et al. (1990, Part I, Chap. 2), and Kushner and Yin (1997, Chaps. 5 and 6) discuss convergence results from the ODE perspective. It turns out that the convergence properties of this *deterministic* differential equation are closely related to the *stochastic* convergence properties of (4.6).

Neither of the "statistics" or "engineering" conditions mentioned above is a special case of the other, so neither set is necessarily weaker. One must consider the application in determining which set of conditions is easier to verify. Note that the conditions given here are not the weakest possible. Rather, they have been chosen to convey the essential flavor of the conditions commonly used to guarantee convergence of SA algorithms. Some of the references cited above (and elsewhere) present conditions more general than those here. As is typical in convergence results for stochastic algorithms, some of these conditions involve aspects of the problem that may be unknown to the analyst (e.g., conditions requiring full knowledge of $\boldsymbol{g}(\boldsymbol{\theta})$). This conundrum seems unavoidable. Meaningful convergence results naturally require assumptions about essential aspects of the problem structure.

These conditions apply when there is a unique root $\boldsymbol{\theta}^*$. Hence, when used for optimization (à la $\partial L/\partial\boldsymbol{\theta} = \mathbf{0}$), they apply when there are no local minima different from the (unique) global minimum. Sections 7.7 and (especially) 8.4 discuss the use of SA for global optimization in the face of multiple local minima.

Also note that while these conditions—together with other similar conditions in the literature—play a central role in the theoretical analysis of SA, they are *sufficient* conditions. Many practical applications of SA produce satisfactory results when one or more of the conditions are not satisfied.

4.3.2 Convergence Conditions

This subsection presents the "statistics" and "engineering" conditions for strong (a.s.) convergence of the SA iterate $\hat{\boldsymbol{\theta}}_k$. Some conditions of the first (statistics) type are given below. These are drawn from Blum (1954a, b) and Nevel'son and Has'minskii (1973, Sect. 4.4).

A.1 **(Gain sequence)** $a_k > 0$, $a_k \to 0$, $\sum_{k=0}^{\infty} a_k = \infty$, and $\sum_{k=0}^{\infty} a_k^2 < \infty$.

A.2 **(Search direction)** For some symmetric, positive definite matrix $\boldsymbol{B}$ and every $0 < \eta < 1$,

$$\inf_{\eta < \|\boldsymbol{\theta} - \boldsymbol{\theta}^*\| < 1/\eta} (\boldsymbol{\theta} - \boldsymbol{\theta}^*)^T \boldsymbol{B}\, \boldsymbol{g}(\boldsymbol{\theta}) > 0 .$$

(The "inf" statement pertains to the infimum [see Appendix A] of the expression $(\boldsymbol{\theta} - \boldsymbol{\theta}^*)^T \boldsymbol{B}\, \boldsymbol{g}(\boldsymbol{\theta})$ over the set of $\boldsymbol{\theta}$ such that $\eta < \|\boldsymbol{\theta} - \boldsymbol{\theta}^*\| < 1/\eta$. One may choose any convenient $\boldsymbol{B}$; often, $\boldsymbol{B} = \boldsymbol{I}_p$.)

A.3 **(Mean-zero noise)** $E[\boldsymbol{e}_k(\boldsymbol{\theta})] = \boldsymbol{0}$ for all $\boldsymbol{\theta}$ and k.

A.4 **(Growth and variance bounds)** $\|\boldsymbol{g}(\boldsymbol{\theta})\|^2 + E\left(\|\boldsymbol{e}_k(\boldsymbol{\theta})\|^2\right) \le c\left(1 + \|\boldsymbol{\theta}\|^2\right)$ for all $\boldsymbol{\theta}$ and k and some $c > 0$.

Let us offer a few comments about the above conditions. From the point of view of the user's input, condition A.1 is the most relevant. This condition provides a careful balance in having the gain a_k decay neither too fast nor too slow. In particular, the gain should approach zero sufficiently fast ($a_k \to 0$, $\sum_k a_k^2 < \infty$) to damp out the noise effects as the iterate gets near the solution $\boldsymbol{\theta}^*$ but should approach zero at a sufficiently slow rate ($\sum_k a_k = \infty$) to avoid premature (false) convergence of the algorithm. Condition A.2 is a fairly stringent condition on the shape of $\boldsymbol{g}(\boldsymbol{\theta})$. The analyst need only find one valid $\boldsymbol{B}$ that satisfies the indicated inequality. For example, in the linear case, if $\boldsymbol{g}(\boldsymbol{\theta}) = \boldsymbol{A}(\boldsymbol{\theta} - \boldsymbol{\theta}^*)$ for some matrix $\boldsymbol{A}$, then one must identify a $\boldsymbol{B}$ such that $\boldsymbol{BA}$ is positive definite. Condition A.3 is the standard mean-zero noise condition and A.4 provides restrictions on the magnitude of $\boldsymbol{g}(\boldsymbol{\theta})$. In particular, this condition says that $\|\boldsymbol{g}(\boldsymbol{\theta})\|^2$ and the variance elements of $\boldsymbol{e}_k$ cannot grow faster than a quadratic function of $\boldsymbol{\theta}$ (note that $\boldsymbol{e}_k$ is allowed to be a function of $\boldsymbol{\theta}$, as discussed in Section 4.1). By the mean-zero condition in A.3, the left-hand side of the inequality in A.4 can be written as $E\left(\|\boldsymbol{Y}_k(\boldsymbol{\theta})\|^2\right) = \|\boldsymbol{g}(\boldsymbol{\theta})\|^2 + E\left(\|\boldsymbol{e}_k(\boldsymbol{\theta})\|^2\right)$ (why?). Note that there are no conditions on the smoothness of $\boldsymbol{g}(\boldsymbol{\theta})$, such as a requirement that $\boldsymbol{g}(\boldsymbol{\theta})$ be differentiable.

Using the ODE approach mentioned above, conditions of the second (engineering) type are given below. These conditions are special cases of more general conditions in Kushner and Clark (1978, Theorem 2.3.1), Metivier and Priouret (1984), and Kushner and Yin (1997, Theorem 5.2.1). While these conditions are included here because of the importance the ODE approach plays in the analysis of SA algorithms, it is recognized that ODEs and some other concepts used in the conditions (such as "infinitely often") are not within the prerequisites of this book. A reader may simply skim these conditions to get the

flavor of the ODE approach. With the exception of Examples 4.5 and 4.6 below, most other aspects of SA to follow do not rest critically on the details here. The broader ODE-based approach, as discussed throughout Kushner and Yin (1997) and in many references cited therein, lends itself to significant generalizations beyond the conditions here. These generalizations include cases involving some types of discontinuities in $\boldsymbol{g}(\boldsymbol{\theta})$ (which are useful, e.g., in areas such as manufacturing and signal processing; see Kushner and Yin, 1997, Chap. 9).

B.1 **(Gain sequence)** $a_k > 0$, $a_k \to 0$, $\sum_{k=0}^{\infty} a_k = \infty$.

B.2 **(Relationship to ODE)** Let $\boldsymbol{g}(\boldsymbol{\theta})$ be continuous on $\mathbb{R}^p$. With $\boldsymbol{Z}(\tau) \in \mathbb{R}^p$ representing a time-varying function (τ denoting time), suppose that the differential equation given by $d\boldsymbol{Z}(\tau)/d\tau = -\boldsymbol{g}(\boldsymbol{Z}(\tau))$ has an asymptotically stable equilibrium point at $\boldsymbol{\theta}^*$ (we use τ, rather than t, to denote time to avoid potential confusion with the elements of $\boldsymbol{\theta}$: $t_1, t_2, \ldots, t_p$). (An asymptotically stable equilibrium has the following two requirements: (i) For every $\eta > 0$, there exists a $\delta(\eta)$ such that $\|\boldsymbol{Z}(\tau) - \boldsymbol{\theta}^*\| \le \eta$ for all $\tau > 0$ whenever $\|\boldsymbol{Z}(0) - \boldsymbol{\theta}^*\| \le \delta(\eta)$, and (ii) there exists a δ_0 such that $\boldsymbol{Z}(\tau) \to \boldsymbol{\theta}^*$ as $\tau \to \infty$ whenever $\|\boldsymbol{Z}(0) - \boldsymbol{\theta}^*\| \le \delta_0$.)

B.3 **(Iterate boundedness)** $\sup_{k \ge 0} \|\hat{\boldsymbol{\theta}}_k\| < \infty$ a.s. Further, $\hat{\boldsymbol{\theta}}_k$ lies in a compact (i.e., closed and bounded) subset of the "domain of attraction" for the differential equation in B.2 infinitely often. (The domain of attraction is that set such that $\boldsymbol{Z}(\tau)$ will converge to $\boldsymbol{\theta}^*$ for any starting point in the domain; "infinitely often" is largely self-descriptive, but is defined more formally, e.g., in Laha and Rohatgi, 1979, p. 73.)

B.4 **(Bounded variance property of measurement error)** Let $\Im_k \equiv \{\hat{\boldsymbol{\theta}}_0, \hat{\boldsymbol{\theta}}_1, \ldots, \hat{\boldsymbol{\theta}}_k\}$ (for $k \ge 1$, the information equivalent to $\hat{\boldsymbol{\theta}}_0$ plus the cumulative inputs $\boldsymbol{Y}_i = \boldsymbol{Y}_i(\hat{\boldsymbol{\theta}}_i)$, $0 \le i \le k-1$). Let $\boldsymbol{b}_k = E[\boldsymbol{e}_k(\hat{\boldsymbol{\theta}}_k) \,|\, \Im_k]$ ($\boldsymbol{b}_k$ for "bias"). Then $E\left[\left\|\sum_{k=0}^{\infty} a_k(\boldsymbol{e}_k - \boldsymbol{b}_k)\right\|^2\right] < \infty$.

B.5 **(Disappearing bias)** $\sup_{k \ge 0} \|\boldsymbol{b}_k\| < \infty$ a.s. and $\boldsymbol{b}_k \to \boldsymbol{0}$ a.s. as $k \to \infty$.

Let us comment briefly on the above conditions. Although B.1 appears slightly weaker than its companion condition, A.1, it is for many practical purposes equivalent. This follows from the practical implications of B.4. In particular, if $\boldsymbol{b}_k = \boldsymbol{0}$ for all k, and $\boldsymbol{e}_i$ and $\boldsymbol{e}_j$ are uncorrelated for all $i \ne j$, then the more general boundedness condition in B.4 can be replaced by $E\left[\sum_{k=0}^{\infty} a_k^2 \|\boldsymbol{e}_k\|^2\right] < \infty$. In addition, if $\text{cov}(\boldsymbol{e}_k) \ge \eta \boldsymbol{I}_p$ for all k and some $\eta > 0$ (implying that $\inf_{k \ge 0} E(\|\boldsymbol{e}_k\|^2) \ge \eta p$; see Exercise 4.6), then condition B.4 requires that $\sum_k a_k^2 < \infty$ (i.e., condition A.1 and B.1 are then effectively the same). There are, however, other convergence results based on ODEs (see, e.g., Kushner and Yin, 1997, Chaps. 5 and 6) where B.1 is sufficient and the

condition $\sum_k a_k^2 < \infty$ is not required. Condition B.2 pertains to the above-mentioned ODE analysis, relating the SA recursion to a deterministic path for a related ODE. Unlike the previous "statistics" conditions, it is assumed that $\boldsymbol{g}(\boldsymbol{\theta})$ is smooth in the sense that it is continuous.

The boundedness condition B.3 is somewhat controversial (e.g., Benveniste et al., 1990, p. 46) since it imposes a requirement on the very iterate that one is trying to analyze. Kushner and Clark (1978, p. 40) point out that this condition is, in fact, not strong since one typically imposes bounds on $\boldsymbol{\theta}$ in practice. Borkar and Meyn (2000) present sufficient conditions for the boundedness to hold in terms of the associated ODE while Chen (2002, Chaps. 2 and 3) focuses on a method of expanding (iterate-varying) truncations to eliminate the condition. Condition B.4 is a special case of the so-called convergence systems (e.g., Lai, 1985) and ensures that the important martingale convergence theorem from probability theory (e.g., Laha and Rohatgi, 1979, pp. 396–400) can be used to cope with the noise effects in the SA recursion. (This book does not delve into martingales, but the subject plays an important role in the convergence theory for SA.) Finally, condition B.5 is a generalization of the mean-zero noise condition in A.3; here the noise is only required to have a mean that *converges to* (versus being *equal to*) zero. Among other uses, this extension is useful in proving convergence for some of the SA methods for optimization without direct gradient measurements (Chapters 6 and 7).

Theorem 4.1. Consider the unconstrained algorithm (4.6) (i.e., $\boldsymbol{g}(\boldsymbol{\theta})$ has the domain $\Theta = \mathbb{R}^p$). Suppose that either conditions A.1–A.4 hold or conditions B.1–B.5 hold. Further, suppose that $\boldsymbol{\theta}^*$ is a unique solution to $\boldsymbol{g}(\boldsymbol{\theta}) = \mathbf{0}$ (i.e., the set of solutions Θ^* is the singleton $\boldsymbol{\theta}^*$). Then $\hat{\boldsymbol{\theta}}_k \to \boldsymbol{\theta}^*$ a.s. as $k \to \infty$.

Comments on proof of Theorem 4.1. The proof under either conditions A.1–A.4 or B.1–B.5 uses mathematical machinery beyond the level of this book. The proof based on A.1–A.4 follows as in the proof of Corollary 4.1 in Nevel'son and Has'minskii (1973, p. 93), which is closely related to the proof in Blum (1954b). For the result based on B.1–B.5, the result follows from the proof of Theorem 2.3.1 in Kushner and Clark (1978, pp. 39–43). Note that B.4 is a sufficient condition for Kushner and Clark's more general sufficient condition A2.2.4 by the martingale convergence result mentioned in Kushner and Clark (1978, p. 27). ❑

4.3.3 On the Gain Sequence and Connection to ODEs

The choice of the gain sequence a_k is critical to the performance of SA. The scaled harmonic sequence $a_k = a/(k+1)$, $a > 0$, $k \geq 0$, is the best-known example of a gain sequence that satisfies condition A.1 (and, of course, B.1). As discussed in Section 4.4, this harmonic decay rate of $O(1/k)$ is optimal with respect to the

limiting rate of convergence of $\hat{\boldsymbol{\theta}}_k$, although slower decay rates may be superior in practical (finite-sample) problems. Note that the sequences $a_k = a/(k+1)^2$ and $a_k = a/\sqrt{k+1}$ do *not* satisfy condition A.1. Usually, some numerical experimentation is required to choose the best value of the coefficient a that appears in the gain.

A common generalization of the harmonic sequence is $a_k = a/(k+1)^\alpha$ for strictly positive a and α. From basic calculus, picking $1/2 < \alpha \leq 1$ yields an a_k satisfying the conditions $\sum_{k=0}^{\infty} a_k = \infty$ and $\sum_{k=0}^{\infty} a_k^2 < \infty$ appearing in A.1. Section 4.4 includes more discussion on the choice of the gain sequences.

A key aspect of the "engineering" conditions B.1–B.5, is the connection of the SA recursion to the underlying differential equation

$$\frac{d\boldsymbol{Z}}{d\tau} = -\boldsymbol{g}(\boldsymbol{Z}), \quad \boldsymbol{Z} = \boldsymbol{Z}(\tau). \tag{4.9}$$

Let us now provide some intuitive basis for this connection. Because $\boldsymbol{g}(\boldsymbol{\theta}^*) = \mathbf{0}$, the constant solution $\boldsymbol{Z}(\tau) = \boldsymbol{\theta}^*$ represents an equilibrium point of the above differential equation. That is, because $d\boldsymbol{Z}(\tau)/d\tau = \mathbf{0}$ at $\boldsymbol{Z}(\tau) = \boldsymbol{\theta}^*$, the system $\boldsymbol{Z}(\tau)$ is not going to move from $\boldsymbol{\theta}^*$ unless an external disturbance is introduced.

To motivate the connection of SA to the ODE, note that a *deterministic* version of (4.6) (equivalent to the steepest descent algorithm in Section 1.4) can be written as

$$\frac{\hat{\boldsymbol{\theta}}_{k+1} - \hat{\boldsymbol{\theta}}_k}{a_k} = -\boldsymbol{g}(\hat{\boldsymbol{\theta}}_k). \tag{4.10}$$

Suppose that a_k is viewed as an increment in time, say $a_k = \tau_{k+1} - \tau_k$, where τ_k represents the kth time point. Equivalently, $\tau_{k+1} = \sum_{i=0}^{k} a_i$. Because $\hat{\boldsymbol{\theta}}_k$ is now a deterministic process, we can write $\hat{\boldsymbol{\theta}}_k = \boldsymbol{Z}(\tau_k)$ for some deterministic function $\boldsymbol{Z}(\cdot)$. Then, (4.10) can be reexpressed as

$$\frac{\boldsymbol{Z}(\tau_{k+1}) - \boldsymbol{Z}(\tau_k)}{\tau_{k+1} - \tau_k} = -\boldsymbol{g}(\boldsymbol{Z}(\tau_k)). \tag{4.11}$$

Because $a_k = \tau_{k+1} - \tau_k \to 0$ as $k \to \infty$ (assumption B.1), the ODE in (4.9) can be regarded as a limiting form of the difference equation (4.11). Hence, for sufficiently large k, the behavior of (4.10) and (4.11) bear close resemblance to the ODE in (4.9).

Of course, the deterministic recursion in (4.10) is *not* identical to the SA recursion of interest, (4.6). The addition of the noise (i.e., $\boldsymbol{Y}(\boldsymbol{\theta})$ measurements instead of $\boldsymbol{g}(\boldsymbol{\theta})$ measurements) represents a fundamental distinction between these two recursions. The non-ODE-related conditions for convergence (B.1 on

the gains, B.4 on the bounded variance, etc.) bridge the gap between the deterministic ODE and the stochastic algorithm of interest.

Let us now present two examples of the construction and analysis of the associated ODE. The first is a simple linear root-finding problem and the second is a nonlinear problem. The ability to verify the ODE conditions in detail (i.e., to write down and solve the ODEs) is generally not possible in practice. In both of the examples below, there is complete knowledge of $\boldsymbol{g}(\boldsymbol{\theta})$, which will not be the case when noise is present. Nevertheless, this idealized structure provides some insight into the role of ODEs in the convergence of SA algorithms.

Example 4.5—ODE for a linear problem. Suppose that $p = 2$ and

$$\boldsymbol{g}(\boldsymbol{\theta}) = \begin{bmatrix} 2 & 1 \\ 1 & 2 \end{bmatrix} \boldsymbol{\theta}.$$

Expressed as an ODE as in (4.9), the above is

$$\frac{d\boldsymbol{Z}}{d\tau} = -\begin{bmatrix} 2 & 1 \\ 1 & 2 \end{bmatrix} \boldsymbol{Z},$$

leading to the solution

$$\boldsymbol{Z}(\tau) = -\begin{bmatrix} C_0 e^{-3\tau} + C_1 e^{-\tau} \\ C_0 e^{-3\tau} - C_1 e^{-\tau} \end{bmatrix},$$

where C_0 and C_1 are constants (Exercise 4.9). The constants are determined from the initial condition $\boldsymbol{Z}(0)$; in particular $-[C_0 + C_1, C_0 - C_1]^T = \boldsymbol{Z}(0)$. We see that $\boldsymbol{\theta}^* = \mathbf{0}$ is an asymptotically stable equilibrium since all initial conditions $\boldsymbol{Z}(0)$ near $\boldsymbol{\theta}^*$ produce solutions $\boldsymbol{Z}(\tau)$ that stay near $\boldsymbol{\theta}^*$ and that converge to $\boldsymbol{\theta}^*$. The domain of attraction is all of $\mathbb{R}^2$ because for any $\boldsymbol{Z}(0) \in \mathbb{R}^2$, $\boldsymbol{Z}(\tau) \to \mathbf{0}$ as $\tau \to \infty$. Hence, the ODE aspects of conditions B.2 and B.3 are satisfied for this problem. ❑

Example 4.6—ODE for a nonlinear problem. Again, suppose that $p = 2$. Consider the nonlinear function

$$\boldsymbol{g}(\boldsymbol{\theta}) = \begin{bmatrix} 2 + 2t_1 + t_2 \exp(t_1 t_2) \\ 6t_2 + t_1 \exp(t_1 t_2) \end{bmatrix},$$

where $\boldsymbol{\theta} = [t_1, t_2]^T$. Let us check the ODE-based convergence conditions by solving the associated ODE for its critical points and inspecting the domain of attraction. Let $\boldsymbol{Z}(\tau) = [Z_1(\tau), Z_2(\tau)]^T$. Analogous to Example 4.5, the system of ODEs based on the form of $\boldsymbol{g}(\boldsymbol{\theta})$ is

$$\frac{d\boldsymbol{Z}}{d\tau} = -\begin{bmatrix} 2 + 2Z_1 + Z_2 \exp(Z_1 Z_2) \\ 6Z_2 + Z_1 \exp(Z_1 Z_2) \end{bmatrix}.$$

This system has no analytical solution, but it can be solved using standard numerical methods for ODEs. An isolated critical point (corresponding to $d\boldsymbol{Z}/d\tau = \mathbf{0}$) is found at $\boldsymbol{Z} = [-1.0643, 0.1510]^T$. This critical point is the unique solution $\boldsymbol{\theta}^*$. The domain of attraction for this point is all of $\mathbb{R}^2$, so the ODE aspects of conditions B.2 and B.3 are all satisfied. A phase plot for this system is shown in Figure 4.2. ❑

4.4 ASYMPTOTIC NORMALITY AND CHOICE OF GAIN SEQUENCE

Of central importance in SA is knowledge that the iterate $\hat{\boldsymbol{\theta}}_k$ will converge to a solution $\boldsymbol{\theta}^*$ as $k \to \infty$. Such knowledge ensures that $\hat{\boldsymbol{\theta}}_k$ gets to within a small neighborhood of $\boldsymbol{\theta}^*$ with enough computational and/or experimental resources devoted to the search process. However, convergence by itself gives no information about the speed with which the iterate approaches $\boldsymbol{\theta}^*$. To address the issue of convergence rate, this section discusses the probability distribution of the iterate. Knowledge of the distribution also provides insight into two related aspects of the algorithm: (i) error bounds for the iterate and (ii) guidance into the choice of the gain a_k so as to minimize the likely deviation of $\hat{\boldsymbol{\theta}}_k$ from $\boldsymbol{\theta}^*$.

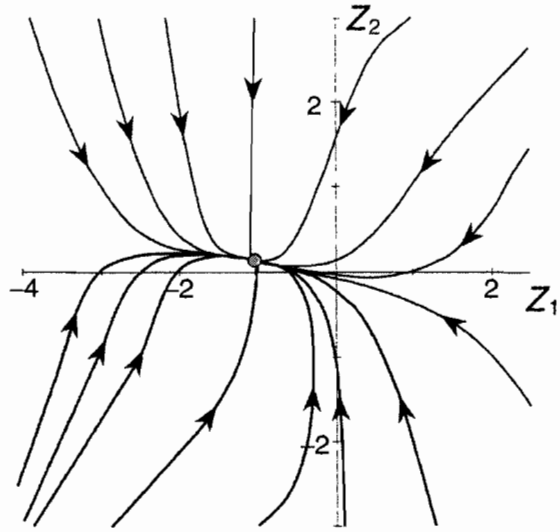

Figure 4.2. ODE convergence paths for Example 4.6. Each line depicts a path that $\boldsymbol{Z}(\tau) = [Z_1(\tau), Z_2(\tau)]^T$ follows over time from a particular initial condition $\boldsymbol{Z}(0)$. There is global convergence to $\boldsymbol{\theta}^* = [-1.0643, 0.1510]^T$ from any $\boldsymbol{Z}(0)$.

Unfortunately, for general nonlinear problems, there is no known finite-sample ($k < \infty$) distribution for the SA iterate. Further, the theory governing the asymptotic ($k \to \infty$) distribution is rather difficult. This is to be expected given the nonlinear transformations of the underlying random effects $\boldsymbol{e}_k$ that enter at each iteration. The nonlinear transformation is apparent by examining the forms in (4.2), (4.6), and (4.7): A value $\boldsymbol{e}_k$ at one iteration is transformed via the generally nonlinear mapping $\boldsymbol{g}(\cdot)$ in obtaining the next measurement $\boldsymbol{Y}_{k+1}$, and this measurement forms the basis for $\hat{\boldsymbol{\theta}}_{k+1}$, which, in addition to the new random effect $\boldsymbol{e}_{k+1}$, is the point of evaluation for $\boldsymbol{g}(\cdot)$ in the next iteration. Additional complications ensue when the noise itself is statistically dependent on the previous $\boldsymbol{\theta}$ estimates. Since this process is repeated for as many iterations as the algorithm takes, the nonlinear transformations of $\boldsymbol{e}_0$, $\boldsymbol{e}_1$, $\boldsymbol{e}_2$,... are usually very complex.

General results on the asymptotic distribution of the SA iterate are given in Fabian (1968). His work is a generalization of the first asymptotic distribution results for SA in Chung (1954) and Sacks (1958). Fabian shows that, under appropriate regularity conditions,

$$k^{\alpha/2}(\hat{\boldsymbol{\theta}}_k - \boldsymbol{\theta}^*) \xrightarrow{\text{dist.}} N(\mathbf{0}, \boldsymbol{\Sigma}) \tag{4.12}$$

as $k \to \infty$, where $\xrightarrow{\text{dist.}}$ denotes "converges in distribution" (as discussed in Appendix C), $\boldsymbol{\Sigma}$ is some covariance matrix that depends on the coefficients in the gain sequence a_k and on the Jacobian matrix of $\boldsymbol{g}(\boldsymbol{\theta})$ (i.e., the derivative $\boldsymbol{H}(\boldsymbol{\theta}) = \partial \boldsymbol{g}/\partial \boldsymbol{\theta}^T$), and α governs the decay rate for the SA gain a_k (e.g., $a_k = a/(k+1)^\alpha$). The intuitive interpretation of (4.12) is that $\hat{\boldsymbol{\theta}}_k$ is approximately normally distributed with mean $\boldsymbol{\theta}^*$ and covariance matrix $\boldsymbol{\Sigma}/k^\alpha$ for k reasonably large. Ruppert (1991) also discusses this result. Various special cases of this result dealing with the situation $\alpha = 1$ are presented in Rustagi (1994, pp. 258–259), Ljung et al. (1992, pp. 71–78), and Ruppert (1991).

Expression (4.12) implies that the rate at which the iterate $\hat{\boldsymbol{\theta}}_k$ approaches $\boldsymbol{\theta}^*$ is proportional in a stochastic sense to $k^{-\alpha/2}$ for large k. That is, allowing for random variation at each k, $\hat{\boldsymbol{\theta}}_k - \boldsymbol{\theta}^*$ decays at a rate proportional to $k^{-\alpha/2}$ to balance the $k^{\alpha/2}$ "blow-up" factor on the left-hand side of (4.12) and yield a well-behaved random vector with the distribution $N(\mathbf{0}, \boldsymbol{\Sigma})$ on the right-hand side of (4.12) (i.e., "well-behaved" in the sense of being neither degenerate 0 nor ∞ in magnitude). Under condition A.1 or B.1 on the gain a_k for convergence of the iterate (Theorem 4.1), the rate of convergence of $\hat{\boldsymbol{\theta}}_k$ to $\boldsymbol{\theta}^*$ is maximized at $\alpha = 1$ when the gain has the standard form $a_k = a/(k+1)^\alpha$. That is, the maximum rate of convergence for the root-finding SA algorithm under the general conditions above is $O(1/\sqrt{k})$ in an appropriate stochastic sense.

Further, through minimizing a norm of the matrix $\boldsymbol{\Sigma}$ appearing in (4.12), it is known (see, e.g., Benveniste et al., 1990, pp. 110–116) that the

asymptotically optimal gain for root-finding SA is a *matrix* gain given by the scaled inverse Jacobian matrix

$$\boldsymbol{a}_k = \frac{\boldsymbol{H}(\boldsymbol{\theta}^*)^{-1}}{k+1}, \quad k \geq 0. \tag{4.13}$$

Recall that this Jacobian matrix corresponds to the Hessian matrix of $L(\boldsymbol{\theta})$ if $\boldsymbol{g}(\boldsymbol{\theta})$ represents a gradient for an underlying problem of minimizing $L(\boldsymbol{\theta})$. With the exception of the decay factor $k+1$ included to damp out the noise effects, the gain in (4.13) is identical to the multiplier of $-\boldsymbol{g}(\boldsymbol{\theta})$ in the deterministic Newton–Raphson search of Section 1.4. With the gain given in (4.13), the limiting covariance matrix $\boldsymbol{\Sigma}$ is equal to the inverse of the average Fisher information matrix across the measurements (see Section 13.3 for a discussion of this matrix). The inverse Fisher information matrix is the smallest possible covariance matrix in the matrix sense discussed in Appendix A (so the optimal gain achieves this smallest possible covariance matrix).

Unfortunately, the gain in (4.13) is largely of theoretical interest only since in practice one does not know either $\boldsymbol{\theta}^*$ or the Jacobian matrix as a function of $\boldsymbol{\theta}$. It is also an asymptotic result, and, as discussed below, optimality for practical finite-sample analysis may impose other requirements. Nevertheless, this asymptotic result provides a type of ideal in designing adaptive SA algorithms (see Subsection 4.5.2 for a discussion of some practical implementations of (4.13)).

In practical problems, it may not be best to choose $\alpha = 1$. Most practitioners find that a lower value of α yields superior finite-sample behavior. This fact is also mentioned occasionally in the more-theoretical literature (e.g., Ruppert, 1991; Kushner and Yin, 1997, p. 328). The intuitive reason for the desirability of $\alpha < 1$ is that a slower decay provides a larger step size in the iterations with large k, allowing the algorithm to move in bigger steps toward the solution. This observation is a practical *finite-sample* result, as the asymptotic theory showing optimality of $\alpha = 1$ is unassailable.

Given the desirability for a gain sequence that balances algorithm stability in the early iterations with nonnegligible step sizes in the later iterations, a recommended gain form is

$$a_k = \frac{a}{(k+1+A)^{\alpha}}, \tag{4.14}$$

where the *stability constant* $A \geq 0$. Choosing $A = 0$ (i.e., no stability constant) is the most widely discussed form of gain sequence in the literature. However, there is a potential problem with $A = 0$. Choosing a large numerator a in the hopes of producing nonnegligible step sizes after the algorithm has been running awhile may cause unstable behavior in the early iterations (when the

denominator is still small). Choosing a small a leads to stable behavior in the early iterations but sluggish performance in later iterations. For this reason, picking $A > 0$ is usually recommended.

A strictly positive A allows for a larger a without risking unstable behavior in the early iterations. Then, in the later iterations, the A in the denominator becomes negligible relative to the k, while the relatively large a in the numerator helps maintain a nonnegligible step size. These larger step sizes often enhance practical convergence. For values of $1/2 < \alpha \leq 1$, a reasonable choice for the stability constant is to pick A such that it is approximately 5 to 10 percent of the total number of expected (or allowed) iterations in the search process. This defines the sense in which A is "negligible" for large k. Ruppert (1991) and Okamura et al. (1995) also consider the gain form including the stability constant; the latter reference demonstrates the improvement possible in a signal processing application of estimation in adaptive filters.

In fact, in many applications, a *constant* step size ($\alpha = 0$) is used as a way of avoiding gains that are too small for large k. Typical applications involve adaptive tracking or control problems where $\boldsymbol{\theta}^*$ is changing in time. The constant gain provides enough impetus to the algorithm to keep up with the variation in $\boldsymbol{\theta}^*$. In contrast, a decaying gain provides too little weight to the current input information to allow for the algorithm to track the solution. Such constant-gain algorithms are also frequently used in neural network training even when there is no variation in the underlying $\boldsymbol{\theta}^*$ (White, 1989; Kuan and Hornik, 1991). As discussed with LMS (Section 3.2), algorithms with constant step sizes will generally not formally converge.[2] Also, note that the limiting distribution for the standardized SA iterate (analogous to the left-hand side of (4.12)) is *not* generally normal with constant step sizes (Mukherjee and Fine, 1996; Pflug, 1986).

Another common ad hoc "trick" to avoid potentially sluggish behavior is to periodically restart the algorithm. That is, after starting the search with a bona fide initial condition $\hat{\boldsymbol{\theta}}_0$, periodically reset the current estimate $\hat{\boldsymbol{\theta}}_k$ to $\hat{\boldsymbol{\theta}}_0$ while restarting the gain sequence at a_0. Ruppert (1991) provides an interpretation of the stability constant in (4.14) in the context of such a restarting method.

---

[2] A form of convergence theory is possible for constant gains. This is typically based on limiting arguments as the gain magnitude gets small. Essentially, one is able to show that the iterate from a constant-gain algorithm will approach the optimal $\boldsymbol{\theta}$ to within some error that decreases as the gain magnitude is made smaller (see, e.g., Macchi and Eweda, 1983; Kushner and Huang, 1983; Pflug, 1986). This was also discussed in the context of LMS in Subsection 3.2.2. Another form of convergence that can cope with nondecaying gains is the *weak convergence* approach. This form of convergence is a generalization of convergence in distribution. As discussed in Kushner and Yin (1997, Chaps. 7 and 8) and Yin and Yin (1996), this form allows one to say that with high probability, the iteration process will emulate the random behavior of the solution to a stochastic differential equation with Wiener noise input. A detailed discussion of weak convergence is beyond the scope of this book.

4.5 EXTENSIONS TO BASIC STOCHASTIC APPROXIMATION

The four subsections here discuss some extensions to the basic SA framework presented above. Subsection 4.5.1 considers the setting where the observation $\boldsymbol{Y}_k = \boldsymbol{Y}_k(\hat{\boldsymbol{\theta}}_k)$ includes a state vector that evolves as $\boldsymbol{\theta}$ is being updated. Subsection 4.5.2 discusses acceleration methods through intelligent gain selection and/or adaptive choices of whether to accept a new iteration value. Subsection 4.5.3 introduces iterate averaging for SA as a means for accelerating convergence and Subsection 4.5.4 considers the setting where the function $\boldsymbol{g}(\cdot)$ may change with time.

4.5.1 Joint Parameter and State Evolution

Consider an important special case of (4.3), where the input $\boldsymbol{x}_k$ represents a state vector related to the system being optimized. There is a statistical model governing the evolution of $\boldsymbol{x}_k$ in this special case, and this model is dependent on previous values of the estimated $\boldsymbol{\theta}$. Much of the book by Benveniste et al. (1990) is devoted to this framework, which was also considered by Ljung (1977) and Metivier and Priouret (1984). We may view $\boldsymbol{x}_k$ as a system observation evolving according to certain restrictions on the conditional probability $P(\boldsymbol{x}_{k+1} \in S \,|\, \hat{\boldsymbol{\theta}}_0, \hat{\boldsymbol{\theta}}_1, ..., \hat{\boldsymbol{\theta}}_k; \boldsymbol{x}_0, \boldsymbol{x}_1, ..., \boldsymbol{x}_k)$ for some set S. The convergence of $\hat{\boldsymbol{\theta}}_k$ (similar to that discussed in Section 4.3) then depends on the properties of this conditional probability. An important special case is when this conditional probability is a Markov process (Appendix E). Then, the convergence of $\hat{\boldsymbol{\theta}}_k$ is tied to the stationarity of the Markov process. Benveniste et al. (1990, Part I, Chaps. 1 and 4) discuss several applications of this framework in the context of telecommunications, fault detection, and signal processing.

One of the common representations for the evolution of $\boldsymbol{x}_k$ is a state equation that is linear in $\boldsymbol{x}_k$ (as in Subsection 3.3.4):

$$\boldsymbol{x}_{k+1} = \boldsymbol{A}(\hat{\boldsymbol{\theta}}_k)\boldsymbol{x}_k + \boldsymbol{B}(\hat{\boldsymbol{\theta}}_k)\boldsymbol{w}_k, \tag{4.15}$$

where $\boldsymbol{A}(\cdot)$ and $\boldsymbol{B}(\cdot)$ are appropriately dimensioned matrices and $\boldsymbol{w}_k$ is a sequence of independent random vectors (see, e.g., Ljung, 1977). In this case, the full SA algorithm is the recursion for $\boldsymbol{\theta}$ in (4.6) or (4.7) together with the state equation (4.15). As noted by Benveniste et al. (1990, p. 27), the coupling of a recursion for $\boldsymbol{\theta}$ together with (4.15) is a common form in the identification of linear systems. Then the evolution of $\boldsymbol{x}_k$ (and hence behavior of the SA recursion for $\boldsymbol{\theta}$) can be tied directly to the stability of the state equation. In particular, in the special constant-coefficient case where $\boldsymbol{A}$ is independent of $\boldsymbol{\theta}$, the well-known requirement that the eigenvalues of $\boldsymbol{A}$ lie inside the unit circle (the circle of radius 1 about the origin) guarantees the existence of the required stationary transition probabilities for the above-mentioned Markov process.

4.5.2 Adaptive Estimation and Higher-Order Algorithms

There are a large number of methods for adaptively estimating the gain a_k (or multivariate analogue of the gain). The aim is to enhance the convergence rate of the SA algorithm. Some of the results rely on choosing the gains adaptively depending on recent information acquired by the algorithm in its search process. Other results are built on SA analogues of the Newton–Raphson search in Section 1.4. In particular, some of the results are aimed at adaptively estimating the Jacobian (or Hessian) matrix in the asymptotically optimal gain $\boldsymbol{a}_k = \boldsymbol{H}(\boldsymbol{\theta}^*)^{-1}/(k+1)$, $k \geq 0$, discussed in Section 4.4 (eqn. (4.13)).

One of the first adaptive techniques is given in Kesten (1958), which is based on the signs ($\pm$) of the differences $\hat{\theta}_{k+1} - \hat{\theta}_k$ in a scalar θ process as a means of designing an adaptive gain sequence a_k. This approach does not explicitly use the connection to the Jacobian matrix as mentioned above. If there are frequent sign changes, this is an indication that the iterate is near θ^*; if the signs are not changing, this is an indication that the iterate is far from θ^*. This forms the basis for an adaptive choice of the gain a_k, where a larger gain is used if there are no sign changes and a smaller gain is used if the signs change frequently. Kesten (1958) established a.s. convergence to θ^* with such a scheme. A multivariate extension (including theoretical justification) of the Kesten idea is given in Delyon and Juditsky (1993).

There exist a number of stochastic analogues of the Newton–Raphson search in the context of parameter estimation for *particular* (possibly linear) models. The scalar gain a_k is then replaced by a matrix that approximates the (unknown) true inverse of the Jacobian (Hessian) matrix. Ljung and Soderstrom (1983, Sects. 2.4 and 3.4, Chap. 4), Macchi and Eweda (1983), Benveniste et al. (1990, Part I, Chaps. 3 and 4), Ljung et al. (1992, Part III), Yin and Zhu (1992), and Ljung (1999, Sect. 11.6) discuss some of the philosophy and mechanics of such adaptive approaches in various special cases.

While the above methods for special cases are effective ways to increase convergence speed, they are restricted in their range of application. More general approaches are described, for example, in Nevel'son and Khas'minskii (1973), Ruppert (1985), and Wei (1987) (note: due to inconsistent Russian translations, Khas'minskii here is the same as Has'minskii in the 1973 book cited in Sections 4.1 and 4.3). The problem formulation used by Ruppert (1985) differs slightly from the standard root-finding form in that he converts the basic problem from one of finding the root to $\boldsymbol{g}(\boldsymbol{\theta}) = \boldsymbol{0}$ to one of minimizing $\|\boldsymbol{g}(\boldsymbol{\theta})\|^2$ (as discussed in Section 1.1, this conversion yields the same $\boldsymbol{\theta}^*$ when there is a unique root).

In both Ruppert (1985) and Wei (1987), each column of the Jacobian matrix is approximated at the current value of $\boldsymbol{\theta}$ by perturbing one of the components of $\boldsymbol{\theta}$ in a positive and negative direction and evaluating a (noisy) $\boldsymbol{g}(\boldsymbol{\theta})$ at each of those two perturbed $\boldsymbol{\theta}$ values. This finite-difference process is repeated for all elements of $\boldsymbol{\theta}$ to get a full Jacobian matrix. In the process of

producing each column, some averaging can be used to smooth out the noise effects. With such a scheme, Ruppert (1985) and Wei (1987) give conditions for a.s. convergence and asymptotic normality of the iterate (analogous to results in Sections 4.3 and 4.4). The author is unaware of any numerical evaluation of this type of approach.

Recent results in Spall (2000) suggest an easier and more efficient way of estimating the Jacobian using the ideas of simultaneous perturbation discussed in Chapter 7. At each iteration, only *two* noisy measurements of $\boldsymbol{g}(\boldsymbol{\theta})$ are required to estimate the Jacobian matrix, irrespective of the dimension p. This contrasts with a number of measurements of order of p in the finite-difference-based approaches above. Section 7.8 discusses this adaptive SA approach in detail. This use of only two measurements can lead to a large cost savings (i.e., fewer $\boldsymbol{g}(\boldsymbol{\theta})$ measurements) when p is large.

4.5.3 Iterate Averaging

Iterate averaging is an important and relatively recent development in SA. Like many good ideas in science and engineering, this idea is simple, and, in some problems, can be very effective. Ruppert (1991, based on an earlier 1988 internal technical report) and Polyak and Juditsky (1992) jointly introduced iterate averaging, together with the key supporting theory. There are several variations, but the basic idea is to replace $\hat{\boldsymbol{\theta}}_k$ as the final best estimate of $\boldsymbol{\theta}$ with the average

$$\bar{\boldsymbol{\theta}}_k \equiv (k+1)^{-1} \sum_{j=0}^{k} \hat{\boldsymbol{\theta}}_j \tag{4.16}$$

where each of the $\hat{\boldsymbol{\theta}}_j$ summands in (4.16) is computed as in (4.6) or (4.7). The basic SA recursion (4.6) or (4.7) does not change (i.e., $\bar{\boldsymbol{\theta}}_k$ is not used in (4.6) or (4.7)); $\bar{\boldsymbol{\theta}}_k$ is only used as the final estimate in place of $\hat{\boldsymbol{\theta}}_k$. Ruppert (1991) and Polyak and Juditsky (1992) show that $\sqrt{k}(\bar{\boldsymbol{\theta}}_k - \boldsymbol{\theta}^*)$ is asymptotically normally distributed with mean zero and a covariance matrix as small as possible in an appropriate statistical sense. That is, the limiting covariance matrix is the inverse of the average Fisher information matrix across the measurements (see Section 13.3); this is identical to the covariance matrix based on the optimal gain in (4.13).

The optimality of (4.16) holds for gain sequences satisfying the standard conditions mentioned in Section 4.3 plus the condition $a_{k+1}/a_k = 1 - o(a_k)$ (with "little-o" implying a term that goes to zero faster than the argument; note that $o(a_k)$ is a positive term in the standard case of a monotonically decaying gain). This important additional condition implies that a_k must decay at a rate *slower* than the asymptotically optimal rate of $O(1/k)$ for the individual iterates $\hat{\boldsymbol{\theta}}_k$ (as discussed in Section 4.4; see Exercise 4.17).

The implications of the above are quite impressive. Namely, one can use the standard algorithm in (4.6) together with the simple averaging in (4.16) to achieve the same optimal rate of convergence that otherwise is possible only with exact knowledge (or a convergent estimate) of the Jacobian matrix $\boldsymbol{H}(\boldsymbol{\theta}^*)$. This asymptotic optimality is available for *any* gain satisfying the above-mentioned general conditions. Hence, in principle, iterate averaging greatly reduces one of the traditional banes of SA—that of choosing the gain sequence in some "good" way.

Variations on the basic iterate averaging approach are readily available. One obvious one is to not include the first few iterations in the average, instead starting the average at some $N > 0$ or else using only a sliding window of the last $k - N$ (say) measurements. In practical implementations, such modifications are likely to help since the first few iterations tend to produce the poorest estimates. In the sliding window approach, formal asymptotic optimality can be shown if the window length grows with time (see, e.g., Kushner and Yang, 1993; Kushner and Yin, 1997, Chap. 11). A further modification to the basic approach is to use the averaged value $\bar{\boldsymbol{\theta}}_k$ (together with $\hat{\boldsymbol{\theta}}_k$) in a modified form of the SA iteration (instead of $\hat{\boldsymbol{\theta}}_k$ alone on the right-hand side of the basic form (4.6) or (4.7)). This is the feedback approach in Kushner and Yang (1995) (see also Kushner and Yin, 1997, Chap. 11), which can be shown to yield further improvement in certain situations.

In practice, however, the results on iterate averaging are more mixed than the above would suggest. While some numerical results have shown considerable promise (e.g., Yin and Zhu, 1992; Kushner and Yin, 1997, Chap. 11), other numerical studies (e.g., Spall and Cristion, 1998; Spall, 2000) have shown that a reasonable tuned a_k sequence often yields results superior to those possible by averaging, including averaging using the same tuned gains (see also Wang, 1996, p. 57 for some cautionary notes). This is not surprising upon reflection as a consequence of the *finite-sample* properties of practical problems.

For iterate averaging to be successful, it is necessary that a large proportion of the individual iterates hover in some balanced way around $\boldsymbol{\theta}^*$, leading to the sample mean of the iterates being nearer $\boldsymbol{\theta}^*$ than the bulk of the individual iterates. A well-designed (stable) SA algorithm will not be jumping approximately uniformly around $\boldsymbol{\theta}^*$ when the iterates are far from the solution (else it is likely to diverge). The only way for the bulk of the iterates to be distributed uniformly around the solution is for the individual iterates to be near the solution, where *near* here is relative to the level of noise in the problem. That is, other things being equal, the domain in which the iterates begin bouncing roughly uniformly around $\boldsymbol{\theta}^*$ gets smaller as the level of noise gets smaller.

With a well-designed algorithm, $\hat{\boldsymbol{\theta}}_k$ in most practical settings moves in a way that *approximately* decreases the distance to $\boldsymbol{\theta}^*$ in a monotonic manner. The "approximately" clause follows partly from the inherent stochastic variability. A user terminates the algorithm when either the budget of iterations has been exceeded or when $\hat{\boldsymbol{\theta}}_k$ begins to move very slowly near (one hopes!) $\boldsymbol{\theta}^*$. The

latter situation is precisely when iterate averaging *starts* to work well. In fact, while the algorithm is in its monotonic phase, iterate averaging tends to *hurt* the accuracy of those components in $\hat{\boldsymbol{\theta}}_k$ that have not yet settled near $\boldsymbol{\theta}^*$!

Figure 4.3 illustrates this dichotomy on a search path for a typical $p = 2$ problem. Figure 4.3(a) shows a standard case of termination without the iterate bouncing around $\boldsymbol{\theta}^*$. Figure 4.3(b) depicts a case where the algorithm is not terminated until the iterate oscillates around the solution for some time. Case (b) is favorable to iterate averaging.

A further contrast of iterate averaging with an optimal algorithm based on the Jacobian matrix or an approximation (Subsection 4.5.2) is that the transform invariance property of Newton–Raphson-type algorithms (Section 1.4) may enhance convergence by improving the search direction when the magnitudes of the $\boldsymbol{\theta}$ components differ significantly. This improvement may occur without the iterate having to bounce around $\boldsymbol{\theta}^*$ (i.e., when the iteration process is in the monotonic phase mentioned above). That is, the optimal algorithm based on the Jacobian matrix does not require the algorithm to run long enough so that it is bouncing around $\boldsymbol{\theta}^*$. (However, the algorithm must run long enough to obtain a good estimate when the Jacobian matrix is being estimated.) This suggests that despite the simplicity and asymptotic justification, iterate averaging in practical finite-sample problems *may* not achieve the efficiency of the optimal algorithm based on the Jacobian matrix.

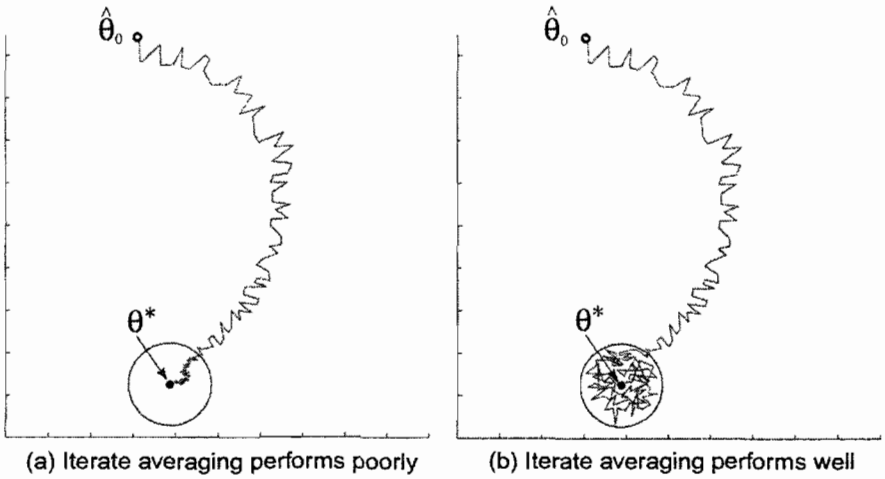

Figure 4.3. Two search paths for a $p = 2$ problem. Note *approximate* monotonic improvement in $\hat{\boldsymbol{\theta}}_k$ until search reaches circled area. Iterate average $\bar{\boldsymbol{\theta}}_k$ is usually poorer estimate than $\hat{\boldsymbol{\theta}}_k$ before search process enters circled area. If process does not oscillate about $\boldsymbol{\theta}^*$ (case (a)), terminal iterate average $\bar{\boldsymbol{\theta}}_k$ will be poorer estimate than $\hat{\boldsymbol{\theta}}_k$. If process oscillates around $\boldsymbol{\theta}^*$ as in case (b), then terminal $\bar{\boldsymbol{\theta}}_k$ is likely to provide better estimate than $\hat{\boldsymbol{\theta}}_k$ if process is allowed to run long enough inside circled area. Case (a) is commonly associated with low noise, while case (b) may be seen with high noise.

4.5.4 Time-Varying Functions

A further generalization is one where the root-finding function varies with k. Among other references, this problem is formally treated in Goodsell and Hanson (1976) and Evans and Weber (1986). The basic idea is that, while the function for which one is finding a root may change shape with k, it is assumed that the underlying root of $\boldsymbol{g}_k(\boldsymbol{\theta}) = \mathbf{0}$, say $\boldsymbol{\theta}_k^*$, is either constant for all k or reaches a limiting value as $k \to \infty$ (even though $\boldsymbol{g}_k(\boldsymbol{\theta})$ may change shape indefinitely). The two references mentioned above show a.s. convergence in these cases for the scalar θ setting; the proofs have to be changed from those commonly seen in SA to accommodate the time-varying loss. A multivariate extension is in Spall and Cristion (1998) in the context of nonlinear adaptive control. As an example of time-varying functions, Section 4.1 discussed the common situation where there are inputs $\boldsymbol{x}_k$ at each noisy evaluation of the function. In particular, $\boldsymbol{g}_k(\boldsymbol{\theta}) \equiv \tilde{\boldsymbol{g}}_k(\boldsymbol{\theta}, \boldsymbol{x}_k)$ for some function $\tilde{\boldsymbol{g}}_k$.

Figure 4.4 depicts the concept for the scalar ($p = 1$) case where $g_k(\theta)$ represents the gradient in a problem of minimizing time-varying loss functions $L_k(\theta)$. This figure depicts a problem where the time-varying loss (and corresponding gradient) functions never settle down, but where the time-varying solution θ_k^* does approach a limiting value θ_∞^*. The above-mentioned theory applies in such a setting, leading to an SA algorithm with an a.s. convergent $\hat{\theta}_k$ (i.e., $\hat{\theta}_k \to \theta_\infty^*$ a.s. as $k \to \infty$).

The time-varying setting in Figure 4.4 arises frequently in control problems. The time-varying loss follows when the target for the system performance may be perpetually varying, even when the underlying system dynamics remain essentially unchanged. Let us sketch an example of an application of this setting to a control problem.

Example 4.7—Sketch of implementation in adaptive control. Suppose that a system output vector $\boldsymbol{z}_k$ is modeled as $\boldsymbol{z}_k = \boldsymbol{h}(\boldsymbol{\theta}, \boldsymbol{x}_k) + \boldsymbol{v}_k$, where $\boldsymbol{h}(\cdot)$ is a representation of the process under study and $\boldsymbol{v}_k$ is the noise (a nonlinear analogue of the linear setting of Chapter 3). The input $\boldsymbol{x}_k$ represents some input (control) value that is being set to try to make $\boldsymbol{z}_k$ perform in a desired way.

A common specific case of a control problem is target tracking, where the aim is to have $\boldsymbol{z}_k$ follow a time-varying desired value $\boldsymbol{d}_k$. As part of determining the best input values $\boldsymbol{x}_k$, it is common to have to estimate $\boldsymbol{\theta}$. (As mentioned in Subsection 3.2.3, *indirect adaptive control* refers to this process of estimating model parameters as a vehicle to obtain optimal control inputs; in contrast, *direct adaptive control* is where the control inputs are determined directly without estimating the model.) In target tracking, one faces an estimation problem associated with minimizing a time-varying error

$$L_k(\boldsymbol{\theta}) = E\left(\|\boldsymbol{z}_k - \boldsymbol{d}_k\|^2\right).$$

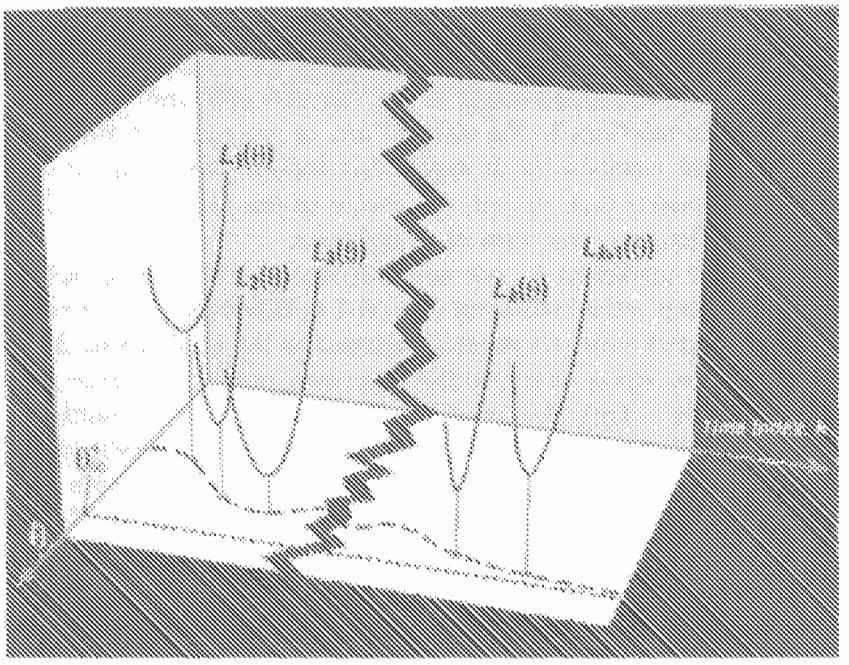

Figure 4.4. Time-varying $L_k(\theta)$ (leading to time-varying $g_k(\theta) = \partial L_k(\theta)/\partial\theta$) for minimization problem with limiting minimum at θ_∞^*.

The function $\boldsymbol{g}_k(\boldsymbol{\theta})$ represents a gradient of the above time-varying loss. Based on noisy measurements of $\boldsymbol{g}_k(\boldsymbol{\theta})$ at each k, SA may be used to find the estimate for $\boldsymbol{\theta}$ according to $\boldsymbol{g}_k(\boldsymbol{\theta}) = \mathbf{0}$. If it is assumed that $\boldsymbol{h}(\boldsymbol{\theta}_{\text{true}}, \boldsymbol{x})$ represents the true process, then, under modest conditions, $\boldsymbol{\theta}_k^* = \boldsymbol{\theta}_{\text{true}}$ for all k. That is, the best value for $\boldsymbol{\theta}$ in the *model* (i.e., the value minimizing $L_k(\boldsymbol{\theta})$) is the true value for $\boldsymbol{\theta}$. As a simple example, suppose that both the model and the true system are represented by the scalar polynomial $h(\boldsymbol{\theta}, x) = b_0 + b_1 x + b_2 x^2$, where $\boldsymbol{\theta} = [b_0, b_1, b_2]^T$. Then, $\boldsymbol{\theta}_k^* = \boldsymbol{\theta}_{\text{true}}$ for all k since there is one set of true b_0, b_1, and b_2. This constant solution for all k applies even though $L_k(\boldsymbol{\theta})$ and $\boldsymbol{g}_k(\boldsymbol{\theta})$ may be perpetually changing. ❑

4.6 CONCLUDING REMARKS

The root-finding stochastic approximation framework is a cornerstone of stochastic search and optimization. Many popular search methods are special cases of root-finding SA. Moreover, there is a deep theory associated with the convergence of SA algorithms. Because of the web of connections to many other methods, the SA theory has broad implications for the performance of general stochastic search and optimization algorithms, especially in cases with only noisy measurements for use in the search process.

This chapter discussed two general approaches to convergence analysis of SA—the "statistics" and "engineering" approaches. The former relies on classical assumptions about the shape of the function $\boldsymbol{g}(\theta)$ and the noise. The latter is built on connections to the stability and convergence of an associated ordinary differential equation. Both approaches include conditions on the all-important gain sequence (a_k). In fact, the choice of the gain sequence remains one of the key challenges in many practical problems.

Some of the conditions for convergence may be difficult to check and/or difficult to satisfy in practical problems (e.g., A.2 or B.2 and B.3 in Section 4.3). Nonetheless, the theory provides general guidance on the expected performance of SA, and, with the appropriate qualifications, may provide guidance in cases that do not entirely satisfy the conditions (e.g., guidance in multiple root problems provided that the search is restricted to certain neighborhoods of the search space Θ). The results of this chapter do not *directly* apply with multiple local minima when used in an optimization context. Extensions of SA to the global optimization problem (introduced in Section 1.1) are discussed in Sections 7.7 and 8.4.

There have been countless applications of SA in the greater than half century since the seminal publication of Robbins and Monro (1951). This chapter touched on only a few of these (univariate and bivariate quantile estimation, model estimation, and adaptive control). Some areas not discussed in detail in this chapter include neural network training, simulation-based optimization, evolutionary algorithms, machine learning, experimental design, and signal processing applications such as noise cancellation and pattern recognition. Some of these are discussed elsewhere in this book. Benveniste et al. (1990, Part I), Kushner and Yin (1997, Chaps. 2 and 3), Yin (2002), and some of the other references cited are among the publications that summarize these and other applications.

While all of the remaining chapters in this book involve at least *some* connection to SA, the next three chapters focus *directly* on SA, especially as related to optimization problems. The range of applications for SA now extends well beyond those envisioned by some of the pioneers in the field. This blossoming is expected to continue as technical challenges exceed the reach of classical deterministic search and optimization methods.

EXERCISES

4.1 Based on (4.3) and $p = 1$, suppose that $Y_k(\theta) = \tilde{g}_k(\theta, \boldsymbol{x}_k) = x_{k1}\theta - \sin(x_{k1} + x_{k2})$, where $\boldsymbol{x}_k = [x_{k1}, x_{k2}]^T$ is uniformly distributed over $[0, \pi/2] \times [0, \pi/2]$ for all k (note that $\tilde{e}_k(\theta, \boldsymbol{x}_k) = 0$). Further, suppose that $E[e_k(\theta)] = 0$ for all k. What are $g(\theta)$ and θ^*?

4.2 Consider the problem of minimizing $L(\boldsymbol{\theta}) = t_1^4 + t_1^2 + t_1 t_2 + t_2^2$, $\boldsymbol{\theta} = [t_1, t_2]^T$. Suppose that the loss gradient $\boldsymbol{g}(\boldsymbol{\theta})$ can only be measured in the presence of

independent $N(\mathbf{0}, \sigma^2 \boldsymbol{I}_2)$ noise (the $\boldsymbol{e}$ term). In particular, using the two gain sequences, $a_k = 0.1/(k+1)$ and $a_k = 0.1/(k+1)^{0.501}$, $k = 0, 1, \ldots$, compare the mean terminal loss function value after 1000 iterations using 200 realizations at each gain sequence (it is not necessary to perform a statistical test in this comparison). Take $\sigma = 0.1$ and $\hat{\boldsymbol{\theta}}_0 = [1, 1]^T$. Repeat the above for the case where $\sigma = 1.0$. What can you say about the relative performance of the two gain sequences for each of the two noise levels?

4.3 For the problem of estimating μ from a sample of independent measurements, consider the recursion in Example 4.1, $\hat{\theta}_{k+1} = \hat{\theta}_k - a_k Y_k(\hat{\theta}_k)$, where $Y_k(\hat{\theta}_k) = \hat{\theta}_k - X_{k+1}$. In contrast to Example 4.1, however, now suppose that $a_k = 0.25/(k+1)$. Calculate 20 replications of $n = 100$, 10,000, and 1,000,000 iterations, where X_k is uniformly distributed over (0, 2) (i.e., $X_k \sim U(0, 2)$) for all k. For each n, report the sample mean of the terminal estimate for μ from the 20 replications together with an approximate 95 percent confidence interval (using the t-distribution). Comment on the observed results relative to those expected from $a_k = 1/(k+1)$, as in Example 4.1.

4.4 In Example 4.2 on quantile estimation, assume that experimental process X is governed by a $U(-1, 1)$ distribution. Suppose that $\hat{\theta}_0 = 0.6$.

(a) Estimate the LD_{50} point using $a_k = 1/(k+1)$ and $a_k = 1/(k+1)^{0.501}$. For each of the two gains, use 100 Monte Carlo replications with $n = 200$ experiments (number of X values) for each replication. Use a different random number seed to initialize each of the sets of 100 Monte Carlo replications. Applying the appropriate unmatched pairs two-sample test in Appendix B, can you draw any valid statistical conclusion about the relative performance of the SA algorithm with the two gain sequences?

(b) After some tuning experiments, choose a new gain sequence of the form in (4.14). Generate another 100 replications with $n = 200$. Does this new gain sequence yield a statistically significant improvement?

4.5 Implement the root-finding procedure in the stochastic gradient mode of Example 4.4 to estimate the parameters $\boldsymbol{\theta} = [\lambda, \beta]^T$. For each replication as requested below, generate data on a grid defined by the integer pairs (C, W) where $C \in [1, 10]$ and $W \in [11, 110]$ (e.g., (5, 15), (4, 87), (8, 33), etc.). Perform 1000 iterations by randomly sampling (uniformly) the 1000 grid points without replacement until all 1000 points have been used. Let $\boldsymbol{\theta}^* = [2.5, 0.7]^T$ and use (4.8) to compute the output z_k with random error v_k defined below. Use $\hat{\boldsymbol{\theta}}_0 = [1.0, 0.5]^T$ as the starting point and $a_k = 0.0015/(k+100)^{0.501}$ as the gain. Consider two distributions for the noise in the measurements z_k: (i) normal error where $v_k \sim N(0, 5^2)$ and (ii) dependent error where $v_k = 0.2h(\boldsymbol{\theta}^*, \boldsymbol{x}_k)w_k$ with $w_k \sim N(0, 1)$ and $\boldsymbol{x}_k = [W_k, C_k]^T$. Do five replications (each of 1000 iterations as described above) for each of the two noise distributions. Compute the distance of the final estimate to $\boldsymbol{\theta}^* = [2.5, 0.7]^T$ for each replication. Report the results for each replication.

4.6 It was stated in Section 4.3 that if $\text{cov}(\boldsymbol{e}_k) \geq \eta \boldsymbol{I}_p$ for some $\eta > 0$, then $E(\|\boldsymbol{e}_k\|^2) \geq \eta p$ when $\boldsymbol{b}_k = \boldsymbol{0}$. Prove this result by using the definition of positive semidefiniteness in matrix relationship (xiii) of Appendix A.

4.7 Show that the condition $\sup_{k \geq 0} \|\hat{\boldsymbol{\theta}}_k\| < \infty$ a.s. in B.3 of Section 4.3 does *not* imply that $\|\hat{\boldsymbol{\theta}}_k\|$ is *uniformly* bounded in magnitude (a.s.) for all k. (This exercise requires a knowledge of measure-theoretic probability at the level of Appendix C.)

4.8 Suppose $p = 2$ and $\boldsymbol{g}(\boldsymbol{\theta}) = \begin{bmatrix} 4 & 0 \\ 0 & 2 \end{bmatrix} \boldsymbol{\theta}$. Based on the associated ODE (4.9), determine whether $\boldsymbol{\theta}^*$ is an asymptotically stable equilibrium and determine the domain of attraction.

4.9 Prove that the indicated solution $\boldsymbol{Z}(\tau)$ in Example 4.5 is the solution to the associated ODE.

4.10 Consider the function $\boldsymbol{g}(\boldsymbol{\theta}) = \left[t_1^2 - t_1 - t_1 t_2 + t_2^2/2,\ t_2^2 - 2t_2 + t_1 t_2 - t_1^2/2\right]^T$, $\boldsymbol{\theta} = [t_1, t_2]^T$. Numerically or analytically identify the candidate points $\boldsymbol{\theta}^*$ where $\boldsymbol{g}(\boldsymbol{\theta}) = \boldsymbol{0}$. For each of these points, show that $\boldsymbol{g}(\boldsymbol{\theta})$ fails convergence condition A.2 when $\boldsymbol{B} = \boldsymbol{I}_2$.

4.11 Consider the function $\boldsymbol{g}(\boldsymbol{\theta})$ in Exercise 4.10. Write the associated ODE, and identify the points where $d\boldsymbol{Z}/d\tau = \boldsymbol{0}$. Show that only one of these points is a (locally) stable equilibrium for the ODE (if $\boldsymbol{Z}(\tau)$ is sufficiently close to this equilibrium for any τ, then $\boldsymbol{Z}(\tau)$ converges to this point as $\tau \to \infty$). Graphically identify the approximate domain of attraction for this equilibrium. Explain why $\boldsymbol{g}(\boldsymbol{\theta})$ violates convergence condition B.2, but intuitively why the ODE analysis indicates that convergence may still be possible with appropriate additional restrictions.

4.12 Verify convergence condition A.2 for the $\boldsymbol{g}(\boldsymbol{\theta})$ in Example 4.6.

4.13 Identify at least one gain form *other* than (4.14) that satisfies condition A.1 together with the range of coefficient values for this form.

4.14 Let $\boldsymbol{\theta} = [t_1, t_2, \ldots, t_{20}]^T$ and $\boldsymbol{g}(\boldsymbol{\theta}) = [4t_1^3, \ldots, 4t_{10}^3; 0, \ldots, 0]^T + 2\boldsymbol{B}\boldsymbol{\theta}$, where $\boldsymbol{B}$ is a symmetric matrix with 1's on the diagonals and 0.5's elsewhere. Measurements $\boldsymbol{Y}(\boldsymbol{\theta}) = \boldsymbol{g}(\boldsymbol{\theta}) + \boldsymbol{e}$ are available to the algorithm of interest, where $\boldsymbol{e}$ is i.i.d. $N(\boldsymbol{0}, \sigma^2 \boldsymbol{I}_{20})$ distributed. Let us compare the unconstrained and constrained algorithms (4.6) and (4.7). Use a gain sequence $a_k = a/(k+1+A)^{0.501}$, $a = 0.01$, and $A = 100$, with $\hat{\boldsymbol{\theta}}_0 = 0.1 \times [1, 1, \ldots, 1]^T$. For $\sigma = 20$ and $\sigma = 1$, compare the unconstrained and constrained results. For (4.7), assume that $\Theta = [-0.1, 0.1]^{20}$ and that the projection Ψ_Θ simply moves any component of $\boldsymbol{\theta}$ lying outside of $[-0.1, 0.1]$ back to the nearest endpoint of the interval. Based on 25 replications and 1000 iterations/replication, use the observed RMS error in the terminal $\boldsymbol{\theta}$ estimate (relative to $\boldsymbol{\theta}^*$) as the basis for comparison (i.e., the square root of the average terminal MSE across the 25 replications).

4.15 For the setting of Exercise 4.2 with $\sigma = 0.1$, implement the iterate averaging form of SA discussed in Subsection 4.5.3 with $a_k = 0.1/(k+1)^{0.501}$. Use both the basic averaging approach in (4.16) together with a sliding window

of the most recent 100 iterates. Compare these two iterate averaging approaches with each other and with the results from a standard SA (no iterate averaging) using the available loss function (not always available in root-finding problems!). In particular, calculate the sample mean of the terminal loss values from 50 realizations of 1000 iterations per realization for each of the three SA implementations.

4.16 Consider the function of Exercise 4.14 with $\sigma = 0.1$. Implement the iterate averaging form of SA discussed in Subsection 4.5.3 based on $a_k = a/(k+1+A)^{0.501}$, $a = 0.2$, and $A = 100$, with $\hat{\boldsymbol{\theta}}_0 = [1, 1, \ldots, 1]^T$. Use both the basic averaging approach in (4.16) together with a sliding window of the most recent 100 iterates. Compare these two iterate averaging approaches with each other and with the results from a standard SA (no iterate averaging). In particular, based on 25 replications and 1000 iterations/replication, calculate the sample RMS error in the terminal $\boldsymbol{\theta}$ estimate (relative to $\boldsymbol{\theta}^*$) for each of the three SA implementations (i.e., the square root of the sample mean of the squared errors across the 25 replications).

4.17 For gains of the form $a_k = a/(k+1)^{\alpha}$, $a > 0$, $\alpha > 0$, show that $a_{k+1}/a_k = 1 - o(a_k)$ implies that $\alpha < 1$ (relevant for iterate averaging in Subsection 4.5.3).