You can observe a lot just by watching.

Berra's Law - Yogi Berra,
US baseball player, coach,
and manager (1925 -)

3

Learning and Optimization with Perturbation Analysis

As shown in Chapter 2, performance derivatives for Markov systems depend heavily on performance potentials. In this chapter, we first discuss the numerical methods and sample-path-based algorithms for estimating performance potentials, and we then derive the sample-path-based algorithms for estimating performance derivatives. In performance optimization, the process of estimating the potentials and performance derivatives from a sample path is called *learning*.

Policy gradients (PG) in reinforcement learning (RL) is almost a synonym for perturbation analysis (PA) in discrete event dynamic systems (DEDS). However, because the terms PG and PA are used by researchers in two different disciplines, there is a different emphasis on different aspects of the analysis. With PA in DEDS, we construct sensitivity formulas by exploring the system's dynamic nature and develop sample-path-based and on-line estimation algorithms for performance derivatives; while with PG in RL we emphasize the algorithmic features of gradient estimation algorithms, such as their efficiency and recursiveness. Therefore, this chapter is closely related to Chapter 6 on reinforcement learning. We will introduce performance gradient algorithms from a sample-path-based perspective and leave the algorithmic features, especially those related to the stochastic approximation approach, to Chapter 6 (see Figure 3.1).

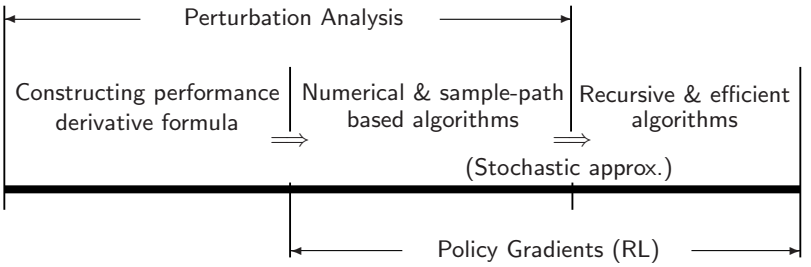

Fig. 3.1. Perturbation Analysis vs. Policy Gradients

3.1 The Potentials

We first study the potentials for ergodic Markov chains (discrete time), and the results can be extended to ergodic Markov processes (continuous time) naturally.

3.1.1 Numerical Methods

With $\pi g = \eta$

The first numerical method depends on the equation for performance potentials ((2.13) and (2.14)):

$$\begin{aligned} g &= (I - P + e\pi)^{-1} f \\ &= \sum_{k=0}^{\infty} [(P - e\pi)^k] f = \Big\{ I + \sum_{k=1}^{\infty} (P^k - e\pi) \Big\} f. \end{aligned}$$

Thus, g can be calculated iteratively by setting:

$$g_0 = f, \quad g_k = f + (P - e\pi) g_{k-1}, \qquad k \geq 0, \tag{3.1}$$

and $g = \lim_{k \to \infty} g_k$. This method requires solving for π first.

With the Realization Factors

Alternatively, we can solve the PRF equation (2.7)

$$\Gamma - P \Gamma P^T = F, \tag{3.2}$$

which does not contain π. Again, its solution is also only up to an additive constant; i.e, if Γ is a solution to (3.2), so is $\Gamma + cee^T$ for any constant c. In addition to (3.2), the PRF matrix $\Gamma = eg^T - ge^T$ also satisfies $\pi\Gamma\pi^T = 0$ or simply $e^T\Gamma e = 0$.

From (3.2), we have

$$\begin{aligned}\Gamma &= P\Gamma P^T + F \\ &= P(P\Gamma P^T + F)P^T + F = P^2\Gamma(P^2)^T + PFP^T + F \\ &= P^k\Gamma(P^k)^T + P^{k-1}F(P^{k-1})^T + \cdots + PFP^T + F.\end{aligned}$$

Since

$$\lim_{k\to\infty} P^k\Gamma(P^k)^T = e\pi\Gamma\pi^T e^T = 0,$$

we have

$$\Gamma = \sum_{k=0}^{\infty} P^k F(P^k)^T,$$

with $P^0 = I$. Therefore, we have the following iterative algorithm

$$\Gamma_0 = F, \quad \Gamma_k = P\Gamma_{k-1}P^T + F, \qquad k \geq 1, \tag{3.3}$$

and $\lim_{k\to\infty}\Gamma_k = \Gamma$. While this algorithm (3.3) does not require solving for π, it has two matrix multiplications in each iteration.

With $g(S) = 0$

Note that in the Poisson equation $(I-P)g+\eta e = f$, the same term η appears in every row. Using this feature, we may develop another numerical algorithm as follows. First, denote the Sth row of P as p_{S*}. Define

$$P_- = P - ep_{S*}.$$

The last row of P_- is zero. Let

$$f_- = [f(1) - f(S), \ldots, f(S-1) - f(S), 0]^T.$$

Subtracting the last row of the Poisson equation from all the rows, and by setting $g(S) = 0$, we get

$$g = P_- g + f_-. \tag{3.4}$$

From this, we can write

$$g = \lim_{L\to\infty}\Big(\sum_{l=0}^{L} P_-^l\Big) f_-. \tag{3.5}$$

Note that because $f_-(S) = 0$ and the last row of P_- is zero, from (3.4) or (3.5) we indeed have $g(S) = 0$. This is consistent with the fact that the potential vector g is unique only up to an additive constant vector, and the g in (3.5) represents one form of the potential vector.

Let $\{1, \lambda_1, \ldots, \lambda_{S-1}\}$ be the set of the eigenvalues of P (see Lemma B.1 in Appendix B). First, we assume that all the eigenvalues are simple. For ergodic chains, we have $|\lambda_i| < 1$ for $i = 1, 2, \ldots, S-1$ [20]. Let $x \neq 0$ be an eigenvector corresponding to one of the eigenvalues, denoted as $\lambda \neq 1$; i.e., $Px = \lambda x$. If $\lambda = 0$, then $Px = 0$ and it is easy to verify that $P_- x = 0$ and $x \neq ce$, with $c \neq 0$ being any constant. That is, $\lambda = 0$ is also an eigenvalue of P_- with eigenvector $x \neq ce$.

Now, we assume that $\lambda \neq 0$. Define $x' = x - \frac{1}{\lambda}(p_{S*}x)e$. Then, we can verify that $x' \neq 0$ and

$$\begin{aligned} P_- x' &= (P - ep_{S*})\Big[x - \frac{1}{\lambda}(p_{S*}x)e\Big] \\ &= \lambda\Big[x - \frac{1}{\lambda}(p_{S*}x)e\Big] = \lambda x', \end{aligned} \tag{3.6}$$

i.e., λ is an eigenvalue of P_- with eigenvector x'. In addition, $P_- e = 0$, i.e., 0 is an eigenvalue of P_-. Therefore, the eigenvalues of $P_- = P - ep_{S*}$ are $\{0, \lambda_1, \ldots, \lambda_{S-1}\}$, with all $|\lambda_i| < 1$, $i = 1, \ldots, S-1$, which are the same as the eigenvalues of $P - e\pi$. One of λ_i, $i = 1, \ldots, S-1$ may be zero (note that we assumed that λ_i, $i = 1, 2, \ldots, S-1$, are different). Therefore, the limit in (3.5) converges at the same rate as (or as fast as) the rate of $\lim_{k\to\infty}(P - e\pi)^k = 0$, or the rate of $\lim_{k\to\infty} P^k = e\pi$.

When there are multiple eigenvalues, we need to examine the multiplicities of the eigenvalues of both P and P_-. First, we assume that $\lambda = 0$ is an eigenvalue of P with $m_0 \geq 0$ multiplicity. We note that for any $x \neq 0$ if $Px = 0$ or $Px = e$ (i.e., $x = e$), then $P_- x = 0$. This means that the space spanned by the eigenvectors of P corresponding to both $\lambda = 0$ and $\lambda = 1$ is a subspace of the space spanned by the eigenvectors of P_- corresponding to $\lambda = 0$.

On the other hand, if $x \neq 0$ and $P_- x = 0$, then either $Px = 0$ or $Px = e$. This can be proved as follows: Because $P_- = P - ep_{S*}$, we have $Px = e(p_{S*}x)$. If $p_{S*}x = 0$, then we have $Px = 0$. If $p_{S*}x \neq 0$, then, without loss of generality, we may assume that $p_{S*}x = 1$. Thus, $Px = e$. This means that the space spanned by the eigenvectors of P_- corresponding to $\lambda = 0$ is a subspace of the space spanned by the eigenvectors of P corresponding to both $\lambda = 0$ and $\lambda = 1$.

Finally, the space spanned by the eigenvectors of P_- corresponding to $\lambda = 0$ is the same as the space spanned by the eigenvectors of P corresponding to both $\lambda = 0$ and $\lambda = 1$; and the multiplicity of $\lambda = 0$ for P_- is $m_0 + 1$.

Let $\lambda \neq 0, 1$ be one of the eigenvalues of P with multiplicity m and x_k, $k = 1, \ldots, m$, be the corresponding linearly independent eigenvectors. As shown in (3.6), $x'_k = x_k - \frac{1}{\lambda}(p_{S*}x_k)e$, $k = 1, \ldots, m$, are eigenvectors of P_-.

We wish to prove that x'_k, $k = 1, \ldots, m$, are linearly independent. Suppose that the opposite is true, i.e., there is a set of real numbers c_k, $k = 1, \ldots, m$, not all of them are zeros, such that $\mathbf{v} = \sum_{k=1}^{m} c_k x'_k = 0$. Set $\mathbf{u} = \sum_{k=1}^{m} c_k x_k$. Because $P_- e = 0$, we have $P_- x'_k = P_- x_k$, $k = 1, 2, \ldots, m$, and

$$P_- \mathbf{u} = P_- \left(\sum_{k=1}^{m} c_k x_k \right) = 0.$$

Thus, $\mathbf{u} \neq 0$ is an eigenvalue of P_- for $\lambda = 0$. Note that $\mathbf{u}$, being a vector spanned by x_k, $k = 1, \ldots, m$, which are eigenvalues of P corresponding to eigenvalue $\lambda \neq 0, 1$, is linearly independent of the eigenvectors of P corresponding to $\lambda = 0$ and 1. Thus, $\mathbf{u}$ adds one to the multiplicity of $\lambda = 0$ for P_-. This implies that the multiplicity of $\lambda = 0$ for P_- is larger than $m_0 + 1$, which is impossible. Therefore, x'_k, $k = 1, \ldots, m$, are linearly independent, and the multiplicity of λ for P_- is the same as that for P.

In summary, we conclude that the eigenvalues of P_- are $\{0, \lambda_1, \ldots, \lambda_{S-1}\}$, with all $|\lambda_i| < 1$, $i = 1, \ldots, S-1$, being the same as those of P. The multiplicity of $\lambda_i \neq 0$, $i = 1, \ldots, S-1$, for P_- are the same as that for P, and the multiplicity of 0 or P_- is $m_0 + 1$.

From (3.5), we have the following iterative algorithm:

$$g_0 = f_-, \quad g_k = f_- + P_- g_{k-1}, \qquad k \geq 1, \tag{3.7}$$

and $g = \lim_{k \to \infty} g_k$.

The above three numerical algorithms have about the same convergence rate (determined by the eigenvalues of P), which is the same as the rate in computing the steady-state probability π using $\lim_{k \to \infty} P^k = e\pi$. The algorithm in (3.7) does not require solving for π, and only one matrix multiplication is needed in each iteration.

In queueing systems, the perturbation realization factors satisfy the set of linear equations (2.107). They can be solved numerically by any standard method for linear equations, and an example is shown in Table 2.9. Further results exploring the special features of these linear equations have not yet been developed in the literature.

3.1.2 Learning Potentials from Sample Paths

The sample-path-based learning algorithms can be derived from (2.16)

$$g(i) = \lim_{L \to \infty} E \left\{ \sum_{l=0}^{L-1} [f(X_l) - \eta] \Big| X_0 = i \right\}, \tag{3.8}$$

and (2.17)

$$\gamma(i,j) = E\left\{ \sum_{l=0}^{L(i|j)-1} [f(X_l) - \eta] \Big| X_0 = j \right\}, \tag{3.9}$$

where $L(i|j) = \min\{l \geq 0 : X_l = i | X_0 = j\}$; or from (2.5) and (2.6),

$$\begin{aligned} &\gamma(i,j) = g(j) - g(i) \\ &= \lim_{L\to\infty} E\left\{ \sum_{l=0}^{L-1} \left[f(\widetilde{X}_l) - f(X_l) \right] \Big| \widetilde{X}_0 = j,\ X_0 = i \right\} \qquad (3.10) \\ &= E\left\{ \sum_{l=0}^{L_{ij}^*-1} \left[f(\widetilde{X}_l) - f(X_l) \right] \Big| \widetilde{X}_0 = j,\ X_0 = i \right\}, \quad i,j = 1,\ldots,S; \qquad (3.11) \end{aligned}$$

at L_{ij}^*, the two sample paths $\widetilde{\boldsymbol{X}}$ and $\boldsymbol{X}$ merge together for the first time.

Algorithms for g

From (3.8), we have the following approximation for $g(i)$,

$$g_L(i) = E\left[\sum_{l=0}^{L-1} f(X_l) \Big| X_0 = i \right] - L\eta, \tag{3.12}$$

with $\lim_{L\to\infty} g_L(i) = g(i)$. The average reward η can be estimated from a sample path by

$$\eta_L = \frac{1}{L} \sum_{l=0}^{L-1} f(X_l), \tag{3.13}$$

with $\eta = \lim_{L\to\infty} \eta_L$, with probability 1. However, because potentials are valid only up to an additive constant, we may ignore the constant $L\eta$ in (3.12) and use its first term as an estimate,

$$g_L(i) = E\left\{ \sum_{l=0}^{L-1} f(X_l) \Big| X_0 = i \right\}. \tag{3.14}$$

With (3.14), the potential g can be estimated on a sample path in a way similar to the estimation of η in (3.13). Let $I_i(x) = 1$ if $x = i$ and $I_i(x) = 0$ if $x \neq i$. Define

$$g_{L,N}(i) = \frac{\sum_{n=0}^{N-L+1} \left\{ I_i(X_n) \left[\sum_{l=0}^{L-1} f(X_{n+l}) \right] \right\}}{\sum_{n=0}^{N-L+1} I_i(X_n)}, \tag{3.15}$$

in which $\sum_{n=0}^{N-L+1} I_i(X_n)$ is the number of visits to state i of the Markov chain in the period of $[0, N-L+1]$. After each such visit, we add up $f(X_n)$ for L transitions, and $g_{L,N}$ is the average of these sums. We have

$$\lim_{N\to\infty} g_{L,N}(i) = g_L(i), \qquad \text{w.p.1.} \tag{3.16}$$

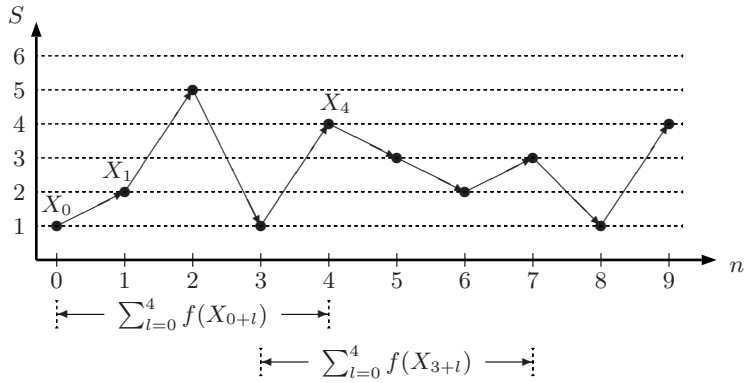

Fig. 3.2. Items in (3.15) Are Not Independent

The proof of (3.16) is not straightforward, since the items $\sum_{l=0}^{L-1} f(X_{n+l})$ for different n may not be independent. For example, given a particular sample path, say $\{1, 2, 5, 1, 4, 3, 2, 3, 1, 6, \ldots\}$ as shown in Figure 3.2, with $L = 5$, the two periods starting from $X_0 = 1$ and $X_3 = 1$ overlap. Both items $\sum_{l=0}^{4} f(X_{0+l}) = f(1) + f(2) + f(5) + f(1) + f(4)$ and $\sum_{l=0}^{4} f(X_{3+l}) = f(1) + f(4) + f(3) + f(2) + f(3)$ contain the same term $f(1) + f(4)$. Therefore, the standard law of large numbers does not apply in this case. The proof of (3.16) is based on a fundamental theorem on ergodicity (see [32]; we state its version on a finite state space $\mathcal{S}$):

The Fundamental Ergodicity Theorem:

Let $\boldsymbol{X} = \{X_n, n \geq 0\}$ be an ergodic Markov chain on state space $\mathcal{S}$; $\phi(x_1, x_2, \ldots)$, $x_i \in \mathcal{S}$, $i = 1, 2 \ldots$, be a function on $\mathcal{S}^\infty$. Then the process $\boldsymbol{Z} = \{Z_n, n \geq 0\}$ with $Z_n = \phi(X_n, X_{n+1}, \ldots)$ is an ergodic Markov chain. In particular, we have

$$\lim_{N\to\infty} \frac{1}{N} \sum_{n=0}^{N-1} \phi(X_n, X_{n+1}, \ldots) = E[\phi(X_n, X_{n+1}, \ldots)], \qquad \text{w.p.1,} \tag{3.17}$$

where "E" denotes the steady-state expectation of the Markov chain $\boldsymbol{Z}$, and the right-hand side of (3.17) does not depend on n.

Since this theorem is very useful in proving the convergence results related to sample-path-based algorithms, we will refer to it as the *Fundamental Ergodicity Theorem*. In our case, we define $Z_n = I_i(X_n)[\sum_{l=0}^{L-1} f(X_{n+l})]$; then, $\{Z_n, n \geq 0\}$ is ergodic. From (3.15), we have

$$g_{L,N}(i) = \frac{\frac{1}{N-L+2}\sum_{n=0}^{N-L+1} Z_n}{\frac{1}{N-L+2}\sum_{n=0}^{N-L+1} I_i(X_n)}.$$

By the fundamental ergodicity theorem, the numerator converges to $E(Z_n) = \pi(i)g_L(i)$ and the denominator converges to $\pi(i)$. Thus, (3.16) holds.

One remaining problem is how to choose L. It is clear that the larger L is, the smaller the bias of (3.15) is. On the other hand, the larger L is, the larger the variance of the estimate is. Therefore, there is a tradeoff in choosing L. We first note that the effect of potentials depends only on their differences, i.e., on the realization factors $\gamma(i,j) = g(j) - g(i)$. Ideally, to estimate $\gamma(i,j)$, the length should be the first passage time from state j to state i (see (3.9)). Therefore, the length of the period, L, should be comparable to the mean of the first passage times from one state to the others. On the other hand, from (3.8), L should be large enough so that $E[f(X_l)]$ is close to η when $l > L$. Because the l-step state transition probability $\mathcal{P}(X_l|X_0)$ converges exponentially fast to the steady-state probability, we may expect that L can be chosen as a small number. The following simulation example provides some empirical evidence.

Example 3.1. We simulated a Markov chain with ten states. The state transition matrix is arbitrarily chosen as

$$P = \begin{bmatrix}
0.20 & 0.00 & 0.05 & 0.10 & 0.15 & 0.15 & 0.05 & 0.05 & 0.05 & 0.20 \\
0.30 & 0.00 & 0.00 & 0.20 & 0.10 & 0.15 & 0.15 & 0.05 & 0.05 & 0.00 \\
0.00 & 0.15 & 0.05 & 0.30 & 0.00 & 0.05 & 0.20 & 0.20 & 0.05 & 0.00 \\
0.05 & 0.10 & 0.25 & 0.00 & 0.30 & 0.00 & 0.05 & 0.20 & 0.05 & 0.000 \\
0.00 & 0.20 & 0.15 & 0.00 & 0.15 & 0.00 & 0.15 & 0.25 & 0.00 & 0.100 \\
0.00 & 0.10 & 0.30 & 0.00 & 0.20 & 0.10 & 0.10 & 0.00 & 0.15 & 0.050 \\
0.10 & 0.10 & 0.10 & 0.10 & 0.10 & 0.10 & 0.10 & 0.10 & 0.10 & 0.100 \\
0.00 & 0.20 & 0.00 & 0.20 & 0.00 & 0.20 & 0.00 & 0.20 & 0.00 & 0.200 \\
0.05 & 0.15 & 0.25 & 0.00 & 0.15 & 0.15 & 0.15 & 0.00 & 0.00 & 0.100 \\
0.15 & 0.05 & 0.00 & 0.20 & 0.15 & 0.10 & 0.20 & 0.10 & 0.05 & 0.000
\end{bmatrix},$$

and the reward function is

$$f = [10, 5, 1, 15, 3, 0, 7, 20, 2, 18]^T.$$

Table 3.1 lists the theoretical and estimated values of the potentials g, in the form of (3.15) and normalized to $\pi g = 0$, estimated with $L = 5$ on a sample path with length $N = 100,000$. The means and standard deviations (SD) are the results of ten simulations. These results indicate that, for a ten-state Markov chain, $L = 5$ yields very accurate estimates for g. □

i	1	2	3	4	5	6	7	8	9	10
Theoretic	1.865	-4.025	-5.121	6.268	-3.259	-13.553	-1.997	14.098	-10.033	9.614
Mean	1.845	-4.056	-5.132	6.243	-3.266	-13.520	-1.893	14.162	-9.902	9.654
SD	0.098	0.088	0.163	0.140	0.140	0.187	0.116	0.110	0.185	0.160

Table 3.1. The Potentials in Example 3.1 with 100,000 Transitions and $L = 5$

Algorithms for $\boldsymbol{\Gamma}$

Next, we derive a sample-path-based algorithm from (3.9). On a sample path of $\boldsymbol{X} = \{X_l, l \geq 0\}$ with $X_0 = i$, for each pair of states j and i, we define two sequences of epochs $\{l_k(j)\}$ and $\{l_k(i)\}$ as follows:

$$\begin{aligned} &l_0(i) = 0, \\ &l_k(j) = \text{the epoch that } \{X_l\} \text{ first visits state } j \text{ after } l_{k-1}(i), \quad k \geq 1, \\ &l_k(i) = \text{the epoch that } \{X_l\} \text{ first visits state } i \text{ after } l_k(j), \quad k \geq 1. \end{aligned} \tag{3.18}$$

Note that $\{l_k(j)\}$ and $\{l_k(i)\}$ are well defined on a sample path; see Figure 3.3.

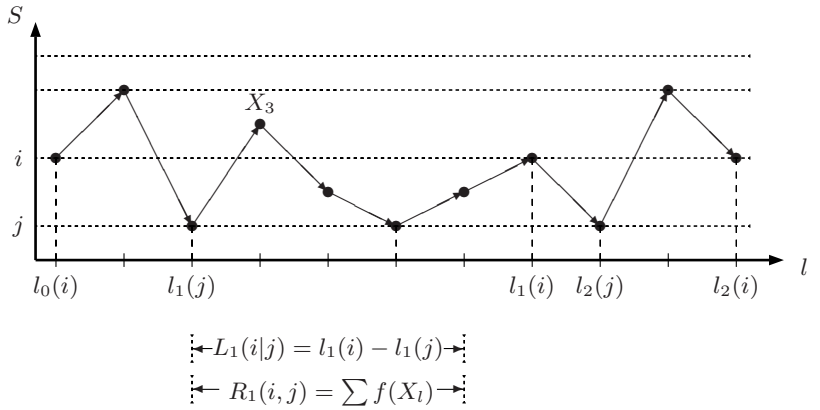

Fig. 3.3. Estimating $\gamma(i, j)$

Now, define $L_k(i|j) = l_k(i) - l_k(j)$ and

$$R_k(i,j) = \sum_{l=l_k(j)}^{l_k(i)-1} f(X_l).$$

The Markov property ensures that the $L_k(i|j)$ and $R_k(i,j)$, $k = 0, 1, \ldots$, are identically and independently distributed (i.i.d), respectively. By the law of large numbers, we have

$$\lim_{N\to\infty} \frac{1}{N} \sum_{k=1}^{N} L_k(i|j) = E[L(i|j)], \qquad \text{w.p.1},$$

and

$$\lim_{N\to\infty} \frac{1}{N} \sum_{k=1}^{N} R_k(i,j) = E\left[\sum_{l=0}^{L(i|j)-1} f(X_l) \Big| X_0 = j \right], \qquad \text{w.p.1}.$$

Therefore,

$$\lim_{N\to\infty} \frac{1}{N} \left\{ \sum_{k=1}^{N} [R_k(i,j) - L_k(i|j)\eta] \right\} = \gamma(i,j), \qquad \text{w.p.1}, \qquad (3.19)$$

where η can be estimated on the sample path using (3.13). Potentials can be obtained by using any row of Γ. We may also use $g = (\pi\Gamma)^T$, which may lead to more accurate estimates because it employs all the rows of Γ.

Example 3.2. We consider the same Markov chain as in Example 3.1. We did ten simulation runs, and each consists of 100,000 state transitions. The theoretical values as well as the means and the standard deviations of the estimated realization factors using (3.19) are listed in Tables 3.2, 3.3, and 3.4, respectively. The estimated matrix Γ is indeed skew-symmetric and standard deviations of most items are of the order 10^{-2}. The statistics of the potentials based on $g = (\pi\Gamma)^T$ are listed in Table 3.5, which shows much smaller standard deviations compared with those in Table 3.1. □

3.1.3 Coupling*

The algorithms based on (3.10) require two sample paths $\boldsymbol{X}$ and $\widetilde{\boldsymbol{X}}$; they are independent and follow the same transition probability matrix P but start from two different states i and j, respectively. However, to estimate $\gamma(i,j)$, the two sample paths do not need to be independent of each other. In fact, it is well known that introducing co-relation between the random samples of two random variables may reduce the variance in estimating the difference of their mean values [257] (also see Problem 3.7). For example, we may use the same sequence of random variables $\{\xi_0, \xi_1, \ldots\}$ to simulate the two sample paths $\boldsymbol{X}$ and $\widetilde{\boldsymbol{X}}$ to obtain estimates for $\gamma(i,j)$ in (3.11). Introducing co-relation between the two sample paths $\boldsymbol{X}$ and $\widetilde{\boldsymbol{X}}$ is called the *coupling approach* in simulation (see, [212]). In the following, we will study this coupling issue in greater detail.

	1	2	3	4	5	6	7	8	9	10
1	0.000	-5.890	-6.987	4.403	-5.124	-15.418	-3.863	12.232	-11.899	7.749
2	5.890	0.000	-1.097	10.293	0.766	-9.528	2.028	18.122	-6.009	13.639
3	6.987	1.097	0.000	11.389	1.863	-8.430	3.124	19.219	-4.912	14.735
4	-4.403	-10.293	-11.389	0.000	-9.527	-19.820	-8.265	7.830	-16.302	3.346
5	5.124	-0.766	-1.863	9.527	0.000	-10.294	1.260	17.356	-6.775	12.873
6	15.418	9.528	8.430	19.820	10.294	0.000	11.556	27.650	3.519	23.167
7	3.863	-2.028	-3.124	8.265	-1.261	-11.556	0.000	16.095	-8.036	11.611
8	-12.232	-18.122	-19.219	-7.830	-17.356	-27.650	-16.095	0.000	-24.130	-4.484
9	11.899	6.009	4.912	16.302	6.775	-3.519	8.036	24.130	0.000	19.647
10	-7.749	-13.639	-14.735	-3.346	-12.873	-23.167	-11.610	4.484	-19.647	0.000

Table 3.2. The Theoretical Values of the Realization Factors in Example 3.2

	1	2	3	4	5	6	7	8	9	10
1	0.000	-5.801	-6.983	4.336	-5.106	-15.377	-3.827	12.286	-11.727	7.756
2	5.800	0.000	-1.012	10.294	0.780	-9.474	2.097	18.204	-5.898	13.682
3	6.983	1.012	0.000	11.381	1.838	-8.416	3.146	19.217	-4.912	14.769
4	-4.336	-10.294	-11.381	0.000	-9.492	-19.690	-8.143	7.876	-16.217	3.442
5	5.105	-0.782	-1.838	9.491	0.000	-10.223	1.390	17.408	-6.582	12.918
6	15.376	9.472	8.414	19.689	10.221	0.000	11.684	27.629	3.647	23.214
7	3.827	-2.098	-3.147	8.142	-1.391	-11.687	0.000	16.014	-7.999	11.629
8	-12.285	-18.204	-19.218	-7.875	-17.409	-27.630	-16.014	0.000	-24.069	-4.491
9	11.726	5.895	4.910	16.214	6.579	-3.653	7.997	24.067	0.000	19.709
10	-7.755	-13.683	-14.768	-3.440	-12.920	-23..216	-11.629	4.493	-19.713	0.000

Table 3.3. The Mean Realization Factors in Example 3.2

Define a composed Markov chain $\widehat{\boldsymbol{X}} := \{(X_l, \widetilde{X}_l), l = 0, 1, \ldots\}$; its state space is

$$\widehat{\mathcal{S}} = \mathcal{S} \times \mathcal{S} = \{(1,1), (1,2), \ldots, (1,S), \\ (2,1), (2,2), \ldots, (2,S), \ldots, (S,1), \ldots, (S,S)\},$$

and its transition probabilities are

$$\widehat{p}[(i',j')|(i,j)] := \mathcal{P}\left(X_{l+1} = i', \widetilde{X}_{l+1} = j' \middle| X_l = i, \widetilde{X}_l = j\right), \\ i, i', j, j' \in \mathcal{S},$$

which equal

	1	2	3	4	5	6	7	8	9	10
1	0.000	0.017	0.060	0.043	0.052	0.017	0.047	0.077	0.144	0.063
2	0.017	0.000	0.020	0.027	0.011	0.025	0.035	0.025	0.109	0.040
3	0.060	0.020	0.000	0.029	0.049	0.054	0.025	0.024	0.065	0.076
4	0.043	0.028	0.029	0.000	0.037	0.028	0.025	0.041	0.116	0.100
5	0.052	0.011	0.050	0.038	0.000	0.021	0.037	0.045	0.041	0.026
6	0.017	0.024	0.054	0.027	0.020	0.000	0.025	0.041	0.037	0.070
7	0.047	0.036	0.025	0.025	0.037	0.025	0.000	0.039	0.059	0.032
8	0.077	0.024	0.024	0.041	0.044	0.041	0.039	0.000	0.128	0.064
9	0.146	0.110	0.065	0.117	0.042	0.039	0.059	0.130	0.000	0.102
10	0.064	0.040	0.076	0.101	0.025	0.068	0.031	0.065	0.097	0.000

Table 3.4. The Standard Deviations of the Realization Factors in Example 3.2

	1	2	3	4	5	6	7	8	9	10
Theoretic	1.865	-4.025	-5.121	6.268	-3.259	-13.553	-1.997	14.098	-10.033	9.614
Mean	1.859	-4.039	-5.092	6.237	-3.273	-13.517	-1.912	14.132	-9.932	9.671
SD	0.0122	0.0074	0.0105	0.0127	0.0111	0.0038	0.0148	0.0146	0.0405	0.0206

Table 3.5. The Potentials Based on the Realization Factors in Example 3.2

$$\widehat{p}[(i',j')|(i,j)] := \mathcal{P}\left(X_{l+1}=i' \,\middle|\, X_l=i, \widetilde{X}_l=j\right) \times \mathcal{P}\left(\widetilde{X}_{l+1}=j' \,\middle|\, X_l=i, \widetilde{X}_l=j, X_{l+1}=i'\right).$$

The transition probability matrix of $\widehat{\boldsymbol{X}}$ is denoted as

$$\widehat{P} = \left[\widehat{p}[(i',j')|(i,j)]\right]_{(i,j),(i,j')\in\widehat{\mathcal{S}}}.$$

To simplify the notation, we denote

$$p_j(i'|i) := \mathcal{P}\left(X_{l+1}=i' \,\middle|\, X_l=i, \widetilde{X}_l=j\right),$$

which is the conditional transition probability distribution of $\boldsymbol{X}$ from state $X=i$ when the Markov chain $\widetilde{\boldsymbol{X}}$ is in state $\widetilde{X}=j$; and

$$\widetilde{p}_{i'|i}(j'|j) := \mathcal{P}\left(\widetilde{X}_{l+1}=j' \,\middle|\, \widetilde{X}_l=j, X_l=i, X_{l+1}=i'\right),$$

which is the conditional transition probability of the Markov chain $\widetilde{\boldsymbol{X}}$ moving from state j to state j', given that the Markov chain $\boldsymbol{X}$ moves from state i to state i'. Thus,

$$\widehat{p}[(i',j')|(i,j)] = p_j(i'|i)\widetilde{p}_{i'|i}(j'|j), \qquad i,i',j,j' \in \mathcal{S}. \tag{3.20}$$

With similar definitions, we have

$$\widehat{p}[(i',j')|(i,j)] = \widetilde{p}_i(j'|j)p_{j'|j}(i'|i), \qquad i,i',j,j' \in \mathcal{S}. \tag{3.21}$$

Summing up both sides of (3.20) and (3.21) over $i' \in \mathcal{S}$, we have

$$\widetilde{p}_i(j'|j) = \sum_{i' \in \mathcal{S}} p_j(i'|i)\widetilde{p}_{i'|i}(j'|j), \qquad j' \in \mathcal{S}.$$

Summing up both sides of (3.20) and (3.21) over $j' \in \mathcal{S}$, we have

$$p_j(i'|i) = \sum_{j' \in \mathcal{S}} \widetilde{p}_i(j'|j)p_{j'|j}(i'|i), \qquad j' \in \mathcal{S}.$$

If $\boldsymbol{X}$ and $\widetilde{\boldsymbol{X}}$ are independent, then $\widetilde{p}_{i'|i}(j'|j) = \widetilde{p}_i(j'|j) = p(j'|j)$ and $p_{j'|j}(i'|i) = p_j(i'|i) = p(i'|i)$, for all $i,i',j,j' \in \mathcal{S}$.

Now, let the reward function of $\widehat{\boldsymbol{X}}$ be $\hat{f}(i,j) = f(j) - f(i)$, the corresponding performance potentials of $\widehat{\boldsymbol{X}}$ be $\widehat{g}(i,j)$, and the steady-state probability distribution of $\widehat{\boldsymbol{X}}$ be $\widehat{\pi}(i,j)$, $i,j \in \mathcal{S}$. We have the Poisson equation for $\widehat{\boldsymbol{X}}$ (assuming $\widehat{P}$ is irreducible):

$$(I - \widehat{P})\widehat{g} + \widehat{\eta}e = \widehat{f}, \tag{3.22}$$

where $\widehat{\eta} = \widehat{\pi}\widehat{f}$ is the average reward.

Equation (3.22) holds for $\widehat{\boldsymbol{X}} = (\boldsymbol{X}, \widetilde{\boldsymbol{X}})$. In our case, both $\boldsymbol{X}$ and $\widetilde{\boldsymbol{X}}$ have the same transition probability matrix P. Thus, their steady-state probabilities are equal; i.e., $\pi(i) = \widetilde{\pi}(i)$, $i \in \mathcal{S}$. Thus, we have $\widehat{\eta} = \widehat{\pi}\widehat{f} = \sum_{i,j \in \mathcal{S}} \widehat{\pi}(i,j)\widehat{f}(i,j) = 0$, and (3.22) becomes

$$(I - \widehat{P})\widehat{g} = \widehat{f}.$$

In addition, although the transitions of $\boldsymbol{X}$ and $\widetilde{\boldsymbol{X}}$ are coupled, the transition of each of $\boldsymbol{X}$ and $\widetilde{\boldsymbol{X}}$ at any time must follow the transition probability matrix P. Precisely, we may require that

$$\widetilde{p}_i(j'|j) = p(j'|j), \qquad i \in \mathcal{S}, \tag{3.23}$$

and that

$$p_j(i'|i) = p(i'|i), \qquad j \in \mathcal{S}. \tag{3.24}$$

Under these conditions the coupling is reflected by the conditional transition probabilities $p_{j'|j}(i'|i)$ and $\widetilde{p}_{i'|i}(j'|j)$. Next, we show that, under these conditions, $\widehat{g}(i,j) = g(j) - g(i)$, $i,j \in \mathcal{S}$ is indeed a solution to (3.22).

To facilitate the matrix manipulation, we need to introduce some notation. Let $A = [a(i,j)]$ be an $m \times n$ matrix and $B = [b(i',j')]$ be an $m' \times n'$ matrix.

The *Kronecker product* of A and B is defined as the $(mm') \times (nn')$ matrix denoted as

$$A \otimes B = \begin{bmatrix} a(1,1)B & \dots & a(1,n)B \\ \vdots & \ddots & \vdots \\ a(m,1)B & \dots & a(m,n)B \end{bmatrix}.$$

For clarity, we use e_m to denote an m-dimensional column vector with all components being one.

With this notation, we can verify that

$$\hat{f} = (e_S \otimes f) - (f \otimes e_S).$$

Conditions (3.23) and (3.24) are equivalent to

$$\widehat{P}(I \otimes e_S) = P \otimes e_S, \tag{3.25}$$

and

$$\widehat{P}(e_S \otimes I) = e_S \otimes P. \tag{3.26}$$

We can easily derive that, for any matrix A and vector g, if Ag is well defined, then $(A \otimes e)g = (Ag) \otimes e$ and $(e \otimes A)g = e \otimes (Ag)$, for an e with any dimension.

Finally, from (3.25) and (3.26), we have

$$\begin{aligned} & (I - \widehat{P})(e_S \otimes I - I \otimes e_S)g \\ &= (e_S \otimes I - I \otimes e_S)g - [e_S \otimes (Pg) - (Pg) \otimes e_S] \\ &= e_S \otimes [(I - P)g] - [(I - P)g] \otimes e_S \\ &= e_S \otimes [(I - P + e_S\pi)g] - [(I - P + e_S\pi)g] \otimes e_S \\ &= e_S \otimes f - f \otimes e_S \\ &= \widehat{f}. \end{aligned}$$

Thus, under conditions (3.23) and (3.24),

$$\widehat{g} = e_S \otimes g - g \otimes e_S$$

is indeed one of the solutions to (3.22). That is, $\widehat{g}(i,j) = g(j) - g(i) = \gamma(i,j)$, $i, j \in \mathcal{S}$, are the realization factors of $\boldsymbol{X}$. We have

$$e_{S^2}^T \widehat{g} = 0, \quad \text{and} \quad e_{S^2}^T \widehat{f} = 0.$$

Equation (3.22) is the *perturbation realization factor (PRF) equation with coupled sample paths*.

Now, we discuss the numerical method for solving (3.22). Let ν be any S^2 dimensional row vector such that $\nu e_{S^2} = 1$, and $\nu\widehat{g} = 0$. For example, we can take $\nu = \frac{1}{S^2} e_{S^2}^T$. We can write the PRF equation (3.22) as

$$(I - \widehat{P} + e_{S^2}\nu)\widehat{g} = \hat{f}.$$

We can prove (see Problem 3.2) that the eigenvalues of $\widehat{P} - e_{S^2}\nu$ are all in the unit circle. Thus, we have the following expansion:

$$\begin{aligned}\widehat{g} &= (I - \widehat{P} + e_{S^2}\nu)^{-1}\widehat{f} \\ &= \sum_{l=0}^{\infty}(\widehat{P} - e_{S^2}\nu)^l \widehat{f}. \end{aligned} \tag{3.27}$$

Let λ be one of the eigenvalues of P and x be its corresponding eigenvector. Define $\widehat{x} = x \otimes e_S$. It is easy to verify that λ is the eigenvalue of $\widehat{P}$ with eigenvector $\widehat{x}$. Therefore, all the eigenvalues of P are the eigenvalues of $\widehat{P}$ (which may have other eigenvalues). Thus, the convergence rate of (3.27) cannot be better than (3.1) or (3.3). Therefore, the coupling approach cannot improve the convergence rate of the numerical algorithms for calculating Γ ($\gamma(i,j) = \widehat{g}(i,j)$).

The coupling method is generally used in simulation to reduce the variance of the estimates for the difference of the mean of two different random variables. Relative references include [31, 33, 91, 92, 115, 127, 177, 179, 212, 213]. Applying this approach to estimate $\gamma(i,j) = g(j) - g(i)$ with two coupled sample paths still requires further research and we will not discuss the details in this book. Problems 3.9 and 3.10 provide a brief introduction to this variance-reduction simulation approach.

3.2 Performance Derivatives

3.2.1 Estimating through Potentials

The performance potentials obtained by numerical methods or by learning from sample paths can be used to calculate the performance derivatives using the performance derivative formula (2.23):

$$\frac{d\eta_\delta}{d\delta} = \pi(\Delta P)g. \tag{3.28}$$

We first give a few numerical examples.

Example 3.3. We consider a Markov chain with the same transition probability matrix P and reward function f as those in Examples 3.1 and 3.2. To study the derivatives of the average reward, we arbitrarily choose the direction of P as

$$\Delta P = \begin{bmatrix} -0.010 & 0.000 & 0.005 & 0.005 & -0.010 & 0.010 & 0.010 & 0.005 & 0.005 & -0.020 \\ -0.010 & 0.000 & 0.000 & 0.015 & 0.005 & 0.005 & -0.005 & -0.005 & -0.005 & 0.000 \\ 0.000 & 0.010 & 0.010 & 0.010 & 0.000 & -0.010 & 0.000 & -0.010 & -0.010 & 0.000 \\ 0.005 & -0.020 & 0.005 & 0.000 & 0.005 & 0.000 & 0.010 & -0.010 & 0.005 & 0.000 \\ 0.000 & 0.010 & -0.010 & 0.000 & 0.010 & 0.000 & -0.010 & 0.010 & 0.000 & -0.010 \\ 0.000 & 0.010 & -0.010 & 0.000 & -0.020 & 0.005 & 0.005 & 0.000 & 0.005 & 0.005 \\ 0.010 & -0.010 & 0.010 & -0.010 & 0.010 & -0.010 & 0.010 & -0.010 & 0.010 & -0.010 \\ 0.000 & 0.010 & 0.000 & -0.010 & 0.000 & 0.010 & 0.000 & -0.005 & 0.000 & -0.005 \\ 0.010 & -0.010 & -0.020 & 0.000 & 0.010 & 0.010 & 0.010 & 0.000 & 0.000 & -0.010 \\ 0.010 & -0.010 & 0.000 & 0.010 & -0.010 & -0.010 & 0.010 & -0.010 & 0.010 & 0.000 \end{bmatrix}.$$

We use the sample-path-based estimates of potentials in the form of (3.15) to compute the derivatives. To study the effect of L, we choose $L = 1$, 2, 3, 5, 10, 15, 20. For each value of L, we do two sets of simulation, with each set having ten runs. Each simulation run contains 100,000 state transitions in the first set and 1,000,000 transitions in the second set. The theoretical value of the derivative is -0.1176. The means and standard deviations of the derivatives calculated by (3.28) using the sample-path-based potential estimates in these two sets of simulations are listed in Tables 3.6 and 3.7.

L	1	2	3	5	10	15	20	Theoretic
Mean	-0.0979	-0.1224	-0.1162	-0.1172	-0.1180	-0.1183	-0.1176	-0.1176
SD	0.00045	0.00059	0.00070	0.00151	0.00186	0.00261	0.00216	-

Table 3.6. The Performance Derivatives in Example 3.3 with 100,000 Transitions

L	1	2	3	5	10	15	20	Theoretic
Mean	-0.0989	-0.1229	-0.1167	-0.1176	-0.1178	-0.1176	-0.1174	-0.1176
SD	0.00009	0.00015	0.00016	0.00025	0.00026	0.00047	0.00059	-

Table 3.7. The Performance Derivatives in Example 3.3 with 1,000,000 Transitions

These tables show that the estimate is quite accurate even when L is as small as 2 or 3. The standard deviation is acceptable even if L is 20. Thus, the results are not so sensitive to the value of L. It is interesting to note that even if we choose $L = 1$ in this case, the error is only about 17%. $L = 1$ means using the reward function to approximate the potentials, i.e., assuming that $g \approx f$. This corresponds to the "myopic" view in optimization: When the system jumps to state i, we just use the one step reward $f(i)$ to represent its effect on the long-run performance. □

3.2.2 Learning Directly

One disadvantage of the approach in Section 3.2.1 is that it requires us to estimate the potentials for all the states. This is sometimes difficult for a number of reasons: The number of states may be too large; some states may be visited very rarely; and for systems with special structures (e.g. queueing networks), it may not be convenient even to list out all the states. In this subsection, we show that the performance derivatives can be estimated directly from sample paths without estimating each individual potential.

An analogue is the estimation of the performance measure itself. There are two ways to do the estimation: We may estimate all $\pi(i)$ first and then use $\eta = \pi f$ to calculate the performance, or we may estimate η directly by

$$\eta = \lim_{L\to\infty} \frac{1}{L} \sum_{l=0}^{L-1} f(X_l), \qquad \text{w.p.1.} \tag{3.29}$$

This direct estimation balances the accuracy of $\pi(i)$ and the frequency of the visits to i: If i is not visited often, then $\pi(i)$ may not be accurately estimated; meanwhile, its effect on η is also small. We wish to develop equations similar to (3.29) for the derivatives of average rewards.

A Basic Formula and a General Algorithm

We first present a basic formula for the direct estimation of the derivatives of average rewards. This formula is the foundation of the sample-path-based algorithms. With this formula, a general algorithm for derivatives can be developed; many other algorithms can be viewed as special cases of this general algorithm [61].

Consider a stationary Markov chain $\boldsymbol{X} = (X_0, X_1, \ldots)$. (This implies that the initial probability distribution is the steady-state distribution π.) Let E denote the expectation on the probability space generated by $\boldsymbol{X}$. Because it is impossible for a sample path with transition matrix P to contain information about $\Delta P = P' - P$, we need to use a standard technique in simulation called *importance sampling*. First, we make a standard assumption in importance sampling: For any $i, j \in \mathcal{S}$, if $\Delta p(j|i) \neq 0$, then $p(j|i) \neq 0$. This assumption allows us to analyze the effect of $\Delta p(j|i)$ based on the information observed when the system moves from state i to state j on $\boldsymbol{X}$. If the assumption does not hold, we may have $p'(j|i) > 0$ while $p(j|i) = 0$ for some $i, j \in \mathcal{S}$. In this case, a sample path of $\boldsymbol{X}$ does not contain any transition from i to j, and we may need to observe two or more transitions (see Problem 3.11).

First, we have (2.23)

$$\begin{aligned} \frac{d\eta_\delta}{d\delta} &= \pi \Delta P g = \sum_{i\in\mathcal{S}} \sum_{j\in\mathcal{S}} [\pi(i)\Delta p(j|i) g(j)] \\ &= \sum_{i\in\mathcal{S}} \sum_{j\in\mathcal{S}} \left\{ \pi(i) p(j|i) \left[\frac{\Delta p(j|i)}{p(j|i)} g(j) \right] \right\}. \end{aligned}$$

For a stationary Markov chain $\boldsymbol{X} = \{X_l, l = 0, 1, \ldots\}$, this is

$$\frac{d\eta_\delta}{d\delta} = E\left\{ \frac{\Delta p(X_{l+1}|X_l)}{p(X_{l+1}|X_l)} g(X_{l+1}) \right\}, \tag{3.30}$$

which does not depend on l. Next, let $\hat{g}(X_{l+1}, X_{l+2}, \ldots)$ be an unbiased estimate of $g(X_{l+1})$; i.e., let

$$g(i) = E\left\{ \hat{g}(X_{l+1}, X_{l+2}, \ldots) | X_{l+1} = i \right\}, \qquad i \in \mathcal{S}. \tag{3.31}$$

With (3.31), we have

$$
\begin{aligned}
& E\left\{\frac{\Delta p(X_{l+1}|X_l)}{p(X_{l+1}|X_l)}\hat{g}(X_{l+1},X_{l+2},\ldots)\right\} \\
&= E\left\{E\left[\frac{\Delta p(X_{l+1}|X_l)}{p(X_{l+1}|X_l)}\hat{g}(X_{l+1},X_{l+2},\ldots)\Big|X_l,X_{l+1}\right]\right\} \\
&= E\left\{\frac{\Delta p(X_{l+1}|X_l)}{p(X_{l+1}|X_l)}E\left[\hat{g}(X_{l+1},X_{l+2},\ldots)\Big|X_l,X_{l+1}\right]\right\} \\
&= E\left\{\frac{\Delta p(X_{l+1}|X_l)}{p(X_{l+1}|X_l)}g(X_{l+1})\right\}.
\end{aligned}
$$

Therefore, we have the following *basic formula*:

$$
\frac{d\eta_\delta}{d\delta} = E\left\{\frac{\Delta p(X_{l+1}|X_l)}{p(X_{l+1}|X_l)}\hat{g}(X_{l+1},X_{l+2},\ldots)\right\}. \tag{3.32}
$$

With this formula, we can develop a general algorithm for estimating derivatives. In fact, for an ergodic Markov chain $\boldsymbol{X} = \{X_0, X_1, \ldots\}$, we have

$$
\frac{d\eta_\delta}{d\delta} = \lim_{N\to\infty}\frac{1}{N}\sum_{n=0}^{N-1}\left\{\left[\frac{\Delta p(X_{n+1}|X_n)}{p(X_{n+1}|X_n)}\right]\hat{g}(X_{n+1},X_{n+2},\ldots)\right\}, \quad \text{w.p.1}, \tag{3.33}
$$

where $\hat{g}(X_{n+1}, X_{n+2}, \ldots)$ is any function satisfying (3.31).

The proof of (3.33) follows directly from the fundamental ergodicity theorem (3.17) by simply defining

$$
\phi(X_n, X_{n+1}, \ldots) = \frac{\Delta p(X_{n+1}|X_n)}{p(X_{n+1}|X_n)}\hat{g}(X_{n+1},X_{n+2},\ldots).
$$

Specific Algorithms

With different estimates or approximations of the potentials, (3.33) leads to a few specific approximate algorithms for the derivatives of average rewards.

Algorithm 3.1. *(Approximation by truncation)*

With (3.14), we have

$$
g(i) \approx g_L(i) = E\left\{\sum_{l=0}^{L-1} f(X_l)\Big|X_0 = i\right\}.
$$

Therefore, from (3.31), we may choose

$$\hat{g}(X_{n+1}, X_{n+2}, \cdots) \approx \sum_{l=0}^{L-1} f(X_{n+l+1}).$$

Using this $\hat{g}$ in (3.32) and (3.33), we get

$$\begin{aligned}\frac{d\eta_\delta}{d\delta} &\approx E\left\{\frac{\Delta p(X_{n+1}|X_n)}{p(X_{n+1}|X_n)}\Big[\sum_{l=0}^{L-1} f(X_{n+l+1})\Big]\right\} \\ &= \lim_{N\to\infty}\frac{1}{N}\left\{\sum_{n=0}^{N-1}\Big[\frac{\Delta p(X_{n+1}|X_n)}{p(X_{n+1}|X_n)}\Big]\Big[\sum_{l=0}^{L-1} f(X_{n+l+1})\Big]\right\}, \qquad \text{w.p.1.}\end{aligned} \tag{3.34}$$

This is equivalent to

$$\frac{d\eta_\delta}{d\delta} \approx \lim_{N\to\infty}\frac{1}{N}\sum_{n=0}^{N-1}\left\{f(X_{n+L})\sum_{l=0}^{L-1}\left[\frac{\Delta p(X_{n+l+1}|X_{n+l})}{p(X_{n+l+1}|X_{n+l})}\right]\right\}, \qquad \text{w.p.1.} \tag{3.35}$$

This algorithm and similar ones for Markov processes and queueing networks are presented in [69].

Example 3.4. We repeat the simulation for the same Markov chain as in Example 3.3 and apply (3.35) to estimate the derivative of the average reward. We perform ten simulation runs for each value of L and the results are listed in Table 3.8. The table shows that for $L = 2$ to 15, (3.35) yields very accurate estimates. When L increases further from 20, the estimate becomes inaccurate because the variance becomes larger. □

By the ergodicity of the Markov chain, (3.35) can be written as

$$\frac{d\eta_\delta}{d\delta} \approx E\left\{f(X_{n+L})\sum_{l=0}^{L-1}\left[\frac{\Delta p(X_{n+l+1}|X_{n+l})}{p(X_{n+l+1}|X_{n+l})}\right]\right\}, \tag{3.36}$$

where "E" denotes the steady-state expectation. Define

$$\rho_L(i) = E\left\{\sum_{l=0}^{L-1}\left[\frac{\Delta p(X_{l+1}|X_l)}{p(X_{l+1}|X_l)}\right]\Big| X_L = i\right\}.$$

Then, (3.36) becomes

L	1	2	3	5	10	15
Mean	-0.0973	-0.1221	-0.1157	-0.1163	-0.1151	-0.1137
SD	0.0033	0.0067	0.0104	0.0162	0.0305	0.00443

L	25	50	75	100	200	Theoretic
Mean	-0.1098	-0.1035	-0.0933	-0.0797	-0.0434	-0.1176
SD	0.0760	0.1522	0.2300	0.3086	0.6351	-

Table 3.8. The Performance Derivatives in Example 3.4 with 100,000 Transitions

$$\frac{d\eta_\delta}{d\delta} \approx \sum_{i\in\mathcal{S}} \pi(i) f(i) \rho_L(i).$$

Define

$$\rho(i) = \lim_{L\to\infty} \rho_L(i), \qquad i \in \mathcal{S}. \tag{3.37}$$

Then, by the above derivation, we have

$$\frac{d\eta_\delta}{d\delta} = \sum_{i\in\mathcal{S}} \pi(i) f(i) \rho(i). \tag{3.38}$$

Equation (3.38) and the convergence of (3.37) can be rigorously proved; see Problem 3.15.

Algorithm 3.2. *(Approximation by discounting)*

Because $\lim_{\beta\uparrow 1} g_\beta = g$, potential g can be approximated by the β-potential g_β in (2.45):

$$g_\beta(i) = E\left\{ \sum_{l=0}^{\infty} \beta^l [f(X_l) - \eta] \Big| X_0 = i \right\},$$

with $0 < \beta < 1$ being a discount factor. Ignoring the constant term, we have the approximation of the potential as follows:

$$g_\beta(i) = E\left\{ \sum_{l=0}^{\infty} \beta^l f(X_l) \Big| X_0 = i \right\}.$$

Therefore, we can choose

$$\hat{g}(X_{n+1}, X_{n+2}, \ldots) \approx \sum_{l=0}^{\infty} \beta^l f(X_{n+l+1}).$$

Using this as the $\hat{g}$ in (3.33), we get

$$\frac{d\eta_\delta}{d\delta} \approx \lim_{N\to\infty} \frac{1}{N} \left\{ \sum_{n=0}^{N-1} \left[\frac{\Delta p(X_{n+1}|X_n)}{p(X_{n+1}|X_n)} \right] \Big[\sum_{l=0}^{\infty} \beta^l f(X_{n+l+1}) \Big] \right\}, \qquad \text{w.p.1}$$

$$= E \left\{ \frac{\Delta p(X_{n+1}|X_n)}{p(X_{n+1}|X_n)} \Big[\sum_{l=0}^{\infty} \beta^l f(X_{n+l+1}) \Big] \right\}. \tag{3.39}$$

The right-hand side of (3.39) equals the sum of the two terms:

$$\lim_{N\to\infty} \frac{1}{N} \sum_{n=0}^{N-1} \left\{ \frac{\Delta p(X_{n+1}|X_n)}{p(X_{n+1}|X_n)} \Big[\sum_{l=0}^{N-n-1} \beta^l f(X_{n+l+1}) + \sum_{l=N-n}^{\infty} \beta^l f(X_{n+l+1}) \Big] \right\}. \tag{3.40}$$

For the second term, we have

$$\left| \frac{1}{N} \sum_{n=0}^{N-1} \left\{ \frac{\Delta p(X_{n+1}|X_n)}{p(X_{n+1}|X_n)} \Big[\sum_{l=N-n}^{\infty} \beta^l f(X_{n+l+1}) \Big] \right\} \right|$$

$$\leq \max_{i,j\in\mathcal{S}} \left| \frac{\Delta p(j|i)}{p(j|i)} \right| \max_{i\in\mathcal{S}} |f(i)| \left\{ \frac{1}{N} \sum_{n=0}^{N-1} \sum_{l=N-n}^{\infty} \beta^l \right\}$$

$$\to 0, \qquad \text{as } N \to \infty.$$

Therefore, the second term in (3.40) is zero, and (3.39) becomes

$$\frac{d\eta_\delta}{d\delta} \approx \lim_{N\to\infty} \frac{1}{N} \sum_{n=0}^{N-1} \left\{ \frac{\Delta p(X_{n+1}|X_n)}{p(X_{n+1}|X_n)} \Big[\sum_{l=0}^{N-n-1} \beta^l f(X_{n+l+1}) \Big] \right\}, \qquad \text{w.p.1.}$$

We exchange the order of the above two finite sums and obtain

$$\frac{d\eta_\delta}{d\delta} \approx \lim_{N\to\infty} \frac{1}{N} \sum_{n=1}^{N} \left\{ f(X_n) \sum_{l=0}^{n-1} \left[\beta^{n-l-1} \frac{\Delta p(X_{l+1}|X_l)}{p(X_{l+1}|X_l)} \right] \right\}, \qquad \text{w.p.1.} \tag{3.41}$$

This is the policy-gradient algorithm developed in [17, 18].

We can calculate $z_n := \sum_{l=0}^{n-1} \left[\beta^{n-l-1} \frac{\Delta p(X_{l+1}|X_l)}{p(X_{l+1}|X_l)} \right]$ recursively: set $z_0 = 0$ and

$$z_{k+1} = \beta z_k + \frac{\Delta p(X_{k+1}|X_k)}{p(X_{k+1}|X_k)}, \qquad k \geq 0.$$

On the other hand, to calculate $\sum_{l=0}^{L-1} \left[\frac{\Delta p(X_{n+l+1}|X_{n+1})}{p(X_{n+l+1}|X_{n+1})} \right]$ in Algorithm 3.1, we have to store L values.

Finally, the discount factor approximation can also be used to reduce the variance in estimating the performance gradients [198].

Example 3.5. We repeat the simulation for the same Markov system as in Examples 3.3 and 3.4 and apply (3.41) to estimate the performance derivative. We perform ten simulation runs for each value of the discount factor β, and the means and standard deviations of the estimates are listed in Table 3.9. The table shows that for $\beta = 0.8$ to 0.9, the algorithm in (3.41) yields very accurate estimates. Of course, when β increases, g_β increases and goes closer to g. However, when β increases, the variance of the estimate also increases. This explains why the estimation error becomes larger for $\beta > 0.9$. Thus, when we choose the value of β, we need to balance both bias and variance. The table shows that $\beta = 0.8$ to 0.9 are the best choices. □

β	0.80	0.85	0.90	0.95	0.97	Theoretic
Mean	-0.113	-0.114	-0.114	-0.111	-0.125	-0.1176
SD	0.016	0.021	0.031	0.061	0.065	-

Table 3.9. The Performance Derivatives in Example 3.5 with 100,000 Transitions

Algorithm 3.3. *(Based on perturbation realization factors)*

It is sometimes easier and more accurate to estimate the potentials via perturbation realization factors $\gamma(i,j) = g(j) - g(i)$, $i, j, \in \mathcal{S}$. This is based on (2.17)

$$\gamma(i,j) = E\left\{ \sum_{l=0}^{L(i|j)-1} [f(X_l) - \eta] \Big| X_0 = j \right\}.$$

To develop a direct algorithm for derivatives of average rewards, we first use the above equation to obtain $\hat{g}$. To this end, we choose any regenerative state i^* as a reference point and set $g(i^*) = 0$. Then, for any state $i \in \mathcal{S}$, we have

$$g(i) = g(i) - g(i^*) = \gamma(i^*, i).$$

For convenience, we set $X_0 = i^*$ and define $u_0 = 0$, and we let $u_{k+1} = \min\{n : n > u_k, X_n = i^*\}$ be the sequence of regenerative points. For any time instant $n \geq 0$, we define an integer $m(n)$ such that $u_{m(n)} \leq n < u_{m(n)+1}$. This implies that $u_{m(n)} = n$ when $i = i^*$. From (2.17), we have

$$g(i) = \gamma(i^*, i) = E\left\{ \sum_{l=n}^{u_{m(n)+1}-1} \Big[f(X_l) - \eta \Big] \Big| X_n = i \right\}.$$

Choosing $\hat{g}(X_{n+1}, \ldots) = \sum_{l=n+1}^{u_{m(n+1)+1}-1} [f(X_l) - \eta]$ in (3.32) and (3.33), we have

$$\frac{d\eta_\delta}{d\delta} = E\left\{\left[\frac{\Delta p(X_{n+1}|X_n)}{p(X_{n+1}|X_n)}\right]\sum_{l=n+1}^{u_{m(n+1)+1}-1}[f(X_l)-\eta]\right\}$$

$$= \lim_{N\to\infty}\frac{1}{N}\sum_{n=0}^{N-1}\left\{\left[\frac{\Delta p(X_{n+1}|X_n)}{p(X_{n+1}|X_n)}\right]\sum_{l=n+1}^{u_{m(n+1)+1}-1}[f(X_l)-\eta]\right\}, \quad \text{w.p.1(3.42)}$$

$$= \lim_{N\to\infty}\frac{1}{N}\sum_{n=1}^{N}\left\{[f(X_n)-\eta]\sum_{l=u_{m(n)}}^{n}\frac{\Delta p(X_l|X_{l-1})}{p(X_l|X_{l-1})}\right\}, \quad \text{w.p.1, (3.43)}$$

where $u_{m(n+1)+1}$ is the first time after X_{n+1} that the Markov chain reaches state i^*.

Next, define

$$\hat{w}_{n+1} = \sum_{l=n+1}^{u_{m(n+1)+1}-1}[f(X_l)-\eta].$$

By the regenerative property, from (3.42) we have

$$\frac{d\eta_\delta}{d\delta} = \frac{E\left\{\sum_{k=u_m}^{u_{m+1}-1}\left(\frac{\Delta p(X_{k+1}|X_k)}{p(X_{k+1}|X_k)}\hat{w}_{k+1}\right)\right\}}{E[u_{m+1}-u_m]}$$

$$= E\left(\frac{\Delta p(X_{k+1}|X_k)}{p(X_{k+1}|X_k)}\hat{w}_{k+1}\right). \quad (3.44)$$

Define

$$\hat{r}_n = \sum_{l=u_{m(n)}}^{n}\frac{\Delta p(X_l|X_{l-1})}{p(X_l|X_{l-1})}.$$

Therefore, (3.43) takes the following form

$$\frac{d\eta_\delta}{d\delta} = \frac{E\left\{\sum_{k=u_m}^{u_{m+1}-1}[f(X_k)-\eta]\hat{r}_k\right\}}{E[u_{m+1}-u_m]}$$

$$= E\left\{[f(X_k)-\eta]\hat{r}_k\right\}. \quad (3.45)$$

The optimization scheme proposed in [197] is essentially a result of combining the above algorithms with stochastic approximation techniques. See Section 6.3.1 for additional discussion.

Example 3.6. We repeat the simulation for the same Markov system as in Examples 3.3, 3.4, and 3.5. We perform ten simulation runs and apply (3.43) to estimate the performance derivatives. The mean is -0.1191 and the standard deviation is 0.0075. □

Algorithm 3.4. *(Parameterized policy spaces)*

Now, we consider a parameterized space of transition probability matrices denoted as $P_\theta = [p_\theta(j|i)]$, $i, j \in \mathcal{S}$, where θ is a continuous parameter and $\frac{d}{d\theta}\{p_\theta(j|i)\}$ exists for all $i, j \in \mathcal{S}$. We assume that the Markov chains under all transition probability matrices are ergodic. The corresponding steady-state probabilities and average rewards are denoted as π_θ and $\eta_\theta = \pi_\theta f$. For simplicity, we assume that the reward function f is the same for all P_θ. (The extension to f_θ depending on θ is straightforward.)

Algorithms for the derivatives of average rewards can be developed by replacing $\Delta p(j|i)$ in (3.32) and (3.33) with $\frac{d}{d\theta}\{p_\theta(j|i)\}$. For example, the basic formula (3.32) becomes

$$\boxed{\frac{d\eta_\theta}{d\theta} = E\left\{\frac{\frac{d}{d\theta}p_\theta(X_{n+1}|X_n)}{p(X_{n+1}|X_n)}\hat{g}(X_{n+1}, X_{n+2}, \ldots)\right\},}$$

where $\hat{g}(X_{n+1}, X_{n+2}, \ldots)$ is an unbiased estimate of $g(i)$, given $X_{n+1} = i$. The specific algorithms (3.34) and (3.39) become

$$\frac{d\eta_\theta}{d\theta} = E\left\{\frac{\frac{d}{d\theta}p_\theta(X_{n+1}|X_n)}{p(X_{n+1}|X_n)}\Big[\sum_{l=0}^{L-1} f(X_{n+l+1})\Big]\right\}, \tag{3.46}$$

$$\frac{d\eta_\theta}{d\theta} = E\left\{\frac{\frac{d}{d\theta}p_\theta(X_{n+1}|X_n)}{p(X_{n+1}|X_n)}\Big[\sum_{l=0}^{\infty} \beta^l f(X_{n+l+1})\Big]\right\}. \tag{3.47}$$

From (3.44), we have

$$\begin{aligned}\frac{d\eta_\theta}{d\theta} &= E\left\{\frac{\frac{d}{d\theta}p_\theta(X_{n+1}|X_n)}{p(X_{n+1}|X_n)}\Big[\sum_{l=n+1}^{u_{m(n+1)+1}-1}[f(X_l)-\eta]\Big]\right\} \\ &= \frac{E\left\{\sum_{n=u_m}^{u_{m+1}-1}\left[\frac{\frac{d}{d\theta}p(X_{n+1}|X_n)}{p(X_{n+1}|X_n)}\Big(\sum_{l=n+1}^{u_{m(n+1)+1}-1}[f(X_l)-\eta]\Big)\right]\right\}}{E[u_{m+1}-u_m]}.\end{aligned} \tag{3.48}$$

From(3.45), we have

$$\begin{aligned}\frac{d\eta_\theta}{d\theta} &= E\left\{[f(X_n)-\eta]\sum_{l=u_{m(n)}}^{n}\frac{\frac{d}{d\theta}p_\theta(X_l|X_{l-1})}{p(X_l|X_{l-1})}\right\} \\ &= \frac{E\left\{\sum_{n=u_m}^{u_{m+1}-1}\left[[f(X_n)-\eta]\sum_{l=u_{m(n)}}^{n}\frac{\frac{d}{d\theta}p_\theta(X_l|X_{l-1})}{p(X_l|X_{l-1})}\right]\right\}}{E[u_{m+1}-u_m]}.\end{aligned} \tag{3.49}$$

Other equations similar to (3.35) and (3.41) can be developed.

Example 3.7. The above estimation algorithms are applied to the partially-observable Markov decision processes (POMDPs) in [17, 18]. (This example can be better understood after reading the materials in Chapter 4 about the Markov decision processes.)

In this example, we use the following simple parameterized model. In addition to the state space $\mathcal{S}$, there is a finite action space denoted as $\mathcal{A}$ and a finite observation space denoted as $\mathcal{Y}$. Each $\alpha \in \mathcal{A}$ determines a transition probability matrix $P^\alpha = [p^\alpha(j|i)]$. When the Markov chain is in state $i \in \mathcal{S}$, an observation $y \in \mathcal{Y}$ is obtained according to a probability distribution $\nu_i(y)$. For any observation y, we may choose a randomized policy $\mu_y(\alpha)$, which is a probability distribution over the action space $\mathcal{A}$. It is assumed that the distribution depends on a parameter θ and therefore is denoted as $\mu_y(\theta, \alpha)$. When $y \in \mathcal{Y}$ is observed, with policy $\mu_y(\theta, \alpha)$, we take action $\alpha \in \mathcal{A}$ with probability $\mu_y(\theta, \alpha)$. Furthermore, we assume that P^α does not depend on θ.

Given an observation distribution $\nu_i(y)$ and a randomized policy $\mu_y(\theta, \alpha)$, the corresponding transition probability is

$$p_\theta(j|i) = \sum_{\alpha,y} \{\nu_i(y)\mu_y(\theta, \alpha)p^\alpha(j|i)\}.$$

Therefore,

$$\frac{d}{d\theta}p_\theta(j|i) = \sum_{\alpha,y} \left[\nu_i(y)p^\alpha(j|i)\frac{d}{d\theta}\mu_y(\theta, \alpha)\right]. \tag{3.50}$$

We further assume that although the state X_n, $n = 0, 1, \ldots$, is not completely observable, the cost at any time $f(X_n)$ is known (e.g., by observation, or it depends only on the action). Then, the algorithms in (3.46) and (3.47) can be used with (3.50). If, in addition, there is a state i^*, which is irreducible for all policies, then the algorithm in (3.49) can be used. □

Finally, all the above algorithms are expressed in sample-path-based averages. Stochastic approximation based recursive algorithms can be developed based on these average-type algorithms. We will study these topics in Chapter 6.

Performance Derivatives for Queueing Systems

A direct learning algorithm for performance derivatives of queueing networks has been presented as Algorithm 2.2 in Section 2.4.1. An algorithm for the derivatives of the mean response time with respect to service rate in an M/G/1 queue is given in Example 2.10 in Section 2.4.3.

As explained in Section 2.4.3, Algorithm 2.2 directly estimates the performance derivative via $\sum_{\text{all } \boldsymbol{n}} \pi(\boldsymbol{n})c^{(f)}(\boldsymbol{n}, v)$ (see (2.108)) without estimating every perturbation realization factor $c^{(f)}(\boldsymbol{n}, v)$ and every steady-state probability $\pi(\boldsymbol{n})$ separately. The same explanation applies to the algorithm in Example 2.10 in Section 2.4.3.

It is interesting to note the difference in the process of developing the PA theory for both queueing systems and Markov systems. For queueing systems, the performance derivative estimation algorithms were developed first, and the concept of the perturbation realization factor and the performance derivative formula were developed later to provide a theoretical background for the algorithms. For Markov systems, the concept of performance potentials and performance derivatives were developed first, and the sample-path-based algorithms, both for potentials and for derivatives directly, were proposed later, by using the formulas.

The algorithms for estimating $c^{(f)}(\boldsymbol{n}, v)$ in queueing systems should be easy to develop; however, there has not been much effort in this direction, perhaps because there have not been many applications with $c^{(f)}(\boldsymbol{n}, v)$ alone so far. On the other hand, as we will see in Chapter 4, the estimated potentials can also be used in policy iteration optimization of Markov systems. In a recent study, the relation between the realization factors $c^{(f)}(\boldsymbol{n}, v)$ (with a queueing model) and the potentials $g(\boldsymbol{n})$ (with a Markov model) is established, and policy-iteration-type algorithms are developed for (customer-average) performance optimization of queueing systems based on $c^{(f)}(\boldsymbol{n}, v)$; see [260]. In such algorithms, the realization factors $c^{(f)}(\boldsymbol{n}, v)$ or their aggregations need to be calculated or estimated on sample paths.

3.3 Optimization with PA

3.3.1 Gradient Methods and Stochastic Approximation

The PA gradient estimates can be used to implement sample-path-based performance optimization. When the sample path is long enough, the estimates are very accurate and we can simply use them in any gradient-based optimization procedure [22, 23, 85] for deterministic systems. If the sample path is short, then the gradient estimates contain stochastic errors, and stochastic approximation techniques have to be used in developing optimization algorithms.

As shown in Figure 3.1, we will leave the stochastic approximation-based recursive algorithms to Chapter 6, in which we first introduce the related material in stochastic approximation in some detail. In this section, we discuss some fundamental methods in performance optimization with accurate estimates of the performance gradients.

Gradient Methods and the Robbins-Monro Algorithm

In general, we consider the optimization of a performance function $\eta(\theta) : \mathcal{D} \to \mathcal{R}$, where $\mathcal{R} = (-\infty, \infty)$ and $\mathcal{D} \subseteq \mathcal{R}^M$ is a convex M-dimensional parameter subset. Denote the performance gradients at any point $\theta \in \mathcal{D}$ as $\frac{d\eta(\theta)}{d\theta} := (\frac{\partial \eta(\theta)}{\partial \theta(1)}, \ldots, \frac{\partial \eta(\theta)}{\partial \theta(M)})^T$, where $\theta(i)$, $i = 1, 2, \ldots, M$, is the ith component

of θ. Let θ^* be a local optimal point of $\eta(\theta)$ in $\mathcal{D}$. We have $\frac{d\eta(\theta^*)}{d\theta} = 0$. We want to find out a local optimal point. This is a constrained optimization problem.

Suppose that the performance gradients $\frac{d\eta(\theta)}{d\theta}$ can be accurately estimated. We may find θ^* iteratively by using any gradient-based method (see, e.g., Chapter 2 of [23]). We start with an initial point $\theta_0 \in \mathcal{D}$. At the kth iteration, we run the system with parameter θ_k, $k = 0, 1, \ldots$, and apply PA on a long sample path to estimate the performance gradients at θ_k, $\frac{d\eta(\theta_k)}{d\theta}$. Set $h_k := \frac{d\eta(\theta_k)}{d\theta}$. In the simplest gradient method, the parameter θ is updated according to

$$\theta_{k+1} = \Pi_{\mathcal{D}}\left(\theta_k + \kappa_k h_k\right), \tag{3.51}$$

where $\Pi_{\mathcal{D}}$ denotes a projection onto $\mathcal{D}$, and $\kappa_k > 0$ is called a *step size*.

It can be shown that under some conditions on $\eta(\theta)$, θ_k converges to a local optimal point θ^* as $k \to \infty$, when κ_k is a small positive constant (e.g., in Example 6.1 in Chapter 6, $\theta_k \to \theta^*$ if $0 < \kappa_k = \kappa < 1$). Under some other conditions on $\eta(\theta)$, the convergence of θ_k requires $\kappa_k \to 0$ and $\sum_{k=0}^{\infty} \kappa_k = \infty$. The convergence of the algorithm (3.51) may be slow, and other methods such as Newton's method and Armijo's rule etc. can be used to improve the convergence rate. The detailed analysis of the gradient algorithms is beyond the scope of this book and can be found in, e.g., [23].

Because of the stochastic nature of the system, the gradient estimate obtained from any sample path with a finite length contains stochastic errors. We denote such a noisy (usually unbiased) estimate as

$$\hat{h}_k := \widehat{\frac{d\eta(\theta_k)}{d\theta}}.$$

The problem becomes to find the zeros of a function $\frac{d\eta(\theta)}{d\theta}$, which cannot be measured accurately. This is a topic in stochastic approximation (SA). With SA, we may simply replace the accurate value of the gradient in (3.51) with its estimate (cf. (6.6) and (6.7) in Chapter 6):

$$\theta_{k+1} = \Pi_{\mathcal{D}}\left(\theta_k + \kappa_k \hat{h}_k\right). \tag{3.52}$$

It is well known that with a properly chosen sequence of κ_k (in general $\sum_{k=1}^{\infty} \kappa_k = \infty$ and $\sum_{k=1}^{\infty} \kappa_k^2 < \infty$, e.g., $\kappa_k = \frac{1}{k}$; these conditions are more strict than those for the deterministic case (3.51)) and under some conditions for the noise in the gradient estimates $\hat{h}_k$ and for the performance function $\eta(\theta)$, we are guaranteed to obtain a sequence of θ_k that converges almost surely (with probability 1) to a local optimal point θ^*, as $k \to \infty$. Equation (3.52) corresponds to the *Robbins-Monro* algorithm in finding a zero point for the performance gradient; it will be discussed in greater detail in Chapter 6.

Sample-Path-Based Implementation

In sample-path-based implementation, the gradient estimation error depends on the length of the sample path. Therefore, the convergence of the optimization algorithm relies on the coordination among the lengths of sample paths in every iteration and the step sizes.

As discussed, there are two ways to implement optimization algorithms with PA. First, we can run a Markov, or a queueing, system under one set of parameters for a relatively long period to obtain an accurate gradient estimate and then update the parameters according to (3.51). When the estimation error is small, we hope that this standard gradient-based method for performance optimization of deterministic systems works well.

Second, when the sample paths are short, we need to use the stochastic approximation based algorithm (3.52). It is well known that the standard step size sequence (e.g., $\kappa_k = \frac{1}{k}$) makes the algorithm very slow, so some ad hoc methods are usually used in practice to speed up the convergence.

Both (3.51) and (3.52) take the same form and the difference is only on the choice of step sizes. On the other hand, there are always stochastic errors even when we run a relatively long sample path. Therefore, ad hoc methods are also useful even when we apply the deterministic version (3.51).

One of the ad hoc methods works as follows. When the sample path of the kth iteration is not long enough, we do not use the gradient estimate obtained when the system is under parameters θ_k in the kth iteration in (3.51). Instead, we may use a weighted sum of the current estimate under θ_k and the previous estimates under θ_{k-1}, θ_{k-2}, etc. as the gradient estimate. This may maintain the accuracy of the estimate since the step size is usually very small, (i.e., θ_k, θ_{k-1}, θ_{k-2} are very close) and, therefore, it may avoid instability caused by the large deviation of the gradient estimates due to the short length of each iteration and therefore it may speed up the convergence process.

There is a trade-off between the lengths of the sample paths and the number of iterations in reaching the optimal point. When the lengths are longer, fewer iterations may be required; and when the lengths are shorter, more iterations may be required. There are not much work in stochastic approximation dealing with the convergence speeds of the algorithms. Therefore, it is not clear which method, with long lengths or short ones, is faster (in terms of the number of transitions). Figure 3.4 illustrates the two optimization approaches with PA-based gradient estimates.

3.3.2 Optimization with Long Sample Paths

To illustrate the optimization approach with long sample paths, we consider the optimization of the system throughput η (see (2.95)) with respect to the mean service times, $\bar{s}_i$, $i = 1, 2, \ldots, M$, in a closed Jackson network (Section C.2). We assume that the mean service times must meet a constraint: The total mean service time is a constant, i.e., $\sum_{i=1}^{M} \bar{s}_i = const$, where "*const*" denotes

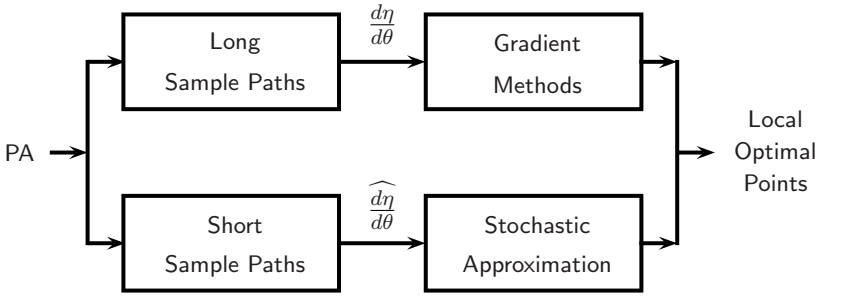

Fig. 3.4. Two Optimization Approaches with PA-Based Gradient Estimates

a constant. This constraint defines the region $\mathcal{D}$ in $\mathcal{R}^M$. The performance gradient

$$\frac{d\eta}{d\bar{s}} := \left[\frac{\partial\eta}{\partial\bar{s}_1}, \ldots, \frac{\partial\eta}{\partial\bar{s}_M}\right]^T$$

can be obtained by PA with a sample path of the queueing system. Let $\bar{s}_{i;k}$ be server i's mean service time at the kth iteration. We update the mean service times as follows:

$$\bar{s}_{i;k+1} = \bar{s}_{i;k} + \kappa_k \left\{\frac{\partial\eta}{\partial\bar{s}_i} - \frac{1}{M}\sum_{j=1}^{M}\frac{\partial\eta}{\partial\bar{s}_j}\right\}_{\bar{s}_i=\bar{s}_{i;k},\ i=1,\ldots,M}. \tag{3.53}$$

It can be easily verified that $\sum_{i=1}^{M}\bar{s}_{i;k} = const$, $k = 1, 2, \ldots$, as long as the initial values satisfy $\sum_{i=1}^{M}\bar{s}_{i;0} = const$.

Next, we provide a numerical example to show how the optimization approach works in practice. Some ad hoc modifications are added in the example to speed up the optimization process.

Example 3.8. Consider a closed Jackson network with $M = 3$ servers and $N = 5$ customers; let the routing matrix be

$$Q = \begin{bmatrix} 0 & 0.3 & 0.7 \\ 0.6 & 0 & 0.4 \\ 0.5 & 0.5 & 0 \end{bmatrix}.$$

The mean service times satisfy the constraint $\sum_{i=1}^{3}\bar{s}_i = 100$. We wish to maximize the system throughput.

We start with arbitrarily chosen initial values $\bar{s}_{1;0} = 80$, $\bar{s}_{2;0} = 10$, and $\bar{s}_{3;0} = 10$. We run the system with these initial values for 1,000 transitions and apply the PA algorithm to obtain an estimate of the performance gradient. Then, we follow (3.53) to update the mean service times. The initial length of 1,000 transitions is relatively short in estimating the gradients, because it is

expected that at the beginning the gradient is relatively large and therefore is easy to be estimated. The length will be adjusted in the parameter updating process. To speed up the convergence process, we apply the following modifications to the algorithm (3.53):

1. We choose the step size as

$$\kappa_k = a_1 \times a_2^r + b,$$

where $1 > \{a_1, a_2\} > 0$, and $b > 0$ are more or less arbitrarily chosen positive numbers, and r is the number of previous iterations that have resulted in degradation, rather than improvement, of the system performance. In this example, we choose $a_1 = 0.2$, $a_2 = 0.2$, and $b = 0.01$.
To speed up the process, we use an exponential decreasing step size rather than an inverse-proportional one. In addition, we reduce the step size only when the performance degrades, indicating that the update in the last iteration might be too large. Finally, we add a positive constant b to set up a lower bound for the step size. Theoretically, such a step size may not guarantee the convergence of the algorithm, but it may reach close enough to the optimal point.
2. If the performance degrades (or r increases), we quadruple the length of simulation in the next iteration to obtain a more accurate estimate of the performance gradient.
3. At each iteration, we update the gradient estimate by a weighted sum of the current estimate and the previous one as follows.

$$\left\{\frac{\bar{s}_i}{\eta}\frac{\partial \eta}{\partial \bar{s}_i}\right\}_{k+1} = w_1 \left\{\frac{\bar{s}_i}{\eta}\frac{\partial \eta}{\partial \bar{s}_i}\right\}_{k} + w_2 \left\{\frac{\bar{s}_i}{\eta}\frac{\partial \eta}{\partial \bar{s}_i}\right\}_{\text{the } (k+1)\text{th run}}, \tag{3.54}$$

where $w_1 = \frac{cL_k}{L_{k+1}+cL_k}$, $w_2 = \frac{L_{k+1}}{L_{k+1}+cL_k}$, and $c < 1$; L_k and L_{k+1} are the lengths of the kth and $(k+1)$th iterations, respectively. In (3.54), $\left\{\frac{\partial \eta}{\partial \bar{s}_i}\right\}_{k+1}$ is the value used in (3.53) to update the mean service times, and $\left\{\frac{\partial \eta}{\partial \bar{s}_i}\right\}_{\text{the } (k+1)\text{th run}}$ is the estimate obtained in the $(k+1)$th run.

After 36 iterations, the algorithm reaches a near-optimal point as

$$(\bar{s}_1, \bar{s}_2, \bar{s}_3) = (30.61,\ 39.69,\ 29.70),$$

with a throughput of 0.06512, as compared with the optimal value obtained by analytical formulas

$$(\bar{s}_1, \bar{s}_2, \bar{s}_3) = (30.58,\ 39.94,\ 29.49),$$

with the optimal throughput of 0.06513. □

In stochastic approximation based approaches with recursive algorithms, the system parameters can be updated within a short period or even at every transition. These topics are discussed in Chapter 6.

3.3.3 Applications

There have been hundreds of papers in the area of PA and its applications in the literature, and it is impossible to review all of them in this book. By and large, the applications cover a wide range of subjects such as capacity planning, inventory problems, resource allocation, flow control, bandwidth provisioning, traffic shaping, pricing, and stability and reliability analysis, in many areas including communications, networking, manufacturing, and logistics. References include [10, 35, 38, 74, 95, 144, 145, 158, 164, 180, 186, 187, 196, 199, 200, 204, 211, 210, 215, 224, 225, 233, 240, 258, 261, 263].

PROBLEMS

3.1. Study the potential with $g(S) = 0$:

a. Prove that the solution to (3.4) satisfies $p_{S*}g = \eta - f(S)$.
b. Derive (3.4) from the Poisson equation $(I - P)g + \eta e = f$ with the normalization condition $p_{S*}g = \eta - f(S)$.

3.2. Let P be an $S \times S$ ergodic stochastic transition matrix and ν be an S-dimensional (row) vector with $\nu e = 1$. Set $P_{-\nu} = P - e\nu$.

a. Suppose that there is a potential g such that $\nu g = \eta$. Prove that $g = P_{-\nu}g + f$.
b. Prove that the eigenvalues of $P - e\nu$ are 0 and λ_i, $i = 1, \ldots, S-1$, where λ_i, with $|\lambda_i| < 1$, $i = 1, \ldots, S-1$, are the eigenvalues of P.
c. Develop an iterative algorithm similar to (3.7).
d. For any vector ν with $\nu e = 1$, we can develop the algorithm in c) without presetting $\nu g = \eta$. Prove that the potential obtained by the algorithm indeed satisfies $\nu g = \eta$.
e. Prove that the algorithm (3.4)-(3.7) is a special case of the above algorithm. Verify that $p_{S*}g = \eta$.

3.3. For any vector ν with $\nu e = 1$,

a. Prove that $g = (I - P + e\nu)^{-1}f$ is a potential vector with normalization condition $\nu g = \eta$.
b. Can you derive a sample-path-based algorithm similar to (2.16) based on a)?

3.4. Consider

$$P = \begin{bmatrix} 0 & 0.5 & 0.5 \\ 0.7 & 0 & 0.3 \\ 0.4 & 0.6 & 0 \end{bmatrix}, \qquad f = \begin{bmatrix} 10 \\ 2 \\ 7 \end{bmatrix}.$$

a. Calculate the potential vector using algorithm (3.1).
b. Calculate the potential vector using algorithm (3.3).

c. Calculate the potential vector using algorithm (3.7).
d. Calculate the potential vector using the algorithm proposed in Problem 3.2.

Observe the convergence speeds and compare them with that of $\lim_{k\to\infty} P^k = e\pi$.

3.5. Suppose that a Markov chain starts from state i and that we use the consecutive visits to the state i as the regenerative points (cf. (3.18)). That is, we set

$$\begin{aligned} &l_0 = 0, \quad \text{with } X_0 = i, \\ &l_k = \text{the epoch that } \{X_l\} \text{ first visits state } i \text{ after } l_{k-1}, \qquad k \geq 1. \end{aligned}$$

Then we denote the first visit epoch to state j in the kth regenerative period as $l_{j,k}$; i.e., $l_{j,k} = \min\{l_{k-1} < l \leq l_k : \ X_l = j\}$. We note that in some periods, such a point may not exist. Can we use the average of the sum $\sum_{l=l_{k-1}}^{l_{j,k}-1} f(X_l)$ as the estimate of $\gamma(j,i)$? If not, why not?

3.6. Let $p(1|1) = 0.5$, $p(2|1) = 0.2$, and $p(3|1) = 0.3$; and $p(1|2) = 0.3$, $p(2|2) = 0.5$, and $p(3|2) = 0.2$. Suppose that $X = 1$ and $\widetilde{X} = 2$ and that we use the same uniformly distributed random variable $\xi \in [0,1)$ to determine the transitions from both $X = 1$ and $\widetilde{X} = 2$, according to (2.2). In this case, what are the conditional transition probabilities $\widetilde{p}_{1|1}(*|2)$, $\widetilde{p}_{2|1}(*|2)$, and $\widetilde{p}_{3|1}(*|2)$?

3.7. Let X and Y be two random variables with probability distributions $\Phi(x)$ and $\Psi(y)$, respectively. Their means are denoted as $\bar{x} = E(X)$ and $\bar{y} = E(Y)$. We wish to estimate $\bar{x} - \bar{y} = E(X - Y)$ by simulation. We generate random variables X and Y using the inverse transformation method. Thus, we have $X = \Phi^{-1}(\xi_1)$ and $Y = \Psi^{-1}(\xi_2)$, where ξ_1 and ξ_2 are two uniformly distributed random variables in $[0,1)$. Prove that if we choose $\xi_1 = \xi_2$, then the variance $Var[X - Y]$ is the smallest among all possible pairs of ξ_1 and ξ_2.

3.8. In the coupling approach, prove the following statements:

a. Let $\widehat{\pi}$ be the S^2 dimensional steady-state probability (row) vector of $\widehat{P}$, i.e., $\widehat{\pi}\widehat{P} = \widehat{\pi}$, and π be the steady-state probability vector of P, i.e., $\pi = \pi P$. Then $\widehat{\pi}(e_S \otimes I) = \widehat{\pi}(I \otimes e_S) = \pi$, and $\widehat{\pi}\widehat{g} = \widehat{\pi}\widehat{f} = 0$.
b. Equation (3.22) can take the form

$$(I - \widehat{P} + e_{S^2}\widehat{\pi})\widehat{g} = \widehat{f},$$

with $\widehat{\pi}\widehat{g} = 0$. Therefore, we have

$$\widehat{g} = \sum_{l=0}^{\infty} \widehat{P}^l \widehat{f}.$$

3.9. To illustrate the coupling approach used in simulation for speeding up the estimation of $\gamma(i,j)$, let us consider a simple Markov chain with transition probability matrix

$$P = \begin{bmatrix} 0.2 & 0.3 & 0.5 \\ 0.2 & 0.3 & 0.5 \\ 0.2 & 0.3 & 0.5 \end{bmatrix}.$$

a. Suppose that we generate two independent Markov chains with initial states $X_0 = 1$ and $X_0' = 2$, respectively. What is the average length from $l = 0$ to L_{12}^*, $E(L_{12}^*)$?
b. If we use the same $[0,1)$ uniformly distributed random variable ξ to determine the state transitions for both Markov chains, what is $E(L_{12}^*)$?
c. Answer the questions in a) and b), if

$$P = \begin{bmatrix} 0.2 & 0.4 & 0.4 \\ 0.4 & 0.2 & 0.4 \\ 0.4 & 0.4 & 0.2 \end{bmatrix}.$$

3.10. The realization factor $\gamma(i,j)$ can be obtained by simulating two sample paths initiated with i and j, respectively, up to its merging point $L_{i,j}$:

$$\gamma(i,j) = E\left\{ \sum_{l=0}^{L_{i,j}-1} [f(X_l') - f(X_l)] \Big| X_0 = i, X_0' = j \right\}.$$

If the two sample paths are independent, as shown in the text, we can obtain the perturbation realization factor equation. However, in simulation, we may use coupling to reduce the variance in estimating the difference of the mean values of two random variables ($\gamma(i,j) = g(j) - g(i)$). In our case, we wish to let the two sample paths, initiated with i and j, merge as early as possible.

To this end, in simulation we can force the two sample paths $\boldsymbol{X}$ and $\boldsymbol{X}'$ with two initial states i and j, respectively, to merge as fast as possible. We may use the same random variable to determine the state transitions in the two paths. For example, if $p(k|i) = 0.3$ and $p(k|j) = 0.2$, instead of using two independent random numbers in $[0,1)$ to determine the state transitions for $X_0 = i$ and $X_0' = j$, respectively, we generate one uniformly distributed random number $\xi \in [0,1)$, and if $\xi \in [0, 0.2)$, we let both $X_1 = X_1' = k$.

We use an example to show this coupling method: Let $p(1|2) = 0.5$, $p(2|2) = 0.3$, $p(3|2) = 0.2$, and $p(1|3) = 0.2$, $p(2|3) = 0.7$, $p(3|3) = 0.1$. The largest probabilities for the two paths starting from $X_0 = 2$ and $X_0' = 3$ to merge at $X_1 = X_1' = 1$ is $\min\{p(1|2), p(1|3)\} = 0.2$, to merge at $X_1 = X_1' = 2$ is $\min\{p(2|2), p(2|3)\} = 0.3$, and to merge at $X_1 = X_1' = 3$ is $\min\{p(3|2), p(3|3)\} = 0.1$. Thus, the largest probability that the two sample paths merge at $X_1 = X_1'$ with the coupling technique is $0.2+0.3+0.1 = 0.6$. We simulate the two sample paths in two steps. In the first step, we generate a uniformly distributed random variable $\xi \in [0,1)$. If $\xi \in [0, 0.2)$, we set $X_1 = X_1' = 1$; if $\xi \in [0.2, 0.5)$, we set $X_1 = X_1' = 2$; if $\xi \in [0.5, 0.6)$, we

set $X_1 = X_1' = 3$. If $\xi \in [0.6, 1)$, we go to the second step: using two other independent random numbers to determine the transitions for the two sample paths.

Continue with the above reasoning and mathematically formulate it. Work on $\gamma(i, S)$ for all states $i \in \mathcal{S}$ and derive the following equation

$$g(i) - g(S) = f(i) - f(S) + \sum_{j=1}^{S} [p(j|i) - p(j|S)]g(j), \qquad i \in \mathcal{S}.$$

Prove that this equation is the same as (3.4).

3.11. One of the restrictions of the basic formula (3.32) is that it requires $p(j|i) > 0$ if $\Delta p(j|i) > 0$ for all $i, j \in \mathcal{S}$. This condition can be relaxed. For example, we may assume that whenever $\Delta p(j|i) > 0$, there exists a state, denoted as $k_{i,j}$, such that $p(k_{i,j}|i)p(j|k_{i,j}) > 0$. Under this assumption, we have

$$\frac{d\eta_\delta}{d\delta} = \sum_{i\in\mathcal{S}} \sum_{j\in\mathcal{S}} \left\{ \pi(i) \left[p(k_{i,j}|i)p(j|k_{i,j}) \frac{\Delta p(j|i)}{p(k_{i,j}|i)p(j|k_{i,j})} g(j) \right] \right\}.$$

Furthermore, we have

$$\frac{d\eta_\delta}{d\delta} = \sum_{i\in\mathcal{S}} \sum_{j\in\mathcal{S}} \left\{ \pi(i) \Big[\sum_{k\in\mathcal{S}} p(k|i)p(j|k) \Big] \left[\frac{\Delta p(j|i)}{\sum_{k\in\mathcal{S}} p(k|i)p(j|k)} g(j) \right] \right\}.$$

a. Continue the analysis and develop the direct learning algorithms for the performance derivatives.
b. Compared with (3.32), what are the disadvantages of this "improved" approach, if any?
c. Extend this analysis to the more general case of irreducible Markov chains.

3.12. In the gradient estimate (3.34), we have ignored the constant term η in the expression of g. A more accurate estimate should be

$$\frac{d\eta_\delta}{d\delta} \approx \lim_{N\to\infty} \frac{1}{N} \left\{ \sum_{n=0}^{N-1} \left[\frac{\Delta p(X_{n+1}|X_n)}{p(X_{n+1}|X_n)} \sum_{l=0}^{L-1} [f(X_{n+l+1}) - \eta] \right] \right\}, \quad \text{w.p.1.}$$

Prove that

$$\frac{d\eta_\delta}{d\delta} \approx \lim_{N\to\infty} \frac{1}{N} \left\{ \sum_{n=0}^{N-1} \left[\frac{\Delta p(X_{n+1}|X_n)}{p(X_{n+1}|X_n)} \sum_{l=0}^{L-1} f(X_{n+l+1}) \right] \right\}, \quad \text{w.p.1,}$$

and discuss the estimation error caused by ignoring the term $L\eta$ in the estimate.

3.13. Discuss the error in the gradient estimate (3.41) caused by ignoring the second term of (3.40) for a finite N. You may set $f \equiv 1$.

3.14. Let η_r be the performance of a Markov chain with transition probability matrix P_r defined as $p_r(i|i) = r$ for all $i \in \mathcal{S}$ and $p_r(j|i) = (1-r)q_{i,j}$, $j \neq i$, $i, j \in \mathcal{S}$, with $\sum_{j \in \mathcal{S}} q_{i,j} = 1$ for all $i \in \mathcal{S}$. Prove $\frac{d\eta_r}{dr} = 0$ for all $0 < r < 1$ using performance derivative formula (3.30).

3.15. In Algorithm 3.1, prove that the following equation holds

$$\lim_{L \to \infty} \left\{ \sum_{l=0}^{L-1} P^l (\Delta P) P^{L-l-1} \right\} = e\pi(\Delta P)(I - P + e\pi)^{-1}.$$

In addition, prove that, at the steady state, we have

$$\begin{aligned} \pi(i)\rho_L(i) &= E\left\{ \sum_{l=0}^{L-1} \frac{\Delta p(X_{l+1}|X_l)}{p(X_{l+1}|X_l)} I_i(X_L) \right\} \\ &= \pi \left\{ \sum_{l=0}^{L-1} P^l (\Delta P) P^{L-l-1} \right\} e_{\cdot i}, \end{aligned}$$

where $e_{\cdot i}$ is the ith column vector of the identity matrix I. Equation (3.38) and the convergence of (3.37) follow directly from these two equations.

3.16. In Problem 3.15, we set $G_L = \sum_{l=0}^{L-1} P^l (\Delta P) P^{L-l-1}$. Prove that

$$G_{L+1} = PG_L + G_L P - PG_{L-1}P,$$

with $G_0 = 0$, $G_1 = \Delta P$. Set $G = \lim_{L \to \infty} G_L$. Explain the meaning of G. Finally, letting $L \to \infty$ on both sides of the above equation, we obtain $G = PG + GP - PGP$. Is this equation useful in any sense?

3.17. Write a computer simulation program

a. to estimate potentials by using (3.15) and (3.19);
b. to estimate the performance derivatives by using (3.35), (3.41), and (3.43).

3.18. The group inverse (2.48) $B^{\#} = -[(I - P + e\pi)^{-1} - e\pi]$ (for ergodic chains) plays an important role in performance sensitivity analysis. Let $b^{\#}(i,j)$ be the (i,j)th component of $B^{\#}$. Consider a Markov chain starting from state $i \in \mathcal{S}$. Let $N_{ij}^{(L)}$ be the expected number of times that the Markov chain visits state $j \in \mathcal{S}$ in the first L stages. Prove (cf. [168]) that

$$\lim_{L \to \infty} \left(N_{ji}^{(L)} - N_{ki}^{(L)} \right) = b^{\#}(k,i) - b^{\#}(j,i).$$

3.19. Given a direction defined by ΔP, is it possible to estimate the second-order derivative $\frac{d^2\eta_\delta}{d\delta^2}$ using a sample path of the Markov chain with transition probability matrix P (cf. Section 2.1.5)? How about the second-order performance derivatives of any given reward function $f(\theta)$?

3.20. Consider a continuous-time Markov process with transition rates $\lambda(i)$ and transition probabilities $p(j|i)$, $i, j = 1, 2, \ldots, S$. Suppose that the transition probability matrix $P := [p(j|i)]_{i,j \in \mathcal{S}}$ changes to $P + \delta \Delta P$, and the transition rates $\lambda(i)$, $i = 1, 2, \ldots, S$, remain unchanged. Let η be the average reward with reward function f. Develop a direct learning algorithm for $\frac{d\eta_\delta}{d\delta}$.

3.21. Consider a closed Jackson network consisting of M servers and N customers with mean service times $\bar{s}_i$, $i = 1, 2, \ldots, S$, and routing probabilities $q_{i,j}$, $i, j = 1, 2 \ldots, M$. Let

$$\eta_T^{(f)} = \lim_{L \to \infty} \frac{1}{T_L} \int_0^{T_L} f(\boldsymbol{N}(t)) dt$$

be the time-average performance. Suppose that the routing probabilities change to $q_{i,j} + \delta \Delta q_{i,j}$, $i, j = 1, 2 \ldots, M$. Develop a direct learning algorithm for the derivative of the time-average reward using performance potentials. Use the intuition explained in Section 2.1.3 to develop the performance derivative formula.