Proceedings of the 1988 Winter Simulation Conference
M. Abrams, P. Haigh, and J. Comfort (eds.)

Using perturbation analysis for gradient estimation, averaging and updating in a stochastic approximation algorithm

Michael C. Fu
Yu-Chi Ho
Division of Applied Sciences
Harvard University
Cambridge, MA 02138

ABSTRACT

We propose a gradient updating procedure for using both "present" and "past" data to improve the convergence properties of a stochastic approximation algorithm. This procedure utilizes second derivatives estimated by perturbation analysis techniques. Experimental evidence provided by simulation runs appear to confirm the improvement in convergence rate gained by this modified algorithm.

1. INTRODUCTION AND OVERVIEW

In the optimization of a performance measure in a discrete-event system, major difficulties often arise due to the stochastic nature of the system, to the point that modelling via simulation is necessary. Perturbation analysis provides a means by which to estimate gradients of performance measures with respect to system parameters from a single simulation run, a valuable tool in simulation optimization. In an extensive experimental study, Suri & Leung [1988] demonstrated experimentally the feasibility of utilizing perturbation analysis in a stochastic approximation algorithm for the optimization of a very simple stochastic system, the M/M/1 queue.

In applying stochastic approximation algorithms to the optimization of a performance measure of a stochastic system, there is often a basic tradeoff between the accuracy of an individual gradient estimate and the number of iterations taken. For example, in optimizing a performance measure of a queueing system, one can observe large numbers of customers between occasional changes of the control parameter or observe small numbers of customers with frequent changes of the control parameter. The accuracy of an estimate (e.g., its variance) in general increases with the number of observations or measurements taken, so that given a fixed measurement budget, it is usually not obvious whether it is preferable to have more iterations with less accurate estimates or vice versa. This tradeoff occurs in stochastic approximation algorithms because each estimate of the gradient is taken at a different value of the control parameter. In the original formulation of stochastic approximation, only measurements at the current value of the control parameter are used; those taken at previous values of the control parameter are discarded. The Suri & Zazanis and Suri & Leung algorithms follow this route, making no use of "past" data.

To improve their algorithm, we wished to combine both "past" and "present" observation data to form a better estimation of the gradient. We propose a type of gradient averaging found in more recent modified versions of stochastic approximation algorithms [Ruszynski & Syski, 1983; Fogel, 1981], and a gradient updating through the use of higher derivative estimates - a new addition made practical by the nature of perturbation analysis. The gradient averaging method can be viewed as a crude means of using previous measurements in improving the present estimate of the gradient. The addition of updating is a more refined means of utilizing "past" data, in which, intuitively, we ask how measurements taken at previous values of the control parameter can be extrapolated to the current value of the control parameter. The use of higher derivatives is a natural answer, and in theory, perturbation analysis can be used not only to generate gradient estimates needed for optimization but also higher derivatives which can be used in the extrapolation process. In our updating/averaging algorithm, the gradient estimate is updated each time the parameter is changed by combining the (updated) gradient estimate used at the previous parameter value with the gradient estimate observed at the present parameter value. The G/G/1 queue, where second derivative estimates of system time are readily available, was used as a means for comparison with the original Suri and Leung algorithm.

2. STOCHASTIC APPROXIMATION (SA)

A thorough discussion and rigorous theoretical analysis of SA can be found in Nevelson & Zalmanovich [1976] and Kushner & Clark [1978], and a good overview of its use in simulation optimization are contained in Glynn [1986] and Meketon [1987]. Here, we summarize briefly classical and more recent results.

The problem under consideration is the following:

$$\min_{x \varepsilon R^k} J(x) \equiv E[f(x,\omega)],\ f: R^k \times \Omega \rightarrow R,$$

where $\omega \varepsilon \Omega$ represents the stochastic effects and J(x) represents the performance measure of the stochastic system. There are usually additional constraints on the parameter.

Under certain smoothness and convexity conditions on J(x), we can find the minimum by solving instead $dJ(x)/dx = dE[f(x,\omega)]/dx = 0$. If $g(x,\omega)$ is an estimate of dJ/dx s.t. $E[g(x,\omega)] = dJ/dx \equiv g^*(x)$, i.e., an unbiased estimate, we equivalently solve the problem $g^*(x)=0$. Stochastic approximation was originally formulated to address this level-crossing problem.

Two SA algorithms, RM [Robbins & Monro, 1951] and KW [Kiefer & Wolfowitz, 1952], were adapted in Suri/Leung's experimental study of the M/M/1 queue. Both algorithms are basically of the same form

$$x_{n+1} = x_n + a_n g(x_n,\omega),$$

except that $g(x_n,\omega)$ is estimated by finite differences in the latter algorithm. In Suri and Leung, the superiority of RM using PA to estimate the gradient over KW using finite difference estimates was demonstrated experimentally for the M/M/1 queue. This is not surprising, since RM yields convergence of order $1/n^{1/2}$ if the estimate g(x) is unbiased, while for KW (in all but very special cases) the convergence is of order $1/n^{1/3}$ [Sacks, 1958]. Also, Zazanis & Suri [1984] showed that for the M/G/1 queue, the IPA estimate of the gradient for the average system time with respect to mean service time converges to the true gradient like 1/M, whereas finite difference estimates converge like $1/M^{1/2}$, where M is the number of observed busy periods.

More recently, SA has been extended to multistep methods, the main idea being "to provide the basic optimization procedure with auxiliary filters averaging past observational data." [Ruszczynski & Syski, 1983] One such two-step algorithm is the following: [Fogel, 1981]

$$x_{n+1} = x_n - a_n d_{n+1},$$
$$d_{n+1} = d_n + b_n(g(x_n,\omega) - d_n).$$

Intuitively, d_n represents the "old" direction, $g(x_n,\omega)$ represents the "new" gradient estimate, b_n represents the averaging coefficient, and d_{n+1} is the resulting new direction. With $b_n=1$, the two-step version reduces to the regular SA algorithm. The basic convergence requirement, in addition to requirements on the coefficients, is that $g^*(x)=E[g(x,\omega)]$ be Lipschitz with respect to x, i.e., there is a C such that $|g^*(y)-g^*(z)| \leq C|y-z|$ for all y,z. Ruszczynski & Syski implemented a practical version of the gradient averaging method to demonstrate numerically the superiority in convergence rate it had over the standard SA algorithm.

The SA algorithm we propose is yet another extension of the above, given in three-step general form as follows:

$$x_{n+1} = x_n - a_n d_{n+1},$$
$$d_{n+1} = \underline{d}_n + b_n(g(x_n,\omega) - \underline{d}_n),$$
$$\underline{d}_n = d_n + T(x_n, \Delta x_n),$$
where $\Delta x_n = x_n - x_{n-1}$.

We call this our SA with gradient averaging and updating. Conceptually, the first line is the usual SA, the second line is the combining of the "new" gradient estimate and an updated version of the "old" previous direction, and the third line is the updating of the previous direction at the updated parameter value, with $T(x_n,\Delta x_n)$ representing the updating function via higher derivatives. In Section IV, we apply a version of this algorithm to four test cases.

3. PERTURBATION ANALYSIS (PA)

The basic motivation behind PA is to extract more information out of a single sample path (corresponding, say, to a single simulation) than has been generally thought possible [Ho, 1979]. Specifically, PA has been realized as a means by which to compute derivatives of performance measures with respect to system parameters, using a single sample path of the system (either simulated or real) to "reconstruct" a perturbed path with minimal additional effort. Because only a single sample path is needed to compute the gradient, PA seems a logical choice for an RM-type optimization algorithm.

PA is quite general for discrete event dynamic systems (DEDS). Numerous papers have shown implementations for various types of systems, including single-server queues [Suri & Zazanis, 1988; Zazanis & Suri, 1985] and queueing networks [Ho & Cao, 1983; Cao, 1986]. Suri [1987] and Cao [1985] provided theoretical foundations for infinitesimal perturbation analysis (IPA), the earliest form of PA, e.g., in proving the consistency of the *sample* gradient estimates for certain systems. The applicability of IPA came down to a question of interchangeability in the order of expectation and differentiation, i.e., does

$$\frac{\partial}{\partial\theta} E[f(\theta,\omega)] = E[\frac{\partial}{\partial\theta} f(\theta,\omega)] \ ?$$

(NB: henceforth, we adopt θ to represent the adjustable parameter, replacing x used in the SA discussion.) Recently, work by Ho & Li [1988] and by Gong & Ho [1987] has overcome some of these difficulties. More detailed explanations of the techniques and theory, which can be found in these and

other papers, will not be undertaken here.

IPA, as the earliest technique developed, is thus, in some sense, the most well-developed. For the G/G/1 queue, the theoretical work is very comprehensive, with most of these results brought together in one work, Zazanis's Ph.D. thesis [1986b]. To summarize, he does the following: proves strong consistency and unbiasedness for the gradient of mean system time w.r.t. a parameter θ; demonstrates how strongly consistent second and higher order derivative estimates can be obtained from a single sample path; proves the asymptotic superiority of PA estimates over conventional finite difference estimates; introduces the single-run optimization method utilizing PA in a preliminary experimental study.

4. SINGLE-RUN OPTIMIZATION OF SYSTEM TIME IN A SINGLE-SERVER QUEUE

4.1 The Problem

To test the algorithms, we applied them to the optimization of a simple stochastic system. Specifically, we wished to minimize the average system time of a customer in a single-server queue subject to some penalty costs, i.e.,the cost function is of the following form:

$$J(\theta) = E[L(\theta,\omega)] = E[T(\theta,\omega) + C(\theta)] = E[T] + C(\theta),$$

where T = system time of a customer in the system,
θ = controllable parameter (possibly vector),
ω = sample path of the system,
$C(\theta)$ = penalty cost, a function of θ only.

The gradient is

$$\frac{dJ}{d\theta} = \frac{\partial E[T]}{\partial \theta} + C'(\theta),$$

and as described in Section III, infinitesimal perturbation analysis can be used to estimate $\partial E[T]/\partial\theta$, while all the other quantities are assumed known. We wish to find θ such that $dJ/d\theta=0$, subject to certain constraints on θ. In this section, we present the results for four examples. We chose analytically tractable systems, because this makes it easier to evaluate the algorithms by checking with the known theoretical optimum.

4.2 The Algorithms

In this section, let $g_{PA}(\theta_n,\omega)$ be the PA estimate of $dE[T]/d\theta$ taken over some fixed number of customers L (i.e., L=# customers/iteration). The algorithm used by Suri & Leung [1987] in their empirical study, which we shall refer to as A0, is basically the original SA algorithm:

$$\theta_{n+1} = \theta_n - a_n J'_n, \qquad [A0]$$

where $J'_n = g_{PA}(\theta_n,\omega) + C'(\theta)$,

and where a_n is the accelerated harmonic sequence [Kesten,1958] defined by

$$1/a_{n+1} = \begin{cases} 1/a_n & \text{if } \operatorname{sgn} J'_{n+1} = \operatorname{sgn} J'_n \\ (1/a_n)+1 & \text{if } \operatorname{sgn} J'_{n+1} \neq \operatorname{sgn} J'_n. \end{cases}$$

The simplest extension, in which we do straight averaging of the gradient with no updating, we call A1, is the following:

$$\theta_{n+1} = \theta_n - a_n J'_{n+1}, \qquad [A1]$$

where $J'_{n+1} = g_{\underline{n}} + C'(\theta)$,

$$g_{\underline{n}} = g_{\underline{n-1}} + b_n(g_{PA}(\theta_n,\omega) - g_{\underline{n-1}}), \qquad \text{(averaging)}$$

and where b_n is the harmonic series (1/n) and a_n is the accelerated harmonic sequence defined above. Note that no extra data collection is needed in this scheme.

In the next extension, we add the updating aspect afforded by second derivative PA estimates. This, which we call A2, is the following:

$$\theta_{n+1} = \theta_n - a_n J'_{n+1}, \qquad [A2]$$

where $J'_{n+1} = g_{\underline{n}} + C'(\theta)$,

$$g_{\underline{n}} = g_n + b_n(g_{PA}(\theta_n,\omega) - g_n), \qquad \text{(averaging)}$$

$$g_n = g_{\underline{n-1}} + g'_{PA*}(\theta_n,\omega)\,\Delta\theta_n, \qquad \text{(updating)}$$

where $\Delta\theta_n=\theta_n-\theta_{n-1}$ and a_n and b_n are as defined in A1. $g'_{PA*}(\theta_n,\omega)$ represents an "averaged" PA estimate of $\partial^2 E[T]/\partial\theta^2$ taken over the entire optimization run up to that point. In implementation terms, this means that the accumulations for calculating the second derivative via PA [viz. Zazanis & Suri, 1985] are not reset after every iteration, as is the case for the gradient estimate $g_{PA}(\theta_n,\omega)$. The implicit assumption is that the second derivative is "more constant" than the gradient over a given range of θ, so that, heuristically speaking, a reduction in the variance of the estimate would occur which would outweigh any introduction of bias. Of course, the updating scheme means that there is additional computation associated with calculating the second derivative, but as in most PA implementations, it is a relatively minor, and easy, addition.

In the implementation of A1 and A2, a "reset" mechanism was added to the updating/averaging of the gradient estimate, occuring when $\Delta\theta_n > 0.3\theta_n$ or when the constraints were violated. The intention was to stabilize the algorithm in case of "glitches" in the gradient estimates. Intuitively, the reset mechanism is such that if the step size taken is "large" in some sense, then the averaging/updating process is restarted from scratch by means of setting b_n back to 1, so that $g_{\underline{n}}=g_{PA}(\theta_n,\omega)$.

For all algorithms, a projection scheme was used when the constraints were violated.

4.3 Comparing the Algorithms

For comparison purposes, we fixed the total number of customers observed in each run, eschewing the problem of determining an appropriate stopping criterion. In this way, both convergence to the correct value and convergence rate could be compared.

Let θ^i = optimized value of the parameter for replication i,
θ^* = true optimal value of the parameter,
$J_i = J(\theta^i)$ = optimized value of the objective function for replication i,
$J^* = J(\theta^*)$ = true optimal (minimum) value of the objective function.

We measure the "goodness" of the algorithms by comparing these values. Error in $J(\theta)$, the means by which Suri & Leung used to evaluate their algorithm, is defined by

$$\bar{e} = \text{mean \% error in } J = \frac{\sum e_i}{n} = \frac{\bar{J} - J^*}{J^*},$$

$$\sigma_e = \text{sd \% error in } J = \sqrt{\frac{\sum (e_i - \bar{e})^2}{n-1}} = \frac{\sigma_J}{J^*},$$

$$\text{where } e_i = \text{\% error in } J_i = \frac{J_i - J^*}{J^*}$$

$$\text{and } \bar{J} = \frac{\sum J_i}{n}, \quad \sigma_J^2 = \frac{\sum (J_i - \bar{J})^2}{n-1},$$

and n = # replications.

A normalized mean-squared error (MSE) combining the two terms was calculated as the sum of the variance and the squared bias:

$$\varepsilon^2 = \text{\% MSE} = \bar{e}^2 + \sigma_e^2.$$

In all the examples that follow, the results are taken over 40 independent replications, i.e., n=40. For comparison in the one-dimensional parameter cases, both error in $J(\theta)$ and in θ were used, with the statistics for θ (or its components in the multi-dimensional examples) calculated in the same manner as for J.

Background. The experimental study of Suri & Leung was a comprehensive empirical "verification" of the promising results of single-run optimization begun by Zazanis utilizing PA gradient estimates of the G/G/1 queue. The objective was to find θ to miminize the following cost function:

$J(\theta) = E[T] + c_1/\theta$, subject to $0<\rho<1$ and $\theta>0$,
where θ = parameter of the service time distribution,
c_1 = cost per unit speed of the server (≥ 0),
ρ = utilization of the server.

Thus, we have

$$\frac{dJ}{d\theta} = \frac{\partial E[T]}{\partial \theta} - c_1/\theta^2,$$

and hence $J' = c_0 g - c_1/\theta^2$, where g is the estimate of $\partial E[T]/\partial\theta$. In the Suri & Leung study, the M/M/1 queue was used because of its analytic tractability. For comparison, this was thus a natural first test case.

M/M/1 Test Case. For the M/M/1, with θ the mean service time and λ the arrival rate, we have

$$E[T] = \theta/(1-\lambda\theta),$$

so the minimum of J is achieved at

$$\theta = \frac{1}{\sqrt{\frac{1}{c_1} + \lambda}}.$$

Again, we reiterate that in actual optimization, the analytical formula for E[T] is assumed unknown, i.e., it is only estimated through simulation, and its derivative is estimated through PA techniques. The analytic optimum is used only a posteriori for comparison with the final parameter value obtained from a single-run optimization run.

As in Suri & Leung, we took λ=0.01, θ_0=50.0 (starting value), and studied the five cases c_1=2000, 8000, 32000, 128000, 512000, corresponding to optimal operation at traffic intensities of ρ_{opt} = 0.31, 0.47, 0.64, 0.78, and 0.87. Also as in the Suri & Leung study, we ran each case at L=5, 10, 20, and 40 customers per iteration. To study the tradeoff between number of iterations and accuracy of the gradient estimate, the total number of customers observed for each case was the same. This means, for example, that a 1000 total customer "budget" would correspond to 200 iterations at L=5 customers/iteration but only 25 iterations at L=40 customers/iteration. Because we expected the runlength needed for convergence to increase with increasing ρ_{opt} (as reflected in the Suri & Leung study), we used the following run lengths:

c_1	2000	8000	32000	128000	512000
ρ_{opt}	0.31	0.47	0.64	0.78	0.87
run length	10,000	20,000	30,000	50,000	100,000

The calculated mean and standard deviation % error for both θ and J are given in Table 1, and the resulting root-%mean-squared errors for $\varepsilon(J)$ are displayed graphically in the plots of Graph 1, from which we observe the following (plots for $\varepsilon(\theta)$, though omitted here, yield the same conclusions): (i) The accuracies of A2 and A1 are more uniform than A0 over different L for a given ρ_{opt}, i.e., they are less dependent on the sample size per iteration. (ii) A2 was clearly the most accurate in the three highest ρ_{opt} cases, with approximately equally best accuracy attained for A2 and A1 in the low ρ_{opt} case, and A2 and A0 in the second lowest ρ_{opt} case. Thus, (iii) A2 always did as well as or much better than the other two algorithms in all cases.

Table 1: M/M/1 Test Case Simulation Results

	c1														
	2000			8000			32000			128000			512000		
L	A0	A1	A2	A0	A1	A2	A0	A1	A2	A0	A1	A2	A0	A1	A2
5	2.58	0.09	0.16	0.05	0.33	0.11	0.02	0.00	0.03	0.16	0.24	0.07	0.77	0.02	0.15
10	2.36	0.12	0.22	0.01	0.30	0.09	0.04	0.01	0.06	0.20	0.24	0.02	0.50	0.02	0.12
20	2.42	0.06	0.29	0.04	0.30	0.09	0.07	0.01	0.06	0.15	0.25	0.03	0.37	0.07	0.11
40	1.84	0.04	0.35	0.10	0.28	0.11	0.07	0.01	0.06	0.25	0.26	0.08	0.12	0.09	0.08

Absolute Mean %error in θ

	c1														
	2000			8000			32000			128000			512000		
L	A0	A1	A2	A0	A1	A2	A0	A1	A2	A0	A1	A2	A0	A1	A2
5	2.39	0.94	0.91	0.78	1.12	0.80	1.53	1.45	0.97	2.92	1.57	0.83	6.39	1.41	0.63
10	2.72	0.94	0.97	0.83	1.12	0.80	1.39	1.43	0.98	2.43	1.57	0.87	5.69	1.41	0.64
20	2.43	0.94	0.97	0.87	1.10	0.78	1.28	1.40	0.95	2.16	1.56	0.86	4.65	1.42	0.62
40	2.01	0.97	0.97	0.91	1.10	0.78	1.23	1.40	0.97	2.03	1.56	0.88	3.82	1.55	0.66

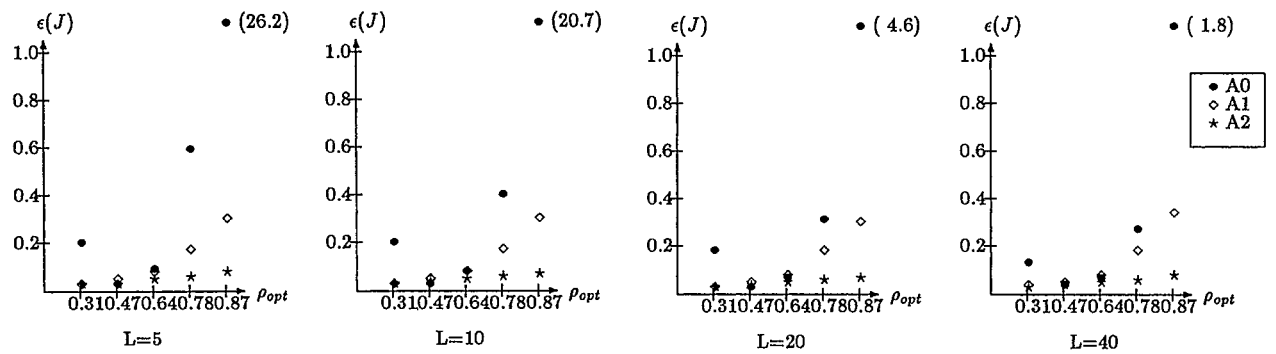
Std Dev %error in θ

	c1														
	2000			8000			32000			128000			512000		
L	A0	A1	A2	A0	A1	A2	A0	A1	A2	A0	A1	A2	A0	A1	A2
5	0.10	0.01	0.01	0.01	0.02	0.01	0.05	0.04	0.02	0.31	0.09	0.03	8.49	0.15	0.03
10	0.11	0.01	0.01	0.01	0.02	0.01	0.04	0.04	0.02	0.22	0.09	0.03	6.57	0.15	0.03
20	0.10	0.01	0.01	0.01	0.02	0.01	0.03	0.04	0.02	0.17	0.09	0.03	2.19	0.15	0.03
40	0.06	0.01	0.01	0.01	0.02	0.01	0.03	0.04	0.02	0.15	0.09	0.03	1.13	0.18	0.03

Mean %error in J

	c1														
	2000			8000			32000			128000			512000		
L	A0	A1	A2	A0	A1	A2	A0	A1	A2	A0	A1	A2	A0	A1	A2
5	0.15	0.01	0.01	0.01	0.02	0.01	0.06	0.05	0.03	0.48	0.12	0.03	24.79	0.24	0.05
10	0.15	0.01	0.01	0.01	0.02	0.01	0.04	0.04	0.03	0.31	0.12	0.03	19.62	0.24	0.04
20	0.13	0.01	0.01	0.01	0.02	0.01	0.04	0.04	0.03	0.23	0.13	0.03	4.10	0.24	0.05
40	0.09	0.02	0.01	0.01	0.02	0.01	0.03	0.04	0.03	0.20	0.14	0.03	1.43	0.26	0.05

Std Dev %error in J



Graph 1: M/M/1 Test Case

M/D/1 Test Case. Our second test case utilized the M/D/1 queue. For the M/D/1, with θ the service time and λ the arrival rate, the expected system time can be found using the P-K formula [e.g., Kleinrock]:

$$E[T] = \theta + \frac{\lambda\theta^2}{2(1-\lambda\theta)} .$$

The minimality condition leads to the fourth-degree polynomial equation

$$\rho^4 - 2\rho^3 + 2(1-d)\rho^2 + 4d\rho - 2d = 0 ,$$

where $\rho = \lambda\theta$ and $d = c_1\lambda^2$.

Somewhat fortuitously, the solution to this quartic falls out rather easily. Since $\rho\varepsilon(0,1)$, our desired minimum is achieved at

$$\rho_{opt} = \frac{1 - \sqrt{1+2d} + \sqrt{2(d-1)+2\sqrt{1+2d}}}{2}$$

As before, we took λ=0.01, θ_0=50.0, and studied the following five cases:

c_1	1250	5000	15000	75000	240000
ρ_{opt}	0.29	0.47	0.62	0.79	0.87
run length	10,000	20,000	30,000	50,000	100,000

Again, we ran each case at L=5, 10, 20, and 40 customers per iteration. From the results displayed in Table 2 and Graphs 2a through 2h, we see that basically the same conclusions can be drawn here as in the M/M/1 test case.

Table 2: M/D/1 Test Case Simulation Results

	c1														
	1250			5000			15000			75000			240000		
L	A0	A1	A2	A0	A1	A2	A0	A1	A2	A0	A1	A2	A0	A1	A2
5	1.84	0.04	0.15	0.62	0.05	0.05	0.08	0.11	0.11	0.09	0.04	0.01	0.31	0.29	0.14
10	2.15	0.01	0.18	0.58	0.02	0.05	0.06	0.12	0.12	0.04	0.05	0.04	0.33	0.28	0.15
20	1.91	0.16	0.32	0.66	0.00	0.02	0.03	0.16	0.11	0.03	0.06	0.09	0.02	0.29	0.19
40	1.94	0.54	0.46	0.86	0.02	0.02	0.01	0.14	0.12	0.01	0.09	0.07	0.09	0.31	0.20

Absolute Mean %error in θ

	c1														
	1250			5000			15000			75000			240000		
L	A0	A1	A2	A0	A1	A2	A0	A1	A2	A0	A1	A2	A0	A1	A2
5	1.35	0.48	0.52	0.92	0.45	0.38	0.63	0.84	0.61	1.33	0.94	0.59	4.25	0.82	0.42
10	1.52	0.52	0.48	0.90	0.45	0.38	0.60	0.83	0.60	1.09	0.91	0.59	3.13	0.82	0.44
20	1.21	0.48	0.48	0.83	0.43	0.41	0.57	0.81	0.58	0.97	0.90	0.62	2.19	0.80	0.44
40	1.00	0.52	0.52	0.81	0.45	0.41	0.57	0.78	0.58	0.90	0.88	0.56	1.75	0.79	0.44

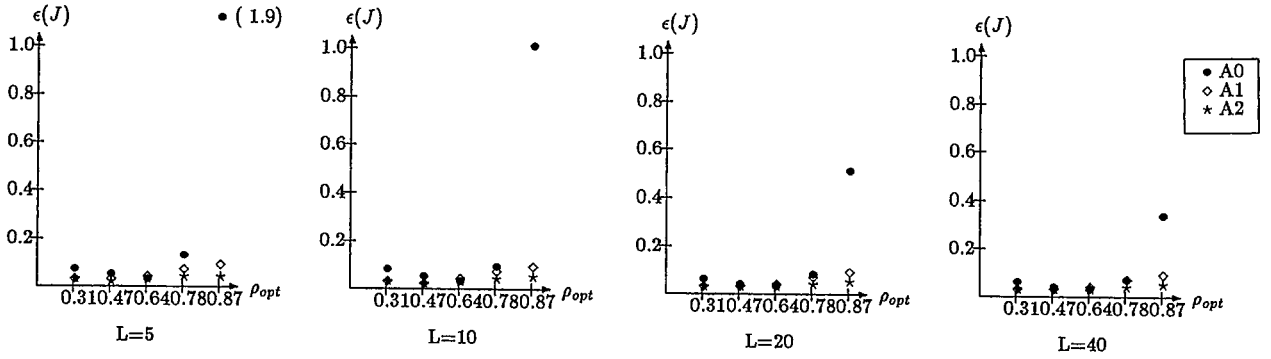
Std Dev %error in θ

	c1														
	1250			5000			15000			75000			240000		
L	A0	A1	A2	A0	A1	A2	A0	A1	A2	A0	A1	A2	A0	A1	A2
5	0.03	0.01	0.01	0.01	0.00	0.00	0.01	0.01	0.01	0.07	0.03	0.01	1.05	0.05	0.01
10	0.04	0.01	0.01	0.01	0.00	0.00	0.01	0.01	0.01	0.04	0.03	0.01	0.60	0.05	0.02
20	0.03	0.01	0.01	0.01	0.00	0.00	0.01	0.01	0.01	0.04	0.03	0.01	0.31	0.05	0.02
40	0.03	0.01	0.01	0.02	0.00	0.00	0.01	0.01	0.01	0.03	0.03	0.01	0.21	0.05	0.02

Mean %error in J

	c1														
	1250			5000			15000			75000			240000		
L	A0	A1	A2	A0	A1	A2	A0	A1	A2	A0	A1	A2	A0	A1	A2
5	0.04	0.00	0.00	0.02	0.01	0.00	0.01	0.02	0.01	0.09	0.04	0.02	1.60	0.05	0.02
10	0.04	0.00	0.00	0.02	0.00	0.00	0.01	0.02	0.01	0.05	0.04	0.02	0.79	0.05	0.02
20	0.03	0.00	0.00	0.02	0.01	0.01	0.01	0.02	0.01	0.04	0.04	0.02	0.38	0.05	0.02
40	0.03	0.00	0.00	0.02	0.01	0.01	0.01	0.02	0.01	0.04	0.05	0.01	0.24	0.05	0.02

Std Dev %error in J



Graph 2: M/D/1 Test Case

M/M/1 2-D Test Case. To test the algorithm in higher dimensions,we used the M/M/1 queue with the following objective function having two controllable parameters:

$$J(\theta) = J(x,\lambda) = E[L((x,\lambda),\omega)] = E[T + c_1/\rho + c_2\lambda],$$

where T = system time,
ρ = utilization of the server,
x = mean service time,
λ = arrival rate,
θ = (x,λ) is the controllable (vector) parameter.

We wish to find θ=(x,λ) to minimize J, subject to x≥0 and 0<λx<1. Using ρ=λx, T(θ,ω) is again the only stochastic quantity in the performance measure. Thus, we have

$$J(x, \lambda) = E[T] + c_1/(\lambda x) + c_2\lambda .$$

Again, we wish to find θ=(x,λ) to minimize J subject to x≥0 and 0<λx<1. The algorithms are extended to multi-dimensions in the natural way, with θ and g now vectors. (NB: the coefficients are still scalars in our implementation; more generally, they can be extended to matrices.) The only other change in the algorithms is reflected in the new objective function J, i.e., the gradient becomes

$$\frac{\partial J}{\partial \theta} = \begin{bmatrix} \frac{\partial E[T]}{\partial x} \\ \frac{\partial E[T]}{\partial \lambda} \end{bmatrix} - c_1 \begin{bmatrix} \frac{1}{\lambda x^2} \\ \frac{1}{\lambda^2 x} \end{bmatrix} + c_2 \begin{bmatrix} 0 \\ 1 \end{bmatrix},$$

and the Hessian matrix $d^2J/d\theta^2$, necessary for A2 and A3, is

$$\frac{d^2J}{d\theta^2} = \frac{\partial^2 E[T]}{\partial \theta^2} + \begin{bmatrix} \frac{2}{\lambda x^3} & \frac{1}{\lambda^2 x^2} \\ \frac{1}{\lambda^2 x^2} & \frac{2}{\lambda^3 x} \end{bmatrix}.$$

As before in the scalar optimization, IPA is used to estimate the $\partial E[T]$ and $\partial^2 E[T]$ quantites (now vectors and matrices, respecively) using the procedure outlined in Section III, yielding the derivative estimates $g_{PA}(\theta_n,\omega)$ and $g'_{PA*}(\theta_n,\omega)$. Note that the advantages of PA over finite difference estimates become even more evident in higher dimensions, because to calculate the gradient vector and Hessian matrix in k dimensions still requires only one sample path in PA, whereas the finite difference method requires 2k sample paths for the gradient and up to $2k^2+1$ sample paths for the Hessian matrix.

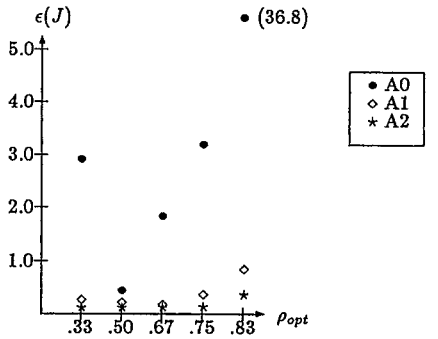
Using the analytic formula for E[T] as before, we find the true minimum occurs at

$$\lambda = \sqrt[3]{\frac{c_1}{c_2^2}}, \quad x = \frac{1}{\sqrt{\frac{\lambda}{c_1}} + \lambda} .$$

For these set of runs we fixed the number of customers/iteration at L=5, took as starting values λ_0=0.5, x_0=1.0, and studied the following five cases:

c_1	1/4	1	8	27	125
c_2	1/2	1	8	27	125
λ_{opt}	1	1	1/2	1/3	1/5
x_{opt}	1/3	1/2	4/3	9/4	25/6
ρ_{opt}	0.33	0.50	0.67	0.75	0.83
run length	10,000	20,000	30,000	50,000	100,000

The calculated mean and standard deviation % error for x, λ and J are given in Table 3, and ε(J) is displayed graphically in Graph 3. A2 did substantially better than A1, which did substantially better than A0, in all cases for all "goodness" measures except sd % sd error in x.



Graph 3: M/M/1 2-D Test Case

Table 3: M/M/1 2-D Test Case Simulation Results: %error (mean/sd)

	c_1/c_2														
	0.25/0.50			1/1			8/8			27/27			125/125		
	A0	A1	A2	A0	A1	A2	A0	A1	A2	A0	A1	A2	A0	A1	A2
x	29.0	6.50	0.50	9.60	5.40	0.20	3.48	2.35	0.17	12.3	9.91	1.47	31.6	20.1	11.0
	9.00	1.80	1.50	4.20	1.60	1.00	1.58	1.80	1.20	1.02	2.27	2.36	0.70	3.41	4.61
λ	28.6	7.80	0.70	10.9	6.70	0.60	2.60	2.80	0.20	15.2	12.2	1.40	50.5	27.5	13.0
	6.70	1.80	1.60	4.20	1.60	1.00	5.60	1.60	0.80	9.90	2.40	2.70	20.0	6.00	6.50
J	2.60	0.16	0.01	0.31	0.11	0.01	0.61	0.07	0.02	2.04	0.24	0.03	19.0	0.69	0.22
	1.09	0.06	0.01	0.16	0.05	0.01	1.62	0.05	0.02	2.31	0.12	0.03	31.5	0.30	0.16

M/U/1 Test Case. Our fourth test case, also a two-dimensional optimization problem, utilized the M/U/1 queue with the following objective function:

$$J(\theta) = J(\theta_1,\theta_2) = E[L((\theta_1,\theta_2),\omega)]$$
$$= E[T - c_1\theta_1 - c_2\theta_2] = E[T] - c_1\theta_1 - c_2\theta_2,$$

where θ_1= mean service time,

θ_2= "half-width" of the distribution,

c_1 = penalty cost on server speed (≥ 0),

c_2 = penalty cost on "uniformity" of the server (≥ 0),

i.e., the service time is distributed uniformly on $[\theta_1 - \theta_2, \theta_1 + \theta_2]$. We wish to find $\theta=(\theta_1, \theta_2)$ to minimize J, subject to $0 \leq \theta_1 \leq \theta_2$ and $0 < \lambda\theta_1 < 1$. The gradient and the Hessian are given by

$$\frac{dJ}{d\theta} = \begin{bmatrix} \frac{\partial E[T]}{\partial \theta_1} \\ \frac{\partial E[T]}{\partial \theta_2} \end{bmatrix} - c_1\begin{bmatrix}1\\0\end{bmatrix} - c_2\begin{bmatrix}0\\1\end{bmatrix}, \quad \frac{d^2J}{d\theta^2} = \frac{d^2E[T]}{d\theta^2}.$$

For the M/U/1, the P-K formula yields

$$E[T] = \theta_1 + \frac{\lambda(\theta_1^2 + \theta_2^2/3)}{2(1-\lambda\theta_1)}.$$

With our objective function and the given constraints, it turns out that for some values of c_1 and c_2, the theoretical optimal solution lies on one of the constraint boundaries, i.e., $dJ/d\theta \neq 0$ for these cases. We avoided these cases in our experiments, because the algorithms will not find these points at which $dJ/d\theta$ is non-zero. A solution at which $dJ/d\theta=0$ exists if

$$c_1 > 6c_2^2 + 3c_2 + 1,$$

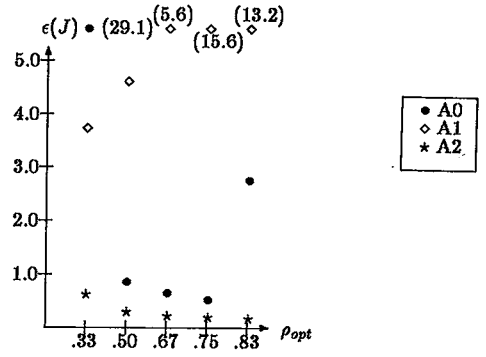
and is given by

$$\theta_1 = \frac{1}{\lambda}\left(1 - \frac{1}{\sqrt{a}}\right), \quad \theta_2 = \frac{3c_2}{\lambda}\left(\frac{1}{\sqrt{a}}\right),$$
$$\text{where } a = 3c_2^2 - 2c_1 + 1.$$

For these set of runs we fixed the number of customers/iteration at L=5 again, took as starting values θ_1=50, θ_2=20, and studied the following five cases:

c_1	1.5	2.5	5.0	14.0	35.0
c_2	0.07	0.2	0.4	0.8	1.5
$\theta_{1\ opt}$	29.0	49.2	65.7	80.0	87.3
$\theta_{2\ opt}$	14.9	30.5	41.1	47.9	57.0
ρ_{opt}	0.29	0.49	0.66	0.80	0.87
run length	10,000	20,000	30,000	50,000	100,000

The calculated mean and standard deviation % error for θ_1, θ_2, and J are given in Table 4, and $\epsilon(J)$ is displayed graphically in Graph 4, from which we observe the following: (i) A2 was the most accurate in all cases, but surprisingly (ii) A1 did very poorly in the high ρ_{opt} cases. We conjecture that in these cases A1 is averaging gradient estimates very different from the true gradient, i.e., in some sense using old data which is "bad." A2 apparently does not suffer this fate, because it uses the second derivative to update this "bad" data into useful data.



Graph 4: M/U/1 Test Case

Table 4: M/U/1 Test Case Simulation Results: %error (mean/sd)

	c_1/c_2														
	0.25/0.50			1/1			8/8			27/27			125/125		
	A0	A1	A2	A0	A1	A2	A0	A1	A2	A0	A1	A2	A0	A1	A2
x	40.5	4.00	2.09	1.68	1.87	0.58	0.90	4.20	0.59	0.63	4.05	0.16	0.38	4.27	0.26
	8.29	21.6	20.3	6.59	14.8	10.3	7.06	15.1	9.53	9.27	14.6	10.9	6.24	10.5	10.7
λ	29.3	11.4	0.24	28.1	3.08	9.38	34.7	3.78	6.71	25.3	5.97	5.99	13.0	10.8	0.47
	8.29	21.6	20.3	6.59	14.8	10.3	7.06	15.1	9.53	9.27	14.6	10.9	6.24	10.5	10.7
J	27.9	2.54	0.37	0.67	2.60	0.14	0.50	3.14	0.09	0.34	6.71	0.07	1.82	8.66	0.05
	8.26	2.60	0.36	0.35	3.69	0.14	0.21	4.69	0.08	0.25	14.1	0.07	1.92	10.0	0.05

5. SUMMARY AND FUTURE WORK

The idea of averaging and updating the gradient estimate used in optimizing system time in a G/G/1 queue has been investigated experimentally. The framework of SA is used as the basic optimization technique, and PA plays the prominent role as the means by which to estimate the gradient and to update it. Experimental convergence was obtained in all the cases studied -- using both single- and multi-dimensional objective functions -- with a convergence rate superior to the original algorithm studied by Suri & Leung. Further improvements in the algorithm include step-size control, expansion to matrices for a_n and b_n coefficients in the multi-dimensional version,and the determination of an appropriate stopping criterion for the algorithm. Also, analytical study of the convergence properties of the algorithm is desirable.

REFERENCES

Cao, X. R. (1985), "Convergence of Parameter Sensitivity Estimates in a Stochastic Experiment," *IEEE Trans. Automatic Control*, Vol. AC-30, pp. 845-853.

Cao, X. R. (1986), "On the Sample Performance Functions of Jackson Queueing Networks, submitted to *Operations Research.*

Fogel, E. (1981), "A Fundamental Approach to the Convergence Analysis of Least Squares Algorithms," *IEEE Trans. Automatic Control*, Vol. AC-26, pp. 646-655.

Glynn, P. W. (1986), "Optimization of Stochastic Systems," *Proc. 1986 Winter Simulation Conference*, pp. 52-59.

Gong, W. B. and Ho, Y. C. (1987), "Smoothed Perturbation Analysis of Discrete Event Dynamic Systems,"*IEEE Trans. Automatic Control,* Vol. AC-32, pp. 858-866.

Ho, Y. C. (1979), "Parameter Sensitivity of a Statistical Experiment, *IEEE Trans. Automatic Control*, Vol. AC-24, pp. 982-983.

Ho, Y. C. (1987), "Performance Evaluation and Perturbation Analysis of Discrete Systems: Perspective and Open Problems," *IEEE Trans. Automatic Control*, Vol. AC-32, pp. 563-572.

Ho, Y. C. and Cao, X. R. (1983), "Perturbation Analysis and Optimization of Queueing Networks," *J. Optim. Theory Applic.*, Vol. 40, No. 4, pp. 559-582.

Ho, Y. C. and Li, S. (1988), "Extensions of Infinitesimal Perturbation Analysis,"*IEEE Trans. Automatic Control*, Vol. AC-33, pp. 427-438.

Kesten, H. (1958), "Accelerated Stochastic Approximation," Ann. Math. Stat., Vol. 29, pp. 41-59.

Kiefer, J. and Wolfowitz, J. (1952), "Stochastic Estimation of the Maximum of a Regression Function," *Ann. Math. Stat.*, Vol. 23, pp. 462-466.

Kleinrock, L. (1975), *Queueing Systems,* John Wiley and Sons.

Kushner, H. J. and Clark, D. C. (1978), *Stochastic Approximation Methods for Constrained and Unconstrained Systems*, Springer-Verlag, New York.

Meketon, M. S. (1987), "Optimization in Simulation: A Survey of Recent Results," *Proc. 1987 Winter Simulation Conference.*

Nevelson, M. B. and Zalmanovich, R. (1976), *Stochastic Approximation and Recursive Estimation.*

Robbins, H. and Monro, S. (1951), "A Stochastic Approximation Method," *Ann. Math. Stat.*, Vol: 22, pp. 400-407.

Ruszczynski, A. and Syski, W. (1983), "Stochastic Approximation Method with Gradient Averaging for Unconstrained Problems," *IEEE Trans. Automatic Control,* Vol. AC-28, pp. 1097-1105.

Sacks, J. (1958), "Asymptotic Distribution of Stochastic Approximation Procedures," *Ann. Math. Stat.*, Vol. 29, pp. 373-405.

Suri, R. (1987), "Infinitesimal Perturbation Analysis for General Discrete Event Systems, *J. of ACM,* pp. 686-717.

Suri, R. and Leung, Y. T. (1988), "Single Run Optimization of Discrete Event Simulations - An Empirical Study Using the M/M/1 Queue," to appear in IIE Transactions.

Suri, R. and Zazanis, M. A. (1988), "Perturbation Analysis Gives Strongly Consistent Estimates for the M/G/1 Queue," *Management Science*, Vol. 34, pp. 39-64.

Zazanis, M. A. (1986b), *Statistical Properties of Perturbation Analysis Estimates for Discrete Event Systems*, Ph.D. thesis, Harvard University.

Zazanis, M. A. and Suri, R. (1984), "Comparison of Perturbation Analysis with Conventional Sensitivity Estimates for Stochastic Systems, submitted to *Operations Research.*

Zazanis, M. A. and Suri, R. (1985), "Estimating First and Second Derivatives of Response Time for G/G/1 Queues From a Single Sample Path," submitted to *Queueing Systems.*

AUTHORS' BIOGRAPHIES

MICHAEL C. FU is a graduate student working towards his Ph.D. in the Division of Applied Sciences at Harvard University. He received an S.B. and S.M. in electrical engineering and an S.B. in mathematics from MIT in 1985, and an M.S. in applied mathematics from Harvard University in 1986. His research is in the areas of perturbation analysis of queueing systems and stochastic optimization.

Michael C. Fu
Division of Applied Sciences
Harvard University
Cambridge, MA 02138
(617) 495-4478

YU-CHI "LARRY" HO is Gordon McKay Professor of Engineering and Applied Mathematics at Harvard University. He received an S.B. and S.M. in electrical engineering from MIT in 1953 and 1955 and a Ph.D. in applied mathematics from Harvard University in 1961. He was a Guggenheim Fellow, and is an IEEE Fellow and a National Academy of Engineering member. His current research interests include modelling, perturbation analysis, optimization, and control of discrete-event dynamic systems, especially with applications to manufacturing systems.

Yu-Chi Ho
Division of Applied Sciences
Harvard University
Cambridge, MA 02138
(617) 495-3992