International Journal for Computational Methods in Engineering Science and Mechanics
Publication details, including instructions for authors and subscription information:
http://www.tandfonline.com/loi/ucme20

Monte Carlo Solution to Find Input Parameters in Systems Design Problems

Hossein Arsham [a]
[a] The Carey Business School, The Johns Hopkins University , Baltimore , MD , USA
Published online: 06 May 2013.

To cite this article: Hossein Arsham (2013) Monte Carlo Solution to Find Input Parameters in Systems Design Problems, International Journal for Computational Methods in Engineering Science and Mechanics, 14:4, 343-353, DOI: 10.1080/15502287.2012.756957

To link to this article: http://dx.doi.org/10.1080/15502287.2012.756957

International Journal for Computational Methods in Engineering Science and Mechanics, 14:343–353, 2013

ISSN: 1550-2287 print / 1550-2295 online
DOI: 10.1080/15502287.2012.756957

Taylor & Francis
Taylor & Francis Group

Monte Carlo Solution to Find Input Parameters in Systems Design Problems

Hossein Arsham
The Carey Business School, The Johns Hopkins University, Baltimore, MD, USA

Most engineering system designs, such as product, process, and service design, involve a framework for arriving at a target value for a set of experiments. This paper considers a stochastic approximation algorithm for estimating the controllable input parameter within a desired accuracy, given a target value for the performance function. Two different problems, what-if and goal-seeking problems, are explained and defined in an auxiliary simulation model, which represents a local response surface model in terms of a polynomial. A method of constructing this polynomial by a single run simulation is explained. An algorithm is given to select the design parameter for the local response surface model. Finally, the mean time to failure (MTTF) of a reliability subsystem is computed and compared with its known analytical MTTF value for validation purposes.

Keywords System design and simulation, Parameter setting design, Reliability, Production design, Goal seeking problem, Discrete event simulation

1. INTRODUCTION

Simulation continues to be the primary method by which engineers obtain information about analysis of complex stochastic systems, such as production assembly lines, flexible manufacturing systems, and reliability systems. Almost all stochastic system performance evaluations can be formulated as an estimation of an expected value. Consider a system with continuous parameter

$v \in V \subseteq R$, where V is an open interval. Let

$$J(v) = E_{Y|v}[Z(Y)] \tag{1}$$

be the steady-state expected performance measure, where Y is a random vector with known probability density function (pdf), f(y;v) depends on v, and Z is the performance measure. For example, in a reliability system, J(v) might be the mean time to failure; Z is the lifetime of a system; Y is the lifetime of the components; and v might be the components' mean lifetimes. In general, v is the shape or scale parameter of the underlying pdf.

In systems analysis, we resort to simulation when Z is either unknown or is too complicated to calculate analytically. Before proceeding further, we distinguish between discrete event static systems (DESS) and discrete event dynamic systems (DEDS). Dynamic systems evolve over time; static systems do not evolve over time. Examples of dynamic systems are the queueing systems; examples of static systems are reliability systems. Note that while in DESS Y is a multidimensional vector, in DEDS Y represents a stochastic process.

Simulation is needed to estimate J(v) for most DESS and DEDS. The principal strength of simulation is its flexibility as a systems analysis tool for highly complex systems.

In discrete event systems, Monte Carlo simulation is usually needed to estimate J(v) for a given value $v = v_0$. By the law of large numbers

$$\hat{J}(v_0) = 1/n \sum_{i=1}^{n} Z(y_i), \tag{2}$$

converges to the true value, where y_i, $i = 1, 2, \ldots, n$ are independent, identically distributed random vector realizations of Y from $f(y; v_0)$, and n is the number of independent replications. The numerical result based on (2) is only a point estimate for J(v) at $v = v_0$. The numerical result based on (2) is a solution to a *system analysis*: Given the underlying pdf with a particular parameter value v_0, estimate the output function $J(v_0)$. The direct problem is widely used in stochastic systems analysis. Now we pose the *system design problem*: Given a target output value of the system and a parameterized pdf family, find an input value for the parameter which generates such an output. The solution to the design problem has potential application in stochastic systems analysis and design. Mathematical formulation of the

The author is most appreciative to the three reviewers for their careful readings, useful comments, and suggestions that are incorporated in the final version. This work is supported by The National Science Foundation Grant CCR-9505732.

Address correspondence to Hossein Arsham, The Carey Business School, The Johns Hopkins University, 100 International Drive, Baltimore, MD 21202, USA. E-mail: harsham@ubalt.edu

design problem is as follows:

Given τ, find $v \in V \subseteq R$ subject to $J(v) = \tau$, where

$$J(v) = E_{Y|v}[Z(Y)] = \int Z(y) f(y; v)\, dy, \tag{3}$$

Z: $R^m \to R$ is a system performance measure
Y $\in R^m$ is a random vector (or a truncated stochastic process) with pdf f (y; v)

The design problem is essentially backwards. The output is given, but the input must be determined. This is easiest to appreciate when a designer wants to match experimental data in order to obtain some basic parameters. The designer simulates the process numerically and obtains an approximation for that same output. The goal is to match the numerical and experimental results as closely as possible by varying the values of input parameters in the numerical simulation. Analyzing this, clearly the output is there, and it is the input quantity that needs to be determined. The most obvious difficulty in solving the design problem is that one cannot simply calculate a straightforward solution and be done. Since the output must be set by varying the input, an iterative method of solution is implied. In the case when v is any controllable or uncontrollable parameter, the designer is interested in estimating J(v) for a *small* change in $v = v_0$ to $v = v_0 + \delta v_0$. This is the so-called what-if problem, which is a direct problem. However, when v is a controllable input, the decision maker may be interested in the goal-seeking problem; i.e., "What value of input parameter v will achieve a desired c the output value $J(v) = J_0$?" While the what-if problem has been extensively studied, the goal-seeking simulation problem is relatively new. Design interpolation based on regression models provides an indirect approach to solve the design problem. In this treatment, one simulates the system for many different values of $v = v_0$, and then one approximates the response surface function J(v). Finally, the fitted function is used to interpolate to obtain the unknown parameter v. Since the shape of J(v) function is unknown, this approach is tedious, time-consuming, and costly. Moreover, in random environments, the fitted model might have unstable estimates for the coefficients. The only information available about J(v) is general in nature; for example, continuity, differentiability, invertability, and so on.

The simulation models based on (2), although simpler than the real-world system, are still a very complex way of relating input (v) to output J(v). Sometimes a simpler analytic model may be used as an auxiliary to the simulation model. This auxiliary model is often referred to as a local response surface model (known also as a metamodel [15]). Local response surface models may have different goals: model simplification and interpretation [13], optimization [2, 3, 4, 31], what-if analysis [1, 6], and generalization to models of the same type. The following polynomial model can be used as an auxiliary model.

$$J(v) = J(v_0) + \delta v J'(v_0) + (\delta v)^2 J''(v_0)/2 + \cdots, \tag{4}$$

where $\delta v = v\text{-}v_0$ and the primes denote derivatives. This local response surface model approximates J(v) for small δv. To estimate J(v) in the neighborhood of v_0 by a linear function, we need to estimate the nominal J(v) based on (2) and its first derivative. Traditionally, this derivative is estimated by crude Monte Carlo; i.e., finite difference, which requires rerunning the simulation model. Methods which yield enhanced efficiency and accuracy in estimating, at little additional cost, are of great value.

There are few ways to obtain efficiently the derivatives of the output with respect to an input parameter [4]. The most straightforward method is the Score Function (SF). The SF approach [19, 26, 27] is the major method for estimating the performance measure and its derivative, while observing *only a single* sample path from the underlying system [26]. The basic idea of SF is that the derivative of the performance function, $J'(v)$, is expressed as expectation with respect to the *same* distribution as the performance measure itself.

This paper treats the design problem as a simulation (as opposed to regression) problem. By this approach, we are able to apply variance reduction techniques (VRT) used in the direct problem. Specifically, we embed a stochastic version of Newton's method in a recursive algorithm to solve the stochastic equation $J(v) = J$ for v, given J at a nominal value v_0.

The explicit use of a linear local response surface model is the target parameter design: Given a desired value $J = J(v)$, find the prerequisite input parameter v.

Most engineering design methods essentially involve a framework for arriving at a target value for product, process, and service attributes, through a set of experiments which include Monte Carlo experiments. To solve the product design problem, we will restrict our model to the first order expansion. For a given J(v), the estimated δv using (4) is

$$\hat{\delta} v = [J(v) - \hat{J}(v_0)]/\hat{J}'(v_0), \tag{5}$$

provided that the denominator in (5) does not vanish for any v_0 in interval V.

The remainder of this article is divided into six sections. The next section contains the construction of a polynomial local response surface model using estimated derivatives of J(v). Section 3 deals with the target setting problem in design of a system. This is followed by construction of an accuracy measure in Section 4. Section 5 develops an iterative solution algorithm for the parameter selection problem. Section 6 illustrates the proposed method using a reliability system. Finally, Section 7 provides some concluding remarks and ideas for further research and extensions.

2. CONSTRUCTION OF POLYNOMIAL LOCAL RESPONSE SURFACE MODEL BY SINGLE-RUN SIMULATION

Simulation models, although simpler than real-world systems, are still very complex tools for relating input parameters (v) to performance measures J(v). Sometimes a simple analytical model may be used as an auxiliary to the simulation model. This auxiliary local response surface model is often referred to as a metamodel [15]. In this treatment, we have to simulate the system for some different values of (v) and then use a "goodness-of-fit" regression. We fit a response surface to these data. Clearly, coupling the simulation model with the Score Function method enhances the efficiency of constructing a local response surface model. A local response surface model can also be constructed by using sensitivities in a Taylor expansion of J(v) in the neighborhood of $v = v_0$. The resulting local response surface model can be used for characterization (such as increasing/decreasing, and convexity/concavity) of the response surface.

Let

$$J(v) = E_{Y|v}[Z(Y)] = \int Z(y)f(y; v)dy, \text{ where} \tag{6}$$

Z is a system performance measure,

$Y \in R^m$ is a random vector (or a truncated stochastic process) with pdf f (y; v) as the steady-state performance measure, then

$$J'(v) = \int [Z(y).f(y; v)]'dy, \tag{7}$$

where the prime (′) denotes the derivative with respect to v. Note that, despite the fact that y depends on v, only the function Z.f is subject to differentiation with respect to v. From (7) it follows that

$$J'(v) = \int Z(y)\, f'(y; v)\, dy = E_{Y|v}[Z(Y).S], \tag{8}$$

where $S = f'(y;v)/f(y;v)$ is the Score Function (in simulation literature it is also known as Likelihood Ratio). Differentiation is with respect to v. This is subject to the assumptions that the differentiation and the integration operators are interchangeable, $f'(y;v)$ exists, and $f(y;v)$ is positive for all $v \in V$, where V is an open interval. A necessary and sufficient condition for the interchangeability used above is that there must be no discontinuity in the distribution with position depending on the parameter v [4]. Similarly, the second derivative is

$$\begin{aligned} J''(v) &= \int [Z(Y)S'f(y; v) + Z(Y)Sf'(y; v)]dy \\ &= E_{Y|v}[Z(Y).H] \end{aligned} \tag{9}$$

where

$$H = S' + S^2. \tag{10}$$

In the multidimensional case, the gradient and Hessian of J(v) could be obtained in a straightforward manner by generalizing these results [6]. The estimator for the first and second derivatives based on (8) and (9) are given by:

$$\hat{J}'(v) = \sum_{i=1}^{n} Z(y_i)S(y_i;\ v)/n \tag{11}$$

$$\hat{J}''(v) = \sum_{i=1}^{n} Z(y_i)H(y_i; v)/n \tag{12}$$

where

$$S(y_i; v) = f'(y_i; v)/f(y_i; v) \tag{13}$$

and

$$H(y_i; v) = f''(y_i; v)/f(y_i; v). \tag{14}$$

Notice that both (11) and (12) estimators are evaluated at $v = v_0$, and y_i's are the same n independent replications used in (2) for estimating the nominal performance $J(v_0)$; therefore they are quite efficient in terms of computation cost. Estimates obtained by using (11) and (12) are unbiased, consistent, and they converge to the true values in the sense of the mean squared error [6]. The estimated gradient can also be used in solving optimization problems by simulation [2]. Other applications of sensitivity information include stability analysis [5].

The following subsection provides a descriptive presentation of other approaches to gradient estimations. For the full algorithmic implementations and their interrelationships; see [5] and references therein.

2.1 Derivative Estimation

In the design, analysis, and operation of Discrete Event Systems (DES), any information about the derivative dJ(v)/dv, is useful to both engineers and managers. The following approaches avoid any numerical problems associated with the finite-differencing ratio as an approximation to the derivative; they are based on *a single simulation run*, and the methods have the potential for real-time applications.

Finite difference approximation: Kiefer and Wolfowitz [18] proposed a finite difference approximation to the derivative. One version of the Kiefer-Wolfowitz (K-W) technique uses two-sided finite differences. The first fact to notice about the K-W estimate is that it requires 2N simulation runs, where N is the dimension of vector parameter v. If the decision maker is interested in gradient estimation with respect to each of the components of v, then 2N simulations must be run for each component of v. This is inefficient. The second fact is that it may have a very poor variance, and it may result in numerical calculation difficulties.

Simultaneous perturbation methods: The simultaneous perturbation stochastic approximation (SPSA) algorithm introduced by Spall [29, 30] has attracted considerable attention.

There has recently been much interest in recursive optimization algorithms that rely on measurements of only the objective function to be optimized, not requiring direct measurements of the gradient of the objective function. Such algorithms have the advantage of not requiring detailed modeling information describing the relationship between the parameters to be optimized and the objective function. For example, many systems involving complex simulations or human beings are difficult to model, and could potentially benefit from such an optimization approach. The SPSA algorithm operates in the same framework as the above Kiefer-Wolfowitz methods, but has the strong advantage of requiring a much lower number of simulation runs to obtain the same quality of result. The essential feature of SPSA, which accounts for its power and relative ease of use in difficult multivariate optimization problems, is the underlying gradient approximation that requires only TWO objective function measurements, regardless of the dimension of the optimization problem (one variation of basic SPSA uses only ONE objective function measurement per iteration). The underlying theory for SPSA shows that the N-fold savings in simulation runs per iteration (per gradient approximation) translates directly into an N-fold savings in the number of simulations to achieve a given quality of solution to the optimization problem. In other words, the K-W method and SPSA method take the same number of iterations to converge to the answer, despite the N-fold savings in objective function measurements (e.g., simulation runs) per iteration in SPSA.

SPSA can be seriously limited by, for example, the stability constraints of the system; e.g., traffic intensity must remain positive but less than one for steady-state sensitivity estimation [3].

Perturbation analysis: Perturbation analysis (PA) [10, 16] computes (roughly) what simulations would have produced, had v been changed by a "small" amount, without actually making this change. The intuitive idea behind PA is that a sample path constructed using v is frequently structurally very similar to the sample path using the perturbed v. There is a large amount of information that is the same for both of them. It is wasteful to throw this information away and to start the simulation from scratch with the perturbed v. In PA, moreover, we can let the change approach zero to obtain a derivative estimator without numerical problems. We are interested in the effect of a parameter change on the performance measure. However, we would like to realize this change by keeping the order of events exactly the same. The perturbations will be so small that only the duration, not the order, of the states will be affected. This effect should be observed in three successive stages:

Step 1: How does a change in the value of a parameter vary the sample duration related to that parameter?

Step 2: How does the change in the individual sample duration reflect itself as a change in a subsequent particular sample realization?

Step 3: Finally, what is the relationship between the variation of the sample realization and its expected value?

Harmonic analysis: Another strategy for estimating the gradient simulation is based on the frequency domain method, which differs from the time domain experiments in that the input parameters are deterministically varied in sinusoidal patterns during the simulation run, as opposed to being kept fixed as in the time domain runs. The range of possible values for each input factor should be identified. Then the values of each input factor, within its defined range, should be changed during a simulation run. In time series analysis, t is the time index. In simulation, however, t is not necessarily the simulation clock time. Rather, t is a variable of the model, which keeps track of certain statistics during each run. For example, to generate the inter-arrival times in a queueing simulation, t might be the variable that counts customer arrivals.

Frequency domain simulation experiments identify the significant terms of the polynomial that approximate the relationship between the simulation output and the inputs. Clearly, the number of simulation runs required to identify the important terms by this approach is much smaller than those of the other alternatives, and the difference becomes even more conspicuous as the number of parameters increases.

Some additional remarks on the various approaches: Using the score function (SF) method, the gradient can be estimated simultaneously, at any number of different parameter values, in a single-run simulation. The basic idea is that the gradient of the performance measure function, $J'(v)$, is expressed as an expectation with respect to the same distribution as the performance measure function itself. Therefore, the sensitivity information can be obtained with little computational (not simulation) cost, while estimating the performance measure. It is known that the crude form of the SF estimator suffers from the problem of linear growth in its variance as the simulation run increases. However, in the steady-state simulation, the variance can be controlled by run length. Furthermore, information about the variance may be incorporated into the simulation algorithm. A recent flurry of activity has attempted to improve the accuracy of SF estimates. Under regenerative conditions, the estimator can easily be modified to alleviate the problem of linear growth, yet the magnitude of the variance may be large for queueing systems with heavy traffic intensity. The heuristic idea is to treat each component of the system (e.g., each queue) separately, which synchronously assumes that individual components have "local" regenerative cycles. This approach is promising, since the estimator remains unbiased and efficient, while the global regenerative cycle is very long.

Now we look at the general (non-regenerative) case. In this case, any simulation will give a biased estimator of the gradient, as simulations are necessarily finite. If n (the length of the simulation) is large enough, this bias is negligible. However, as noted earlier, the variance of the SF sensitivity estimator increases with increase in n; thus a crude SF estimator is not even

approximately consistent. There are several ways to attack this problem. Most of the variations in an estimator come from the score function. The variation is especially high when all past inputs contribute to the performance and the scores from all are included. When one uses batch means, the variation is reduced by keeping the length of the batch small.

A second way is to reduce the variance of the score to such an extent that we can use simulations long enough to effectively eliminate the bias. This is the most promising approach. The variance may be reduced further by using the standard variance reduction techniques (VRT), such as importance sampling. Finally, we can simply use a large number of iid replications of the simulation.

PA and SF (or LR) can be unified [4]. Further comparison of the PA and SF approaches reveals several interesting differences. Both approaches require an interchange of expectation and differentiation. However, the conditions for this interchange in PA depend heavily on the nature of the problem, and must be verified for each application, which is not the case in SF. Therefore, in general, it is easier to satisfy SF unbiased conditions. PA assumes that the order of events in the perturbed path is the same as the order in the nominal path, for a small enough change in v, allowing the computation of the sensitivity of the sample performance for a particular simulation. For example, if the performance measure is the mean number of customers in a busy period, the PA estimate of the gradient with respect to any parameter is zero! The number of customers per busy period will not change if the order of events does not change.

In terms of ease of implementation, PA estimators may require considerable analytical work on the part of algorithm developer, with some "customization" for each application, whereas SF has the advantage of remaining a general definable algorithm whenever it can be applied.

Perhaps the most important criterion for comparison lies in the question of accuracy of an estimator, typically measured through its variance. If an estimator is strongly consistent, its variance is gradually reduced over time and ultimately approaches zero. The speed with which this happens may be extremely important. Since, in practice, decisions normally have to be made in a limited time, an estimator whose variance decreases fast is highly desirable. In general, when PA does provide unbiased estimators, the variance of these estimators is small. PA fully exploits the structure of DES and their state dynamics by extracting the needed information from the observed sample path, whereas SF requires no knowledge of the system other than the inputs and the outputs. Therefore, when using SF methods, variance reduction is necessary. The question is whether or not the variance can be reduced enough to make the SF estimator useful in all situations to which it can be applied. The answer is certainly yes. Using the standard variance reduction techniques can help, but the most dramatic variance reduction (VR) occurs using new methods of VR, such as conditioning, which is shown numerically to have a mean squared error that is essentially the same as that of PA.

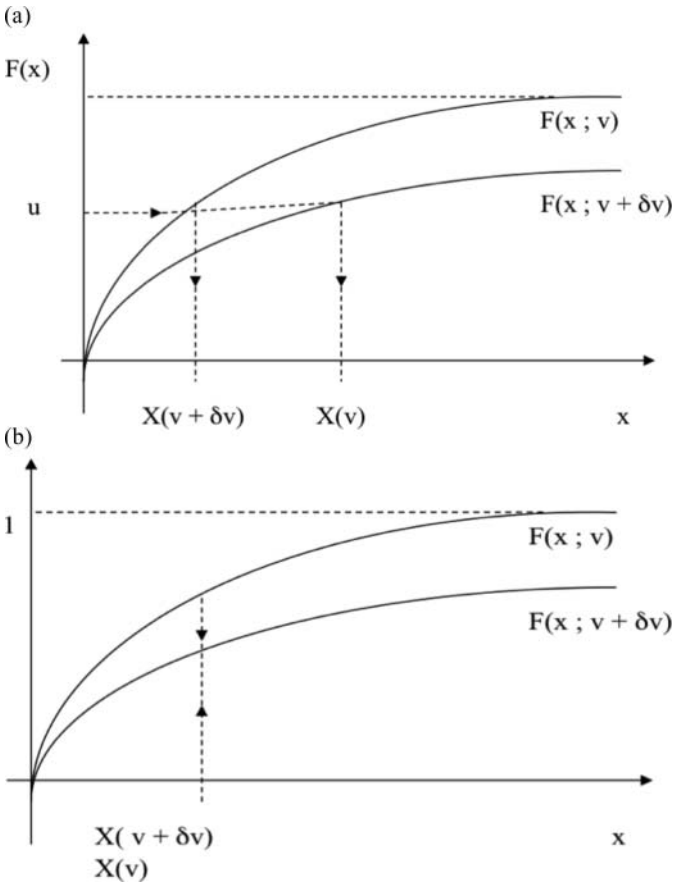

FIG. 1. (a) Parameter perturbation via perturbation analysis approach. (b) Parameter perturbation via likelihood ratio approach.

Estimating system performance for several scenarios via simulation generally requires a separate simulation run for each scenario. In some very special cases (to prevent confusion, in this paragraph we use random variable X instead of Y), such as the exponential density $f(x; v) = ve^{-vx}$, one could have obtained the perturbed estimate using Perturbation Analysis directly as follows. Clearly, one can generate random variate Y by using the following inverse transformation:

$$X_i = (1/v)\,\mathrm{Ln}\,(1/U_i)$$

where Ln is the natural logarithm and U_i is a random number distributed uniformly [0,1]. In the case of perturbed v, the counterpart realization using the same U_i is

$$X_i = [1/(v + \delta v)]\mathrm{Ln}\,(1/U_i)\,.$$

Clearly, this single-run approach is limited, *since the inverse transformation is not always available in closed form.* Figures 1a and 1b illustrate the Likelihood Ratio and Perturbation Analysis Approach, respectively. Since the Perturbation Analysis Approach has this serious limitation, we presented Score Function (Likelihood Ratio) using a single-sample path, as shown

in Figure 1b. The rationalization is that the generated random X(v) is roughly representative of X with pdf of f(x;v). However, each of these random observations could have also hypostatically come from f(x; v + δv). The factor score function weighs the gradient according this phenomenon; i.e., working in the same probability space as the nominal simulation.

3. TARGET-SETTING PROBLEM IN DESIGN

Most engineering system designs [13, 27], such as product, process, and service design, involve a framework for arriving at a target value for a set of experiments, which may include Monte Carlo experiments. A random quality loss function $L(Z_i)$ for a given system can be expanded in the neighborhood of the target value τ as follows:

$$L(Z_i) = L(\tau) + (Z_i - \tau)L'(\tau) + (Z_i - \tau)^2 L''(\tau)/2 + \cdots \quad (15)$$

It can be shown that $L(Z_i)$ converges in *mean squared error* if $| Z_i - \tau | < 1$ and derivatives are finite. Since the *optimal* loss is zero at τ, Eq. (15) reduces to the following quadratic approximation

$$L(Z_i) = K(Z_i - \tau)^2 \quad (16)$$

In (16), K is some constant that can be determined in terms of the customer's tolerance limit ($\tau - \delta$v), which suggests that the product performs unsatisfactorily when Z_i slips below this limit. Given that the cost to customer is A dollars, then $K = A/\delta v^2$. Without loss of generality, for simplicity let K = 1.

The goal of parameter design is to choose the setting of the design parameter v that minimizes the average loss (the risk function). The risk function $R(\tau)$ is the expected value of the loss function, which can be shown as:

$$R(\tau) = E\{L(Z_i)\} = (J - \tau)^2 + Var(Z_i), \quad (17)$$

This risk function measures the average loss due to a product performance that is proportional to the square of the deviation from the target value τ, as shown in Figure 2. A parabolic representation estimates the quality loss, expressed monetarily, which results when quality characteristics deviate from the target values. The cost of this deviation increases quadratically as the characteristic moves farther from the target value. The acceptance range is between J(L) and J(U). Below the lower limit, the product is rejected; above the upper limit, the product must be reworked.

The parabolic curve shown in Figure 2 represents the Taguchi loss function. From the curve, you can interpret that the amount of loss is minimum for the target (or nominal) value; as you deviate from the target, the amount of loss increases, even if you are within the specified limits of the process.

The non-adjustable variational noise; i.e.,

$$Var(Z_i|v) = Var(Z_i), \quad (18)$$

is a measure of variation among products. However, the role of product design is to reduce the $(J - \tau)^2$ part of risk, which is our interest in this paper. Note that all estimates involved in computing δv based on (5); i.e., in

$$\hat{\delta}v = [J(v) - \hat{J}(v_0)]/\hat{J}'(v_0) \quad (19)$$

are computed *simultaneously* from *a single-run simulation* of the nominal system ($v = v_0$). This was achieved by transforming all probability space to the nominal one. Note that, to estimate the derivative, we do not need to rerun the simulation. Estimating the derivatives adds only moderate computational cost to the base simulation.

4. ACCURACY OF THE ESTIMATE

In the design problem, input parameter is random, while the output is fixed and given as a target value. Upon estimating the input parameter, we must provide a measure, such as a confidence interval, to reflect the precision of the estimate. To construct a confidence interval for δv using the estimator (19), let

$$A_i = J(v) - Z(y_i; v_0), \quad (20)$$

$$B_i = Z(yi; v_0)S(y_i; v_0) \quad (21)$$

and denote

$$A = \sum A_i/n, \text{ and } B = \sum B_i/n; \quad (22)$$

Then

$$S^2 = S_{11}^2 - 2\hat{v}\, S_{12} + [\hat{v}]^2\, S_{22}, \quad (23)$$

where

$$S_{11} = \sum (A_i - A)^2 / (n - 1),$$

$$S_{22} = \sum (B_i - B)^2 / (n - 1), \quad (24)$$

and

$$S_{12} = \sum (A_i - A)(B_i - B) / (n - 1). \quad (25)$$

An exact 100% $(1 - \alpha)$ confidence interval for δv is given by

$$P\left[n^{1/2} \frac{|\delta v - v|}{S/B} \le t_{n-1,\, 1-\alpha/2}\right] \ge 1 - \alpha, \quad (26)$$

where $t_{n\text{-}1,1\,-\,\alpha/2}$ is the 100 $(1 - \alpha/2)$ percentile of Student's t distribution with (n-1) degrees of freedom [20].

5. A RECURSIVE SOLUTION ALGORITHM

The solution to the design problem is a solution of the stochastic equation J(v) = J, which we assume lies in some bounded open interval V. The problem is to solve this stochastic equation by a suitable experimental design to ensure convergence as δv approaches zero. The following is a Robbins-Monro algorithm [24], which is a root-finding procedure for functions

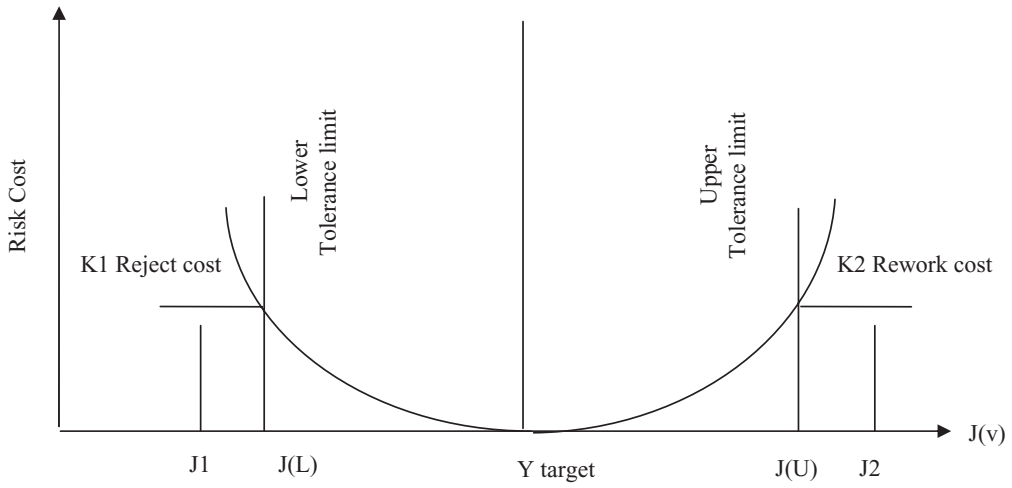

FIG. 2. Tolerance concept in target design.

whose exact values are not known but are observed with noise. It involves placing experiment j+1 according to the outcome of experiment j immediately preceding it. That is,

$$v_{j+1} = v_j + d_j[\tau - \hat{J}(v_j)]/\hat{J}'(v_j), \tag{27}$$

where d_j is any sequence of positive numbers satisfying the following conditions:

$$\sum_{j=1}^{\infty} d_j = \infty, \tag{28}$$

and

$$\sum_{j=1}^{\infty} d_j^2 < \infty, \tag{29}$$

The first condition is a necessary condition for the convergence δv to approach zero, while the second condition asymptotically dampens the effect of the simulation random errors [12, 19]. These conditions are satisfied, for example, by the harmonic sequence $d_j = 1/j$. With this choice, the rate of reduction of d_i is very high initially but may reduce to very small steps as we approach the root. Since simulation is partly statistical data generation, one performs simulation experimentation in order to generate "good" data. Instead, of classical $d_j = a/(a + j)$ with $a = 1$, we have performed some *pilot-runs* for integer values of $1 \leq a \leq 10$, and found that for $a = 9$ one gets considerable saving in number of iterations. Therefore, we have used the better choice, $d_j = 9/(9 + j)$, for the application presented in the latter section. However, as always, one must be careful in generalizing any results, since we have used specific applications.

Pilot-Runs: To estimate by simulation, the number of simulation runs (n) is critical. The confidence level of simulation output drawn from a set of simulation runs depends on the size of data set. The larger the number of runs, the higher is the associated confidence. However, more simulation runs also require more effort and resources for large systems. Thus, the main goal must be in finding the smallest number of simulation runs that will provide the desirable confidence. Since the needed statistics for number of simulation runs is not available from existing database, a pilot simulation is needed.

For large enough pilot-runs (n), say over 30 (to invoke the Central Limit Theorem), the simplest number of runs determinate is:

$$[(Z_{\alpha/2})^2\ S^2]/\Delta_1^2$$

where Δ_1 is the desirable absolute error, which is the half-length of the confidence interval with $100(1 - \alpha)\%$ confidence interval, where Z is the critical value of a standard normal distribution. S^2 is the variance obtained from the pilot-run. One may use the following sample size determinate for a desirable relative error Δ_2 in%, which requires an estimate of the coefficient of variation (C.V. in%) from a pilot-run with n over 30:

$$[(Z_{\alpha/2})^2\ (\text{C.V.})^2]/\Delta_2^2$$

The aim of applying either is to improve your pilot estimates at feasible costs.

Usually, when modelers choose a DES approach, they often model the system as an open loop or nearly open loop system, making the system behave as if there where no superior agent controlling the whole production/service/ process. Closing the loops, as shown in Figure 3, should be an elemental task that a simulation modeler should take care of, even if the scope does not involve doing it. There must be awareness of system behavior, particularly if it is known that the system is under human decisionmaking processes/activities.

The inverse simulation algorithm is based on an iterative method using differentiation and a feedback structure as shown by Figure 3.

Since the adjustments are made in proportion to the recent value, we must be sure that the results remain finite. This requires that $J'(v)$ does not vanish for $v \in V$, where V is an open interval.

To prevent excessive over-correction, we assume further that the solution lies in some finite interval V. Under these not unreasonable conditions, this algorithm will converge in mean square; moreover, it is an almost sure convergence. For some generalizations and studies concerning speed of convergence and acceleration techniques, see [14]. Finally, as in Newton's root-finding method [11, 12], it is impossible to assert that the method converges for just any initial $v = v_0$, even though $J'(v)$ may satisfy the Lipschitz condition over V. The function f(x) satisfies the Lipschitz condition on [a, b] for finite realnumbers a, and b, if for some real constant L, and for all x, y ∈ [a, b],

$$|f(x) - f(y)| \leq L|x - y|$$

Intuitively, a Lipschitz continuous function is limited in how fast it can change: for every pair of points on the graph of this function, the absolute value of the slope of the line connecting them is no greater than a definite real number; this bound is called the function's "Lipschitz constant." For example, $J(v) = v^{1/3}$ with an initial guess of $x = 1$. These numbers are growing (in absolute value) instead of converging. In fact, we have

$$v_n = (-1)^n\, 2^{n-1}$$

Hence the sequence fails to converge. However, it is clear that there is a root at $v = 0$. Notice that at $v = 0$ the derivative is undefined. Although is it Lipschitz continuous, the derivative is unbounded at the origin; see also Pintér [23].

ALGORITHM

Step 0: INPUTS
τ = Desired output
j = Iteration number
v_j = Controllable input parameter v
n = Sample size
U = Desired upper limit for absolute increment
$u = v_{j+1} - v_j$
α = A desired significance level

Step 1: INITIALIZATION
Set $j = 1$
Set $v_j = v_0$

Step 2: ESTIMATIONS
$J(v_j)$ using (2)
$J'(v_j)$ using (9)

Step 3: COMPUTATIONS
$u = 9[\tau - \hat{J}(v_j)]/[(9 + j)\, \hat{J}'(v_j)]$
If $|u| < U$
Construct $100(1 - \alpha)\%$ confidence interval for v using (20)
Stop.
Otherwise
set $v_{j+1} = v_j + u$ and $j \rightarrow j+1$

Step 4: RESET: Reset the seeds of random number generators to their initial values. Go to step 2.

Note that, by resetting the seeds to their initial values, we are using the Common Random Variate [2, 26] approach as a variance reduction technique.

6. DESIGN OF A RELIABILITY SUBSYSTEM

For most complex reliability systems, the performance measures, such as mean time to failure (MTTF), are not available in analytical form. We resort to Monte Carlo Simulation (MCS) to estimate MTTF function from a family of single-parameter density functions of the components' life with specific value for the parameter. The purpose of this section is to solve the design problem that deals with the calculation of the components' life parameters (such as MTTF) of a homogeneous subsystem, given a desired target MTTF for the system. A stochastic approximation algorithm is used to estimate the necessary controllable input parameter within a desired range of accuracy. The potential effectiveness is demonstrated by simulating a reliability system with a known analytical solution.

Consider a coherent reliability sub-system consists of four homogeneous elements; i.e., manufactured by an identical process, components having independent random lifetimes Y_1, Y_2, Y_3, and Y_4, which are distributed exponentially with rates $v = v_0 = 0.5$.

The first two and the last two elements are in series, while these two series, each with two components, are in parallel, as shown in Figure 4.

The system lifetime is $Z(Y_1, Y_2, Y_3, Y_4; v_0) = \max[\min(Y_3, Y_4), \min(Y_1, Y_2)]$. It is readily seen that the theoretical expected lifetime of this system is $J(v_0) = 3/(4\, v_0)$, [7]. Now we apply our results to compute a necessary value for v to obtain a particular value for J(v), say $J(v) = 2$. For this reliability system, the underlying probability density function is:

$$f(y; v) = v^4 \exp\left(-v \sum y_i\right),\ i = 1,\ 2,\ 3,\ 4., \qquad (30)$$

The Score Function is

$$S(y) = f'(y; v)/f(y; v) = 4/v - \sum y_i,\ i = 1,\ 2,\ 3,\ 4, \qquad (31)$$

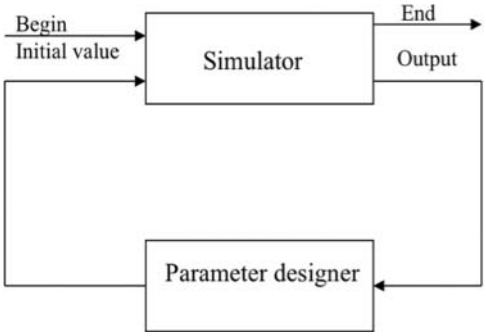

FIG. 3. System simulation with a feedback loop.

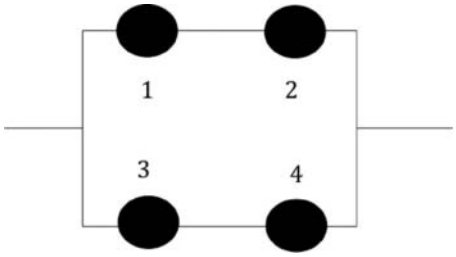

FIG. 4. A reliability subsystem.

$$H(y) = f''(y;\, v)/f(y;\, v) = \left[v^2 \left(\sum y_i\right)^2 - 8v \left(\sum y_i\right) + 12\right] \Big/ v^2, \quad i = 1,\ 2,\ 3,\ 4. \tag{32}$$

The estimated average lifetime and its derivative for the nominal system ($v = v_0 = 0.5$) based on (2) and (9) are

$$J(v_0) = \sum \max[\min(Y_{3,j}, Y_{4,j}),\ \min(Y_{1,j}, Y_{2,j})]/n, \tag{33}$$

and

$$J'(v_0) = \sum \max[\min(Y_{3,j}, Y_{4,j}),\ \min(Y_{1,j}, Y_{2,j})].\ S(Y_{i,j})/n, \tag{34}$$

$$J''(v_0) = \sum \max[\min(Y_{3,j}, Y_{4,j}),\ \min(Y_{1,j}, Y_{2,j})].\ H(Y_{i,j})/n, \tag{35}$$

respectively, where $Y_{i,j}$ is the jth observation for the ith component ($i = 1, 2, 3, 4$). We have performed a Monte Carlo experiment for this system by generating $n = 10000$ independent replications using SIMSCRIPT II.5 [9] random number streams 1 through 4 to generate exponential variates Y_1, Y_2, Y_3, Y_4, respectively, on a PC. The estimated performance is $J(0.5) = 1.5024$, with a standard error of 0.0348. The first and second derivative estimates are -3.0933 and 12.1177 with standard errors of 0.1126 and 1.3321, respectively.

A Quadratic Metamodel: The response surface approximation in the neighborhood $v = 0.5$ is:

$$J(v) = 1.5024 + (v - 0.5)(-3.0933) + (v - 0.5)^2(12.1177)/2 + \cdots \approx 6.0589v^2 - 9.1522v + 4.5638 \tag{36}$$

A numerical comparison based on direct simulation and local response surface model (36) is given in Table 1. The relative error as presented is the difference between the metamodel and the analytical values. Notice that the largest error in Table 1 is 0.33%, which could be reduced by either more accurate estimates of the derivatives and/or using a higher-order Taylor expansion. A comparison of the errors indicates that the errors are smaller and more stable in the direction of increasing v. This behavior is partly due to the fact that lifetimes are exponentially distributed with variance 1/v. Therefore, increasing v causes less variance than the nominal system (with $v = 0.50$).

Now assume that the manufacturer wants to improve the average lifetime of the system to $J(v) = \tau = 2$. To achieve this goal, we have set $v_0 = 0.5$ and $U = 0.0001$ in the proposed algorithm. The numerical results are tabulated in Table 2.

The estimated input parameter to achieve the output $J(v) = \tau = 2$ is 0.375. A 90% confidence interval based on this estimate using (20) is:

$$P[0.374 \le v \le 0.377] \ge 0.90, \tag{37}$$

Comparing the theoretical value $v_0 = 0.3750$, obtained from $J(v) = 3/(4v_0) = 2$, with our computational value suggests that the results based on the proposed algorithm are quite satisfactory. In fact, running this system with $v = 0.375$ and $n = 10000$, we obtained an estimated MTTF of $J(v) = 2.0000$. Hence the discrepancy in the estimated input parameter by this algorithm must be considered as a pure random error, which can be reduced by increasing n. The metamodel (36) could also be applied to $J(v) = 2$ to estimate the desirable v. Solving the resulting quadratic metamodel equation, the relevant root is $v = 0.3725$. This result is an inferior estimate for v compared

TABLE 1
A second-order polynomial local response surface model and direct simulation

v	Analytic	Simulation	Metamodel	Abs. error(%)
0.40	1.8750	1.8780	1.8723	0.14
0.42	1.7857	1.7885	1.7887	0.17
0.44	1.7045	1.7072	1.7098	0.31
0.46	1.6304	1.6330	1.6359	0.33
0.48	1.5625	1.5650	1.5667	0.27
0.50	***1.5000***	***1.5024***	***1.5024***	***0.16***
0.52	1.4423	1.4446	1.4430	0.05
0.54	1.3889	1.3911	1.3884	0.04
0.56	1.3393	1.3414	1.3386	0.05
0.58	1.2931	1.2951	1.2937	0.05
0.60	1.2500	1.2520	1.2537	0.30

Note: Nominal values are in italics.

TABLE 2
Iterative decision parameter estimate for the reliability system

(1) Iteration number j (2) Fixed input v_j (3) Estimated MTTF			(4) Estimated derivative (5) Change in v_j (6) New input parameter v_{j+1}		
(1)	(2)	(3)	(4)	(5)	(6)
1	0.5000	1.5024	−2.9598	−0.1513	0.349
2	0.3487	2.1544	−6.0862	−0.0208	0.369
3	0.3694	2.0333	−5.4217	+0.0046	0.374
4	0.3740	2.0083	−5.2888	+0.0011	0.375

with the iterative method, although the accuracy of the latter comes with greater computational cost.

7. CONCLUSIONS AND SOME DIRECTIONS FOR FUTURE RESEARCH

Conventional approaches to simulation involve finding the response of a system to a particular input or disturbance. Inverse simulation reverses this and attempts to find the control input required, achieving a particular response. An inverse simulation method is introduced to be used in the analysis and design of systems. The methodology is presented in the context of a reliability system application. Section 6 includes a presentation of a solution algorithm for the inverse simulation and issues of numerical stability and accuracy. The methodology includes an iterative method based on differentiation of the performance measure and use of feedback structures for generation of an inverse model based on a stochastic version of Newton's method. Almost all discrete event systems' simulation computation can be formulated as an estimation of an expected value of the system performance measure, which is a function of an input parameter of the underlying probability density function. In the ordinary system simulation, this input parameter must be known in advance to estimate the output of the system. From the designer's point of view, the input parameters can be classified as controllable and uncontrollable [6]. The influential controllable input can be recognized by factor screening methods [22]. In this paper, we considered the design problem: "What should be the controllable input parameter value to achieve a desired output value?"

As an alternative to other product design and development methods, the techniques introduced in this paper should be welcomed by the systems designers, Ulrich and Eppinger (32).

The approach used in this study was:

1. To estimate the derivative of the output function with respect to the input parameter for the nominal system by a single-run and on-line simulation;
2. To use this estimated derivative in a Taylor's expansion of the output function in the neighborhood of the parameter; and finally,
3. To use a recursive algorithm based on the Taylor's expansion to estimate the necessary controllable input parameter value within a desired accuracy.

Under some mild and reasonable conditions, the algorithm converges to the desired solution with probability 1. The efficiency of the proposed algorithm in terms of accuracy is tested using a reliability product design with satisfactory results. The approach may have major implications for simulation modelers and practitioners in terms of time and cost savings. As always, since this experiment was done on these specific numerical examples, one should be careful in making any other generalizations.

This paper introduced the general concepts of inverse simulation. An effective solution algorithm for inverse simulation is presented from first principles. The impact of the proposed inverse simulation method in conveying real understanding about the discrete event properties of the systems is now made available. The inverse simulation method is also found to be of value for the validation and control of complex discrete event simulation models with numerical stability and desirable accuracy.

The proposed inverse simulation techniques can also be applied as a measuring tool and decision procedure for the validation of simulation models. In the course of future research:

1. We expect to introduce other efficient variance reduction techniques (VRT). The Common Random Variates as a VRT are already embedded in the algorithm. Notice that since

$$\mathrm{E}[\mathrm{S}] = \mathrm{E}[\mathrm{Lnf}]' = \int [\mathrm{Lnf}]' \, \mathrm{fdx} = \int \mathrm{f}' \mathrm{dx} = \left[\int \mathrm{fdx} \right]' = 0. \tag{38}$$

We can express the gradient in terms of covariance between Z and S

$$\mathrm{J}'(\mathrm{v}) = \mathrm{Cov}[\mathrm{Z}(\mathrm{Y}), \mathrm{S}] = \mathrm{E}[\mathrm{ZS}] - \mathrm{E}[\mathrm{Z}]\,\mathrm{E}[\mathrm{S}]. \tag{39}$$

and

$$\mathrm{J}'(\mathrm{v}) = \mathrm{E}[\mathrm{Z}(\mathrm{Y})\,\mathrm{S}] + \alpha \mathrm{E}[\mathrm{S}] \tag{40}$$

where α could be the optimal linear control. Note also that (6) can be written as:

$$J'(v) = \int Z(y) f'(y; v) dy = \int Z(y)[f'(y; v)/\varphi(y; v)]\varphi(y; v) dy. \quad (41)$$

The best choice for φ is the one proportional to $Z(y)\, f'(y; v)$. This minimizes the variance of $J'(v)$; however, this optimal φ depends on the performance function $Z(y)$, which is not known in advance for most cases. One may use the empirical version of $Z(y)\, f'(y; v)$. We recommend a pilot run to study the effectiveness of these and other variance reduction techniques before implementing them.

2. We expect to extend our methodology to higher-order Taylor's expansion. We believe that there is a tradeoff between number of iterations, sample size, and the order of Taylor's expansion. Clearly, estimating the second derivative requires a larger sample size n, but fewer iterations are required to achieve the same accuracy.
3. We also expect to extend our methodology to the design problems with two or more unknown parameters by considering two or more relevant outputs to ensure uniqueness. By this generalization, we could construct a linear system of stochastic equations to be solved simultaneously by multidimensional versions of the stochastic approximation proposed in [8, 26] as well as the Newton method [28] using the second-order derivatives (e.g., Hessian).
4. The algorithms in this paper are presented in English-like, step-by-step format to facilitate implementation in a variety of operating systems and computers, thus improving portability. However, there is a need to develop an expert system that makes the algorithms more practically applicable to simulation in system design [8].

REFERENCES

1. H. Arsham, Performance Extrapolation in Discrete-event Systems Simulation, Journal of Systems Science, vol. 27, pp. 863–869, 1996.
2. H. Arsham, Stochastic Optimization of Discrete Event Systems Simulation, Microelectronics and Reliability, vol. 36, 1357–1368, 1996.
3. H. Arsham, Techniques for Monte Carlo Optimizing, Monte Carlo Methods and Applications, vol. 4, pp. 181–230, 1998.
4. H. Arsham, Gradient-based Optimization Techniques for Discrete Event Systems Simulation, in Benjamin W. Wah (ed.), Wiley Encyclopedia of Computer Science and Engineering, 1–17, 2008.
5. H. Arsham, Algorithms for Sensitivity Information in Discrete-event Systems Simulation, Simulation Practice and Theory, vol. 6(1), pp.1–22, 1998.
6. H. Arsham, A. Feuerverger, D. McLeish, J. Kreimer, and R. Rubinstein, Sensitivity Analysis and the "Wwhat-if" Problem in Simulation Analysis, Mathematical and Computer Modelling, vol. 12, pp. 193–219, 1989.
7. R. Barlow, F. Proschan, Statistical Theory of Reliability and Life Testing Probability Models, Holt Rinehart and Winston, New York, 1975.
8. A. Benveniste, M. Metivier, P. Priouret, Adaptive Algorithms and Stochastic Approximations, Springer-Verlag, New York, 1990.
9. CACI, PC Simscript II.5: Introduction and User's Manual, CACI, San Diego, California, 1987. Available at URL: http://www.simscript.com.
10. X.-R. Cao, Perturbation Analysis of Discrete Event Systems: Concepts, Algorithms, and Applications, European Journal of Operational Research, vol. 91, pp. 1–13, 1996.
11. C. Cassandras, Discrete Event Systems: Modeling and Performance Analysis, Irwin, Boston, MA, 1993.
12. H. Chen, B. Schmeiser, Stochastic Root Finding via Retrospective Approximation, IIE Transactions, vol. 33, pp. 259–275, 2001.
13. D. Clark, Necessary and Sufficient Conditions for the Robbins-Monro Method, Stochastic Processes and their Applications, vol. 17, pp. 359–367, 1984.
14. J. Clymer, System Design and Evaluation Using Discrete Event Simulation with AI, European Journal of Operational Research, vol. 84, pp. 213–225, 1995.
15. J. Dippon, J. Renz, Weighted Means in Stochastic Approximation of Minima, SIAM Journal of Control and Optimization, vol. 35, pp. 1811–1827, 1997.
16. L. Friedman, The Simulation Metamodel, Kluwer Academic Publishers, Norwell, MA, 1996.
17. M. Fu, Optimization for Simulation: Theory vs. Practice, INFORMS Journal on Computing, vol. 14, pp. 192–227, 2002.
18. P. Glynn, Likelihood Ratio Derivative Estimation for Stochastic Systems, Communications of the ACM, vol. 33, pp. 75–84, 1990.
19. J. Kiefer, J. Wolfowitz, Stochastic Estimation of the Maximum of a Regression Function, Annals of Mathematical Statistics, vol. 23, pp. 462–466, 1952.
20. J. Kleijnen, R. Rubinstein, Optimization and Sensitivity Analysis of Computer Simulation Models by Score Function Method, European Journal of Operational Research, vol. 88, pp. 413–427, 1996.
21. J. Kleijnen, W. Van Groenendaal, Simulation: A Statistical Perspective, Wiley, Chichester, UK, 1992.
22. P. L'Ecuyer, On the Interchange of Derivative and Expectation for Likelihood Derivative Estimation, Management Science, vol. 41, pp. 738–748, 1995.
23. D. Morrice, I. Bardhan, A Weighted Least Squares Approach To Computer Simulation Factor Screening, Operations Research, vol. 43, pp. 792–806, 1995.
24. J. Pintér, Global Optimization in Action: Continuous and Lipschitz Optimization: Algorithms, Implementations and Applications, Springer, New York. 2010.
25. H. Robbins, S. Monro, A Stochastic Approximation Method, Annals of Mathematical Statistics, vol. 22, pp. 400–407, 1951.
26. Ph. Ross, Taguchi Techniques for Quality Engineering, McGraw Hill, New York, 1996.
27. R. Rubinstein, B. Melamed, Modern Simulation and Modeling, Wiley, New York, 1998.
28. R. Rubinstein, A. Shapiro, Discrete Event Systems: Sensitivity Analysis and Stochastic Optimization by the Score Function Method, Wiley, New York, 1998.
29. D. Ruppert, A Newton-Raphson Version of the Multivariate Robbins-Monro Procedure, Annals of Statistics, vol. 13, pp. 236–245, 1985.
30. J. Spall, Introduction to Stochastic Search and Optimization: Estimation, Simulation, and Control, John Wiley & Sons, New York. 2003.
31. J. Spall, Adaptive Stochastic Approximation by the Simultaneous Perturbation Method, IEEE Transactions on Automatic Control, vol. 45, pp. 1839–1853, 2000.
32. S. Yakowitz, P. L'Ecuyer, F. Vazquez-Abad, Global Stochastic Optimization with Low-dispersion Point Sets, Operations Research, vol. 48, pp. 939–950, 2000.
33. E. Ulrich, S. Eppinger, Product Design and Development, McGraw-Hill/Irwin, New York, 2011.