Robust Parameter Design Optimization of Simulation Experiments Using Stochastic Perturbation Methods

ANA K. MIRANDA and ENRIQUE DEL CASTILLO*

Dept. of Industrial and Manufacturing Engineering,

The Pennsylvania State University, University Park, PA 16802, USA

August 25, 2009

Abstract

Stochastic perturbation methods can be applied to problems for which either the objective function is represented analytically, or the objective function is the result of a simulation experiment. The Simultaneous Perturbation Stochastic Approximation (SPSA) method has the advantage over similar methods of requiring only 2 measurements at each iteration of the search. This feature makes SPSA attractive for robust parameter design problems where some factors affect the variance of the response(s) of interest. In this paper, the feasibility of SPSA as a robust parameter design optimizer is presented, first when the objective function is known, and then when the objective function is estimated by means of a discrete-event simulation.

Keywords: Simulation Optimization, Noise Factors, Crossed Arrays, Non-homogeneous variance.

Introduction

The Simultaneous Perturbation Stochastic Approximation (SPSA) method, proposed by Spall (1992), is a method for optimization of multivariate stochastic systems. It

*To whom correspondence should be addressed.

only needs 2 measurements for the estimation of the gradient at each iteration, regardless of the dimension of the problem p. This property makes SPSA suitable for high dimensional problems, since the total number of loss function evaluations required to find the optimal settings $\boldsymbol{\theta}^*$ is considerably reduced compared to similar methods. This paper presents an extension of the basic SPSA algorithm to the problem where the variance of the error depends (perhaps in an unknown way) on some known factors, the so-called noise "factors" in the Design of Experiments literature (Del Castillo, 2007; Myers and Montgomery, 2002). The goal is to find a solution in the space of controllable factors that is not sensitive with respect to variability in the noise factors, which are usually uncontrollable.

The rest of this paper is organized as follows. The next section presents an overview of the basic SPSA algorithm followed by a review of other approaches to robust parameter design (RPD) in simulation optimization. Next, we study the behavior of the basic SPSA algorithm when the variance function of the errors ε in $y(\boldsymbol{x}) = L(\boldsymbol{s}) + \varepsilon(\boldsymbol{x})$ depends on the controllable factors $\boldsymbol{x}$. This is related to the RPD problem, since factors that affect $\text{Var}(\varepsilon)$ can be thought of as noise factors. The main result of our paper is presented next, where a modification of SPSA for RPD of simulated systems is discussed. Finally, the performance of the proposed methods is illustrated with two example applications: a single stage simulated inventory system and a more realistic simulated manufacturing system. The paper ends with conclusions and directions for further research.

SPSA and Robust Parameter Design

We first review the basic SPSA algorithm followed by a review of RPD approaches in simulated systems.

The Basic SPSA Algorithm

SPSA uses the recursion $\hat{\boldsymbol{\theta}}_{k+1} = \hat{\boldsymbol{\theta}}_k - a_k \hat{\boldsymbol{g}}_k(\hat{\boldsymbol{\theta}}_k)$ where, $\hat{\boldsymbol{g}}_k(\hat{\boldsymbol{\theta}}_k)$ is the estimate of the gradient at $\hat{\boldsymbol{\theta}}_k$ calculated by randomly perturbing all the components of the $\hat{\boldsymbol{\theta}}_k$ vector, that is:

$$\hat{\boldsymbol{g}}_k(\hat{\boldsymbol{\theta}}_k) \quad = \quad \frac{y(\hat{\boldsymbol{\theta}}_k + c_k\boldsymbol{\Delta}_k) - y(\hat{\boldsymbol{\theta}}_k - c_k\boldsymbol{\Delta}_k)}{2c_k} \left[\Delta_{k1}^{-1}, \Delta_{k2}^{-1}, \ldots, \Delta_{kp}^{-1}\right]^T \tag{1}$$

where $y(\cdot)$ is the observed value of the response of interest, c_k is an SPSA parameter (to be discussed below) and $\boldsymbol{\Delta}_k = [\Delta_{k1}^{-1}, \Delta_{k2}^{-1}, \ldots, \Delta_{kp}^{-1}]^T$ is a perturbation vector. For convergence of $\hat{\boldsymbol{\theta}}_k$ to $\boldsymbol{\theta}^*$, the elements of the perturbation vector $\boldsymbol{\Delta}_k$ are required to be independent and symmetrically distributed about zero with finite inverse moments (Spall, 1992). Hutchison (2002) offers some empirical results supporting the statement that no choice of the distribution of the perturbation $\boldsymbol{\Delta}_k$ provides better performance in SPSA than the symmetric Bernoulli case at ± 1. Other valid distributions are a split uniform, an inverse split uniform or a symmetric double triangular (Spall, 2003). The fastest possible stochastic rate at which the error $\hat{\boldsymbol{\theta}}_k - \boldsymbol{\theta}^*$ goes to zero is proportional to $1/k^{1/3}$ for a large number of iterations k (Spall, 2003). This rate of convergence follows from the asymptotic distribution of the estimates, which is $k^{\beta/2}(\hat{\boldsymbol{\theta}}_k - \boldsymbol{\theta}^*) \quad \underrightarrow{dist.} \quad N(\mu_{SD}, \Sigma_{SD}) \;\; as \; k \to \infty$ (Spall, 1992). Chin (1997) indicates that SPSA yields the smallest optimal mean square error (MSE) among similar algorithms and shows that $\dfrac{number\ of\ y(\theta)\ values\ in\ SPSA}{number\ of\ y(\theta)\ values\ in\ FDSA} \to \dfrac{1}{p}$.

The basic SPSA algorithm and its extensions have been considered also for solving constrained stochastic optimization problems. For this purpose Sadegh (1997) used a projection algorithm based on SPSA. Gerencsér (1998) analyzed the SPSA method for state-dependent noise, i.e., the error or noise depends on $\hat{\boldsymbol{\theta}}_k$. This type of noise typically takes the form of a correlated process and its evolution obeys a stochastic process with memory. Both SPSA with state-dependent noise and SPSA with state-independent noise have been applied in many different areas. Rezayat (1994) added a special cause control chart for quality improvement purposes. Cupertino et al. (2006) focused on control of induction motors and Spall and Cristion (1997)

looked at the control of a wastewater treatment facility. Other areas of application are supply chain management (Schwartz et al., 2006; Wang, 2006), queuing systems (Fu and Hill, 1997) and parameter optimization (Kocsis and Szepesvári, 2006). SPSA is well suited in these types of problems because it does not require full knowledge of the system input-output relationships.

Robust Parameter Design in simulation

Although SPSA has been recognized by Beyer and Sendhoff (2007) as an efficient approach for robust parameter design optimization, there is no published literature that demonstrates this, at least not to the knowledge of the authors. There have been several attempts at solving RPD problems in simulation optimization. Ittiwattana (2002) proposed a genetic algorithm that finds application in the robust engineering design process. Wild and Pignatiello (1991) presented a strategy for designing robust systems based on cross arrays. They emphasized the usefulness of their approach in dynamic environments where uncertainty exists and the demand for reliability is high. Kleijnen et al. (2003) demonstrated the idea of robustness by simulating three different supply chain configurations. Their methodology consisted in an initial sequential bifurcation to identify the important factors, followed by a classification into controllable and environmental factors. Sohn (2002) used a Monte-Carlo simulation to find robust levels of server characteristics for an M/M/1 queue. He assumed that arrival and service rates are the noise factors, both partly random; and the response to be optimized is MSE of the traffic intensity parameter. Mayers and Benjamin (1992) use Response Surface Methods, but with the output of simulation experiments instead of physical experiments, to determine the direction of search that gives a robust manufacturing design. Approaches to simulation optimization, not necessarily for *robust* optimization have been studied by numerous authors, e.g. (Ho et al., 2000; Hurrion, 1997; Yeomans, 2002). Finally, Benjamin and Erraguntla (1995) proposed a heuristic to solve a robust design problem based on a bi-criteria formulation. SPSA, however, has not been fully explored in Robust Parameter Design problems, and in particular, not

in robust design problems in simulation optimization, which is our main goal.

SPSA under non-homogeneous variance

The efficiency of SPSA as an optimization method has been proven in a large number of simulation experiments (Spall, 1992; Chin, 1997), where only loss functions with homogeneous variance have been considered. Since we are interested in the RPD problem, where, by definition, the variability of an important response is affected by certain variables (making the variance non-homogeneous), we first conducted simulation experiments for loss functions with non-homogeneous error variance. The general form of the noisy function is given by

$$y(x_1, x_2, ..., x_k) = L(x_1, x_2, ..., x_k) + \varepsilon(x_1, x_2, ..., x_k) \tag{2}$$

where $L(x_1, x_2, ..., x_k)$ is a deterministic function and the errors $\varepsilon(x_1, x_2, ..., x_k)$ follow an i.i.d normal distribution with mean zero and variance $V(x_1, x_2, ..., x_k)$, that is $\varepsilon \sim N(0, V(x_1, x_2, ..., x_k))$.

Three different functions $L(x_1, x_2, ..., x_k)$, taken from (Del Castillo, 2007) and (Moré et al., 1981), were considered: a quadratic polynomial function, a modified quadratic polynomial function, and the Freudenstein and Roth function (Moré et al., 1981):

$$L(x_1, x_2) = (-13 + x_1 + ((5 - x_2)x_2 - 2)x_2)^2 + (-29 + x_1 + ((x_2 + 1)x_2 - 14)x_2)^2.$$

Each of these functions was tested along with a variance function ($V(x_1, x_2, ..., x_k)$), which drives the process noise. The selection of the SPSA parameters follows Spall's guidelines, but in addition the cases $c < \sigma_{max}$, and $c > \sigma_{max}$, where σ_{max} is given by the square root of the maximum value of $V(x_1, x_2, ..., x_k)$ in the region of interest and c is the first element of the positive sequence $\{c_k\}$ in eq. (1), were also tried.

The quadratic loss function $L(x_1, x_2) = 0.066x_1^2 + 0.076x_2^2 - 36.77x_1 - 23.97x_2 + 7741.8$ has one minimum $L^* = 774.53$ at $\boldsymbol{x}^* = (276.08, 157.82)$, and is tested with additive error that follows the next variance functions $V(x_1, x_2)$:

1. $V(x_1, x_2) = -0.0666x_1^2 - 0.0760x_2^2 + 36.7740x_1 + 23.9761x_2 - 500$

2. $V(x_1, x_2) = -0.0666x_1^2 - 0.0760x_2^2 + 36.7740x_1 + 23.9761x_2 + 33033$

3. $V(x_1, x_2) = -0.0666x_1^2 - 0.0760x_2^2 + 36.7740x_1 + 23.9761x_2 + 10000000$

These variance functions are the reflection of the quadratic function for $L(\boldsymbol{x})$ and differ between them only in their height determined by the intercept. This is done to force a strong trade-off between the variance and the mean of the response.

Simulations consisting of 1000 and 5000 SPSA iterations, replicated 50 times, were carried out. The collected statistics were the average and standard deviation of the distance from the final estimate ($\hat{\boldsymbol{x}}$) to the optimum point ($\boldsymbol{x}^*$), and the average difference between the estimated optimal value ($\hat{L}$) and the real optimal value (L^*).

The behavior of the SPSA search turned out to be very similar for the three variance functions. However, for different values of c, changes in behavior are evident. The performance statistics are summarized in Figure 1. From Figure 1 it can be seen that the average distance from the final estimate to the optimum ($||\hat{\boldsymbol{x}} - \boldsymbol{x}^*||$) is considerably reduced with increasing number of iterations. From this figure, it can also be seen that the average difference $\hat{L} - L^*$ goes to zero as the number of iterations increases. All these results are independent of the variance function used, that is, for the quadratic function studied here there is no evidence that the results are affected by the current noise level.

INSERT Figure 1 about here.

Consider next the modified quadratic loss function $L(x_1, x_2) = 0.066x_1^2 + 0.076x_2^2 + 0.015x_1x_2 - 36.77x_1 - 23.97x_2 + 7741.8$. This has one minimum $L^* = 1356.8$ at $\boldsymbol{x}^* = (261.22, 131.95)$. The variance function considered here only depends on x_1, and is given by

$$V(x_1) = -(x_1 - 261.2208)^2 + 11000 \tag{3}$$

This case, where the loss function has an interaction term of the form $(x_1 \cdot x_2 \cdot ... \cdot x_k)$, and the noise is function only of a subset of $(x_1, x_2, ..., x_k)$, is particularly interesting since robust parameter design problems typically have this type of structure.

Plots of the performance statistics are shown in Figure 2. From the graphs on the figure it is seen that the number of iterations does not have a significant impact on the collected statistics. As opposed to experiments with the previous function, here the statistics when 1000 iterations are performed are very close to the statistics when 5000 iterations are performed.

INSERT Figure 2 about here.

As a last example of a non-homogeneous variance function consider the Freudenstein and Roth function. This function has two minima, $L_1^* = 0$ at $\boldsymbol{x}_1^* = (5, 4)$, and $L_2^* = 48.98$ at $\boldsymbol{x}_2^* = (11.41..., -0.8968...)$. The variance function used in this case is given by

$$V(x_1, x_2) = -x_1^2 - x_2^2 + 10x_1 + 8x_2 + 339 \tag{4}$$

The results obtained for this function differ considerably from the two previous functions. From Figure 3 it can be seen that the algorithm searches in the wrong direction as c gets larger. Furthermore, it exceeds the distance between the initial point and the optimal point. The immediate consequence of this is that the difference between the estimated optimum and real optimum grows larger. This shows the importance of choosing an adequate value of the c parameter.

INSERT Figure 3 about here.

The experimental results obtained suggest that SPSA is indeed a useful tool for finding the optimal point of a function that has non-homogeneous variance, and consequently, they suggest the potential of SPSA to solve RPD problems. However, it is clear that the choice of the parameters is critical for the performance of this algorithm and further investigation about parameter selection is required. The

guidelines provided by Spall gave good results for the quadratic functions, but not for the Freudenstein and Roth function when a large value of c was used. This implies that in a high noise setting selecting the parameters of the algorithm according to the suggested guidelines might result in ineffective searches, since the algorithm might not be able to approximate the optimum. For such settings, it is imperative to consider other criteria. Adaptive versions of the SPSA algorithm (Spall, 2003) eliminate the burden of finding the "correct" parameters. However, our simulation experience (Miranda, 2008) indicate these methods fail to provide a good performance in the case of non-homogeneous variance, and hence, will not be considered any further in the remaining of this paper, where we turn to a different extension of the basic SPSA algorithm aimed for RPD optimization.

SPSA as a simulation optimization technique for RPD problems

Since SPSA uses only two estimates of the objective function itself, it is suitable for optimizing a system that can be simulated. In other words, the output of a simulated system, which typically is a performance measure, can be used as the input to the SPSA algorithm. Thus, in this section, SPSA is utilized to solve robust parameter design problems on simulated systems. The objective is to determine the operating conditions for a process so that a performance measure y is as close as possible to the desired target value and the variability around that target is minimized. For this purpose, the mean square error (MSE) of y is chosen to be the expression to minimize:

$$\widehat{\mathrm{MSE}}(y) = \widehat{Var}(y) + (\bar{y} - target)^2 \tag{5}$$

When minimizing this function, two MSE(y) estimates are required at each SPSA iteration. We can obtain these estimates based on a *crossed array* experimental design (Myers and Montgomery, 2002). In particular, let $\boldsymbol{x} = (x_1, x_2, ..., x_p)$ be the p-dimensional vector of controllable factors, $\{\boldsymbol{\Delta}_k\}$ be a vector sequence of Bernoulli-distributed perturbations, $\{c_k\}$ be a positive sequence converging to zero, and $\boldsymbol{D}$ be

an experimental design used to vary the noise factors $(z_1, z_2, ..., z_p)$. Then, the first MSE(y) measurement is obtained from the performance measure of interest at points generated when $(\hat{\boldsymbol{x}}_k + c_k\boldsymbol{\Delta}_k)$ is crossed with $\boldsymbol{D}$. The second MSE(y) measurement is obtained from the performance measure at points generated when $(\hat{\boldsymbol{x}}_k - c_k\boldsymbol{\Delta}_k)$ is crossed with $\boldsymbol{D}$. Thus, if $\boldsymbol{D}$ is $(n_2 \times p)$, one iteration of the RPD version of SPSA requires $n_2 \times 2$ simulations (runs).

Figure 4 illustrates how SPSA is implemented via a crossed array design. In the figure, x_1 and x_2 are the controllable factors, while z_1 and z_2 are the noise factors. The corner points of the larger square around $\hat{\boldsymbol{x}}_1$ form the inner array. Two of these points are selected by the SPSA random perturbations $(\hat{\boldsymbol{x}}_1 + c_1\boldsymbol{\Delta}_1)$ and $(\hat{\boldsymbol{x}}_1 - c_1\boldsymbol{\Delta}_1)$. The resulting points are then crossed with the points in the outer array (smaller squares). The two MSE(y) values are calculated at each of the smaller squares and then the direction of movement is determined following the SPSA algorithm. The crossed array design is repeated at $\hat{\boldsymbol{x}}_2$, $\hat{\boldsymbol{x}}_3$, and so on. Evidently this way of obtaining the MSE(y) from a crossed array design can be applied to any number of controllable and noise factors.

INSERT Figure 4 about here.

Examples

Example 1: a single stage inventory control system

As a first illustration of how SPSA can be used to determine robust operating conditions (with respect to noise factor variations) of a simulated system, let us suppose that in a single stage inventory system one wishes to determine the time between orders (T) and the quantity to be ordered (R) of a unique type of product, in such a way that the MSE of the total cost associated with the levels of T and R is minimized. This implies T and R are the controllable factors, and the total cost (y) is the performance measure of interest. The *target* of y is zero. Evidently, y depends

on both R and T, and is given by

$$y = h \cdot \bar{I} + \pi \cdot \bar{F} + s \cdot N \tag{6}$$

where h is the inventory carrying cost, $\bar{I}$ is the average inventory, π is the shortage cost, $\bar{F}$ is the average number of shortages, s is the cost of placing an order, and N is the total number of orders (Elsayed and Boucher, 1985). Let us assume that we wish to consider stochastic demand, a fixed ordering cost, lost sales, and a positive delivery lead time (Elsayed and Boucher, 1985). The lead time (ℓ) is modeled as a U(a, b) random variable, where we can think of a and b as the noise factors that affect R and T. These noise factors will be varied at two different levels in the outer array. Analytical solutions for the stochastic lead time/stochastic demand case are not available, but this is a well-known problem if the lead time is constant (Elsayed and Boucher, 1985).

The specific values used in the discrete event simulation model are $s = \$200$, $\pi = \$9$, and $h = 0.04$ per month. The system was simulated for one year. The quantity demanded by each customer was modeled as a discrete random variable that follows the probability mass function shown in Table 1.

INSERT Table 1 about here.

The inventory system was simulated for 1000 SPSA iterations. In each SPSA iteration the total number of simulations required was $2 \cdot 2^2$. The SPSA algorithm was started from the initial point (R_0, T_0): (400,100). One instance of the trajectory followed by SPSA is shown in Figure 5 along with the MSE(y) contours. In this example the SPSA algorithm is effective since the final estimate $(\hat{R}, \hat{T})$ lies in the region of true lowest MSE(y) values. Results from three other initial points (R_0, T_0): $(170, 50), (500, 25)$, and $(800, 250)$ lead to the same conclusion.

INSERT Figure 5 about here.

Example 2: Simulation optimization of several buffer sizes in a manufacturing system

As a second and more realistic example of the application of SPSA for RPD problems, consider a small manufacturing system that uses free-path vehicles of capacity one to transport parts through four manufacturing processes and one inspection process (see Figure 6). A part arrives at the "get_on" station. It is then taken to the input buffer of the first manufacturing process, where it waits until it can be processed. After that, the part visits the remaining processes in numerical sequence, with a possible waiting time in each corresponding input buffer. There is only one machine at each process which can operate one part at a time.

INSERT Figure 6 about here.

It is well known that unless any two machines finish each production cycle at precisely the same moment, they will interfere with each other and production capacity will be lost. If one machine finishes before its successor, it must wait to dispose its finished part in order to begin a new one, that is, the machine will be blocked. If the second machine finishes before its predecessor, it will be starved. Hence the objective is to find the size of the input buffers $(B_1, B_2, B_3, B_4, B_5)$ that reduces the frequency and severity of blockage and starvation, but simultaneously minimizes the MSE of the cost given by $C = L + S + J$, where L is the cost of losing production capacity, S is the cost of each input buffer spot, and J is the cost of having an idle vehicle. J is included because it is desirable that vehicles move as many parts as they can, so that starvation is not attributed to a materials handling problem. This is because when a vehicle finds the input buffer full, it has to stay still until one spot is available so that it is able to unload the part and move. In this case the $target$ of C is zero. It will be assumed that the five machines are subject to failures. The time until a failure occurs in each machine was assumed i.i.d Exp(λ_i) for $i = 1, 2, ..., 5$. The parameters λ_i take two different values each, 800 minutes and 960 minutes. Since failures are uncontrollable in practice and their rates not

known with complete certainty, the parameters λ_i were considered the noise factors of the system. The five input buffers associated with the 4 operations plus inspection $(B_1, B_2, B_3, B_4, B_5)$ are the controllable factors.

In this example, the outer array for the noise factors (D) is a 2^{5-2} fractional factorial. With this design, each SPSA iteration requires only $2 \cdot 2^{5-2} = 16$ simulation runs. Evidently a full factorial design would result in a larger number of runs required for one SPSA iteration.

Further assumptions made in the manufacturing model are that the parts arrive according to an Exp(4) distribution, the service time is assumed to be the same for all 5 machines and follows an Exp(4) distribution. The number of vehicles in the system was fixed and equal to 8. They do not accelerate or decelerate when moving around the system. Regarding the costs, it is assumed that losing production capacity is much more expensive than having larger buffers or having an idle vehicle. Hence, $L \gg J > S$. Despite S being small compared to the other costs, it is not acceptable to have very large buffers; physical space restricts their size. Therefore, in this case, two constraints were imposed over the buffers decision variables, $B_i \leq 35$ and $B_i \geq 0$. These constraints were handled by means of projections into the feasible region (Sadegh, 1997). In actual practice, the optimization of the system must be done over discrete sets given the discrete nature of parts and buffer capacities. These can be handled by a modification of the SPSA algorithm due to Hill et al., Hill et al. (2004) whose modified SPSA algorithm works for functions defined on a grid of points having integer coordinates.

Ten replicates each of 1000 SPSA iterations were conducted from 10 different initial points. The final estimate that gives the minimum average MSE(C) was the buffer sizes $(11.55, 19, 10, 11.55, 7.66)$ (first row in Figure 7). However (11.44, 25.88, 12.55, 11.33, 19.55) (second row in Figure 7) could also be a good combination because even though its average MSE(C) is higher, its standard deviation is lower. We note that some initial points were notably better than others. For instance, from the last point the algorithm ends up on the boundary of the feasible region, unable to get

away from it. Hence the importance of trying several initial points.

INSERT Figure 7 about here.

For more details about the simulation programs utilized in these examples and more extensive computational results of the SPSA algorithm see (Miranda, 2008).

Conclusions.

SPSA is an easy-to-implement optimization algorithm. It is very efficient in settings where only noisy measurements of the objective function are available because only 2 observations are required to find the direction of search. It was observed that SPSA is effective when the objective function has non-homogeneous variance. Simulation results suggest that SPSA is able to find the optimal points when some of the variables also affect the variability of the response. The latter is particularly important since it implies that SPSA is efficient for solving RPD problems.

SPSA was extended to obtain a solution of a discrete-event simulated system that is robust to noise factor variability. For this purpose the key tool was the use of a crossed array design and a MSE objective function. The MSE objective accounts for both the variance and the mean of the response of interest as a function of the controllable factors, which are then optimized using SPSA.

Two example applications were utilized to demonstrate the modified SPSA algorithm for RPD problems: a single stage inventory system for which the quality of the solutions was easy to verify, and a more realistic manufacturing system. The simulation solutions were checked graphically and compared with their analytical counterparts when possible. In general, they showed the ability of the algorithm to converge to a robust solution, that is, a solution that is not sensitive to variation in the noise factors. Nonetheless, the potential of SPSA for achieving large savings in the total number of measurements required to estimate the optimum is only realized when the crossed array structure is not overly large. Further research is hence necessary

for finding crossed array structures with a small number of runs that allow solving an RPD problem, while concurrently taking advantage of the SPSA features.

The evidence in the simulations performed in this work suggests that SPSA is indeed a useful tool for robust parameter design optimization, not only for systems modeled analytically, but also more importantly, for systems for which only simulated values of the objective function are available.

Acknowledgment- We thank Dr. David Muñoz (ITAM-México) for providing the simulation code used in the second example.

References

Benjamin P and Erraguntla M (1995). Simulation for robust system design. *Simulation* **65**:116–128.

Beyer H and Sendhoff B (2007). Robust optimization - a comprehensive survey. *Comput Method Appl M* **196**:3190–3218.

Chin D (1997). Comparative study of stochastic algorithms for system optimization based on gradient approximations. *IEEE T Syst Man Cy B* **27**.

Cupertino F, Mininno E, Naso D, and Turchiano B (2006). An experimental implementation of SPSA algorithms for induction motor adaptive control. *IEEE Mountain Workshop on Adaptive and Learning Systems* .

Del Castillo E (2007). Process Optimization A Statistical Approach. Springer.

Elsayed E A and Boucher T O (1985). Analysis and Control of Production Systems. Prentice Hall.

Fu M and Hill S (1997). Optimization of discrete event systems via simultaneous perturbation stochastic approximation. *IEEE Trans* **29**:233–243.

Gerencsér L (1998). SPSA with state-dependent noise - a tool for direct adaptive control. *Proceedings of the 37th IEEE Conference on Decision and Control* .

Hill S, Gerencsér L, and Vágó Z (2004). Stochastic approximation on discrete sets using simultaneous difference approximations. *Proceedings of the 2004 American Control Conference* .

Ho Y C, Cassandras C G, Chen C H, and Dai L (2000). Ordinal optimisation and simulation. *J Oper Res Soc* **51**:490–500.

Hurrion R (1997). An example of simulation optimisation using a neural network metamodel: finding the optimum number of kanbans in a manufacturing system. *J Oper Res Soc* **48**:1105–1112.

Hutchison D (2002). On an efficiente distribution of perturbations for simulation optimization using simultaneous perturbation stochastic approximation. *Proceedings of the IASTED International Conference Applied Modeling and Simulation* .

Ittiwattana W (2002). A method for simulation optimization with applications in robust process design and locating supply chain operations. PhD dissertation. The Ohio State University.

Kleijnen J, Bettonvil B, and Persson J (2003). Robust solutions for supply chain management: simulation, optimization, and risk analysis. Department of Information Management/Center for Economic Research (CentER), Tilburg University (UvT), http://centerkubnl/staff/kleijnen/ .

Kocsis L and Szepesvári C (2006). Universal parameter optimisation in games based on SPSA. *Mach Learn* **63**:249–286.

Mayers R J and Benjamin P (1992). Using taguchi paradigm for manufacturing system design using simulation experiments. *Comput Ind Eng* **22**:195–209.

Miranda A K (2008). Stochastic perturbation methods for robust optimization of simulation experiments. MS thesis. The Pennsylvania State University.

Moré J, Garbow B, and Hillstrom K (1981). Testing unconstrained optimization software. *ACM T Math Software* 7:17–41.

Myers R H and Montgomery D C (2002). Response Surface Methodology Process and Product Optimization Using Designed Experiments. Wiley-Interscience.

Rezayat F (1994). On the use of SPSA-based model-free controller in quality improvement. *Proceedings of the 33th Converence on Decision and Control* .

Sadegh P (1997). Constrained optimization via stochastic approximation with a simultaneous perturbation gradient approximation. *Automatica* **33**:889–892.

Schwartz J, Wang W, and Rivera D (2006). Simulation-based optimization of process control policies for inventory management in supply chains. *Automatica* **42**:1311–1320.

Sohn S Y (2002). Robust design of server capability in M/M/1 queues with both partly random arrival and service rates. *Comput Oper Res* **29**:433–440.

Spall J (1992). Multivariate stochastic approximation using a simultaneous perturbation gradient approximation. *IEEE T Automat Contr* **37**:332–341.

— (2003). Introduction to Stochastic Search and Optimization. Wiley-Interscience.

Spall J and Cristion J (1997). A neural network controller for systems with unmodeled dynamics with applications to wastewater treatment. *IEEE T Syst Man Cy B* **27**.

Wang L (2006). Augumented simultaneous perturbation stochastic approximation (Aspsa) for discrete supply chain inventory optimization problems. PhD dissertation. The Pennsylvania State University.

Wild R and Pignatiello J (1991). An experimental design strategy for designing robust systems using discrete-event simulation. *Simulation* **57**:358–368.

Yeomans J (2002). Automatic generation of efficient policy alternatives via simulation-optimization. *J Oper Res Soc* **53**:1256–1267.

Captions and Headings

Figure 1: Left: Average distance between the final estimate and the optimal point ($||\hat{\boldsymbol{x}} - \boldsymbol{x}^*||$) when the quadratic function is used; Right: Average difference between the estimated optimal value and the real optimal value ($\hat{L} - L^*$) when the quadratic function is used.

Figure 2: Left: Average distance between the final estimate and the optimal point ($||\hat{\boldsymbol{x}} - \boldsymbol{x}^*||$) when the modified quadratic function is used; Right: Average difference between the estimated optimal value and the real optimal value ($\hat{L} - L^*$) when the modified quadratic function is used

Figure 3: Left: Average distance between the final estimate and the optimal point ($||\hat{\boldsymbol{x}} - \boldsymbol{x}^*||$) when the Freudenstein and Roth function is used; Right: Average difference between the estimated optimal value and the real optimum value ($\hat{L} - L^*$) when the Freudenstein and Roth function is used

Figure 4: Crossed array designs in an SPSA search

Figure 5: SPSA search trajectory from initial point (400,100), example 1

Figure 6: Manufacturing system layout, Example 2

Figure 7: Simulation results for the manufacturing system (example 2), 1000 SPSA simulations

Table 1: PMF of quantity demanded in the single stage inventory system (example 1).

Tables and Figures

x	1	2	3	4	5
$\mathrm{P}(X = x)$	0.3	0.1	0.1	0.3	0.2

Table 1

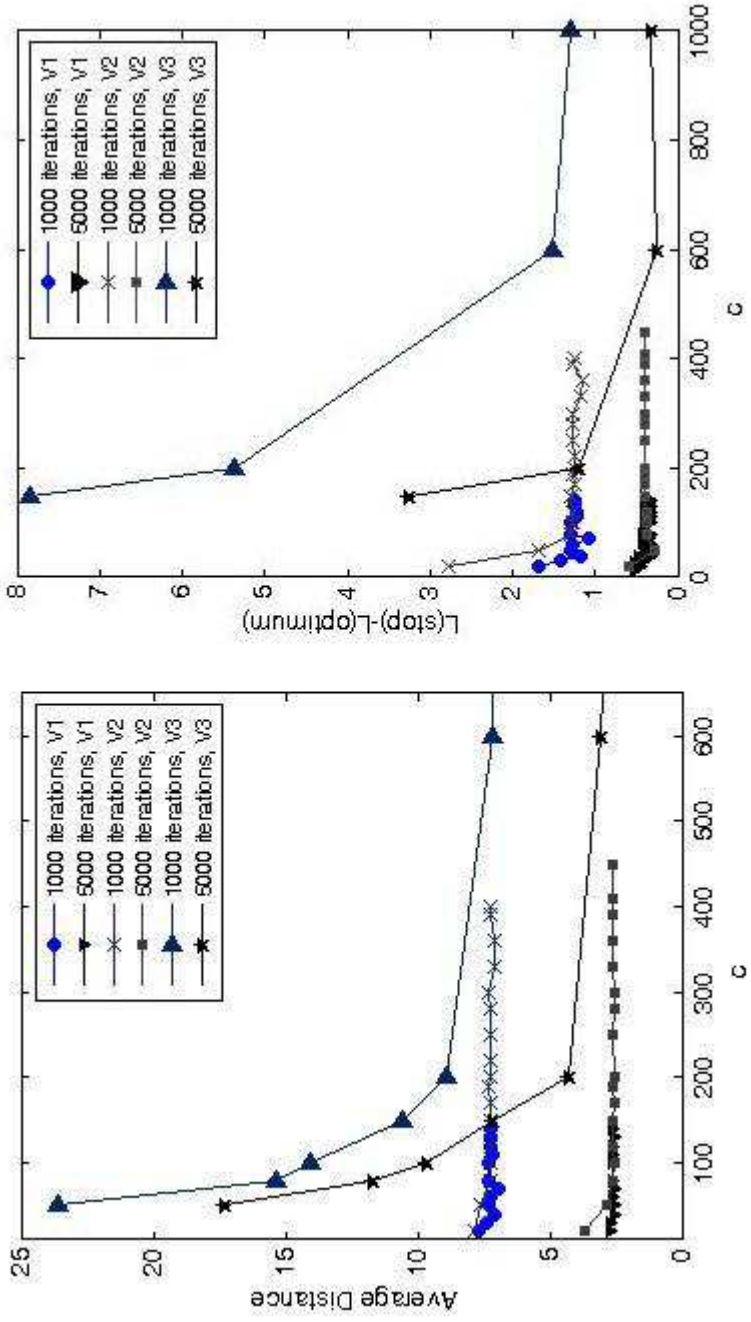
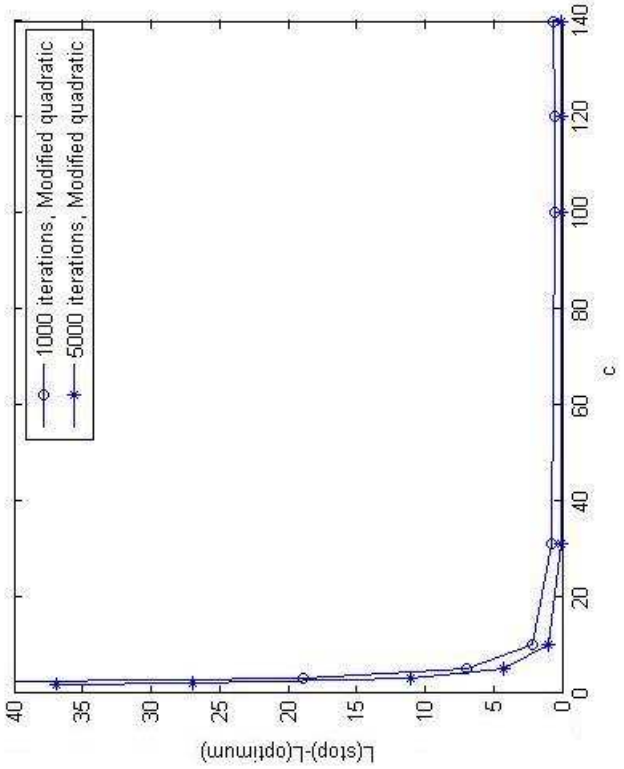

Average Distance
c
1000 iterations, V1
5000 iterations, V1
1000 iterations, V2
5000 iterations, V2
1000 iterations, V3
5000 iterations, V3
L(stop)-L(optimum)
c
1000 iterations, V1
5000 iterations, V1
1000 iterations, V2
5000 iterations, V2
1000 iterations, V3
5000 iterations, V3

Figure 1


L(stop)-L(optimum)
c
1000 iterations, Modified quadratic
5000 iterations, Modified quadratic

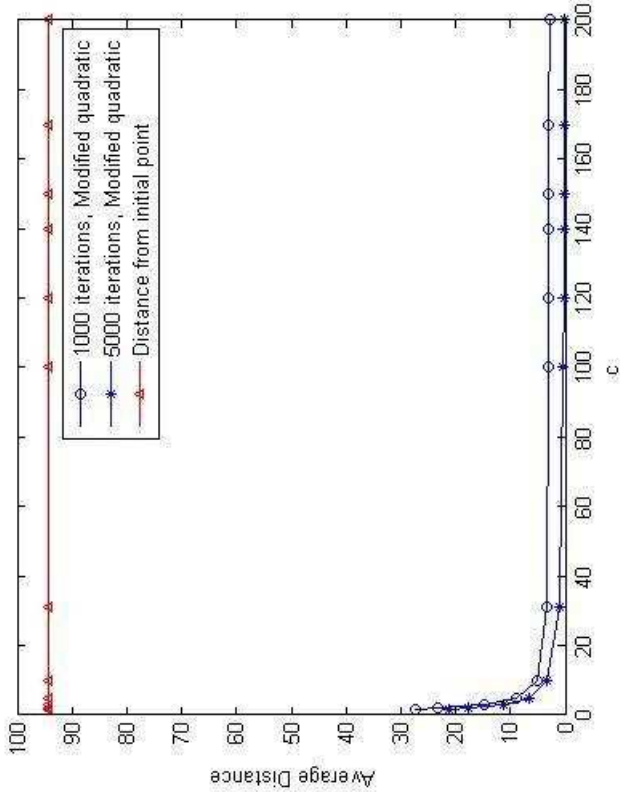

Average Distance
c
1000 iterations, Modified quadratic
5000 iterations, Modified quadratic
Distance from initial point

Figure 2

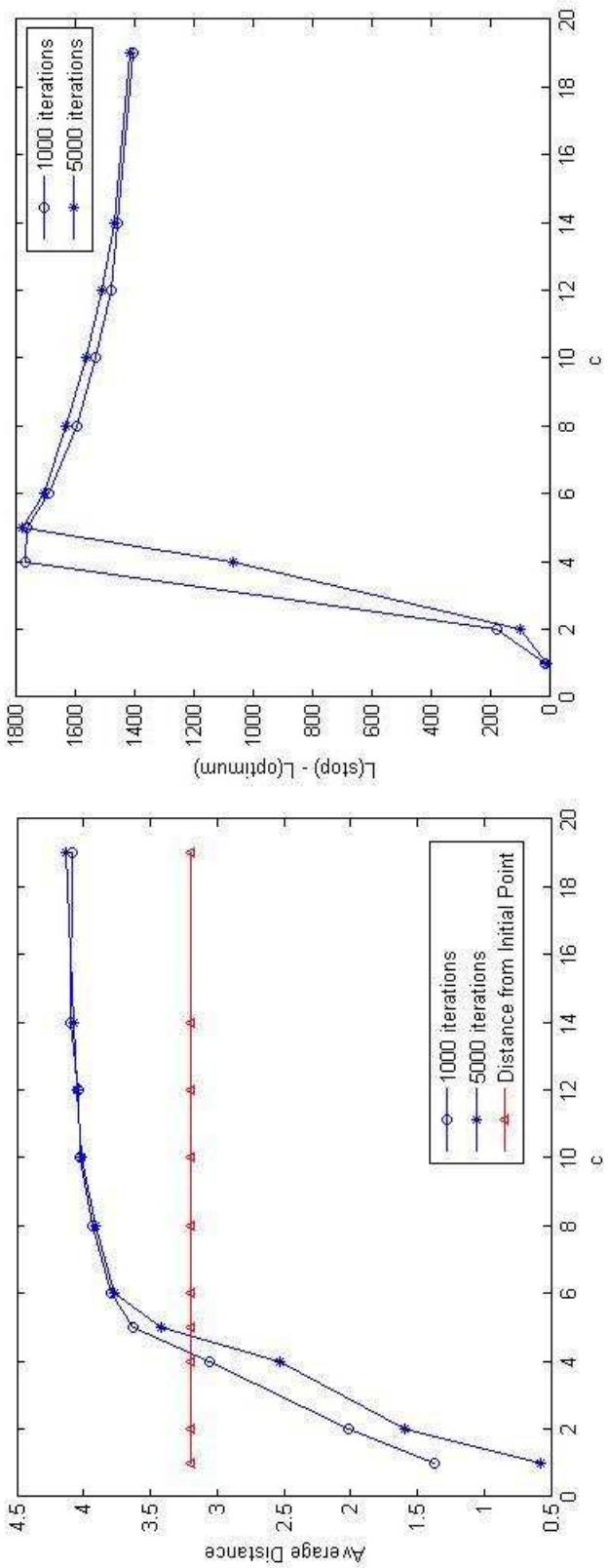

L(stop) - L(optimum)
1000 iterations
5000 iterations
c
Average Distance
1000 iterations
5000 iterations
Distance from Initial Point
c

Figure 3

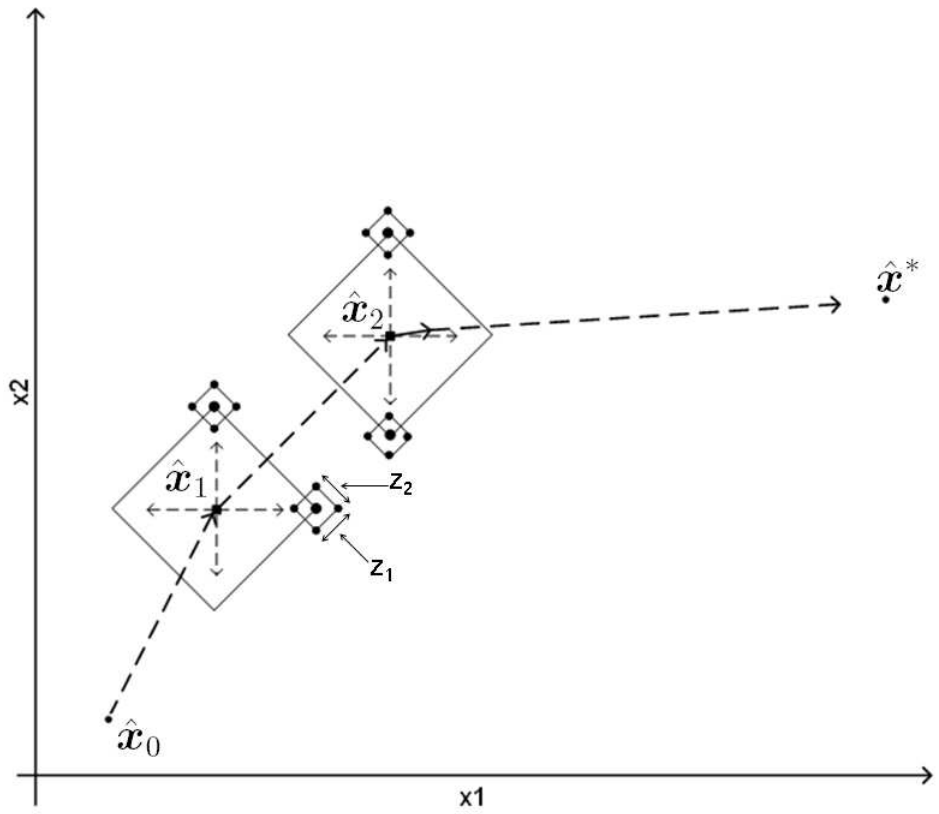


Figure 4

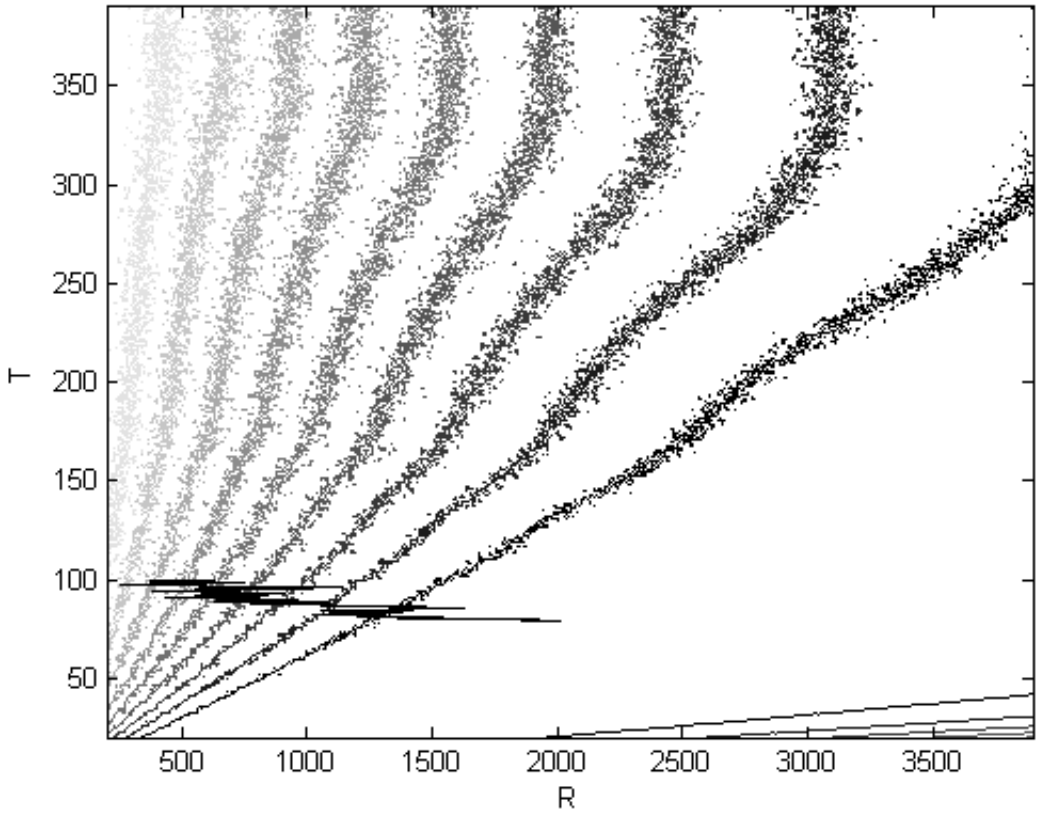


Figure 5

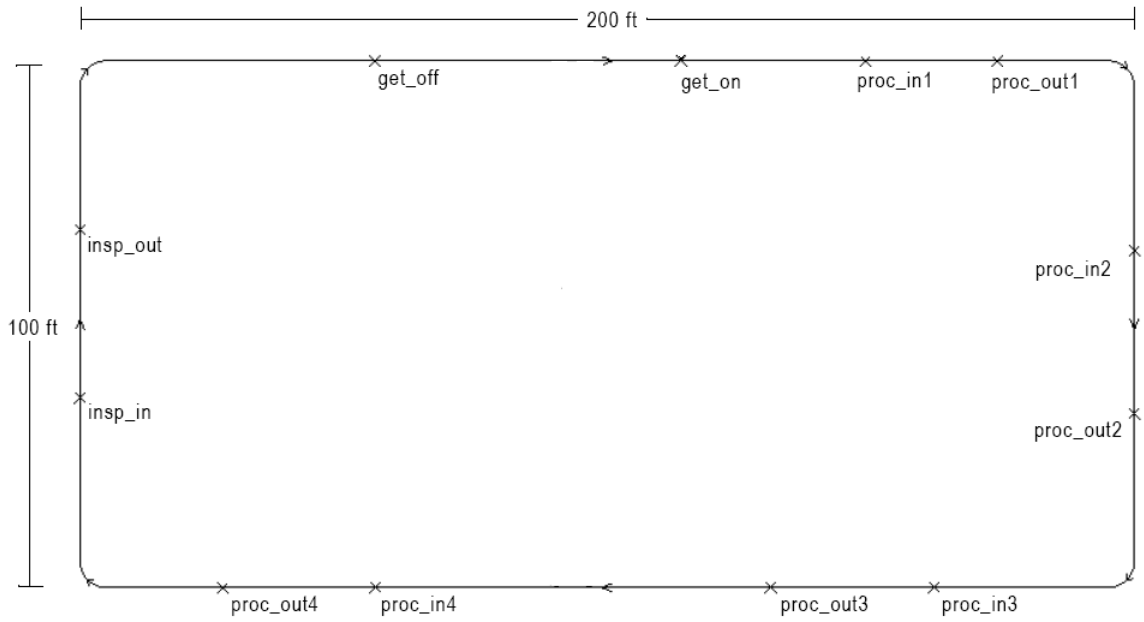

200 ft
100 ft
get_off
get_on
proc_in1
proc_out1
insp_out
proc_in2
insp_in
proc_out2
proc_out4
proc_in4
proc_out3
proc_in3

Figure 6

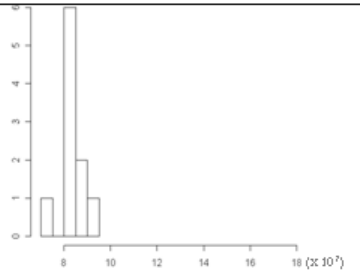
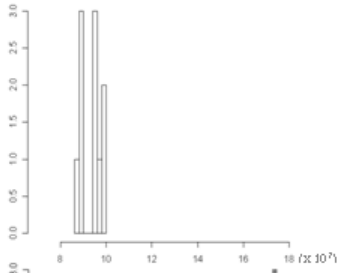
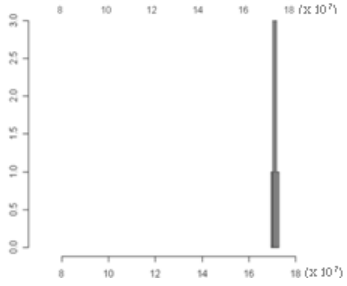
Initial Point $(B_1, B_2, B_3, B_4, B_5)$	$\bar{\bar{\boldsymbol{B}}}$ $(\bar{\bar{B}}_1, \bar{\bar{B}}_2, \bar{\bar{B}}_3, \bar{\bar{B}}_4, \bar{\bar{B}}_5)$	$S_{\hat{B}}$	Histogram of MSE(C)
(3,11,1,9,2)	(11.55, 19, 10, 11.55, 7.66)	(2.65, 5.31, 3,57, 5.05, 4.09)	
(1,17,5,3,12)	(11.44, 25.88,12.55,11.33,19.55)	(2.50, 2.02, 2.60, 2.44,2,92)	
(2,4,16,7,18)	(33.12, 34.75, 33.75, 31.25, 33.87	(2.29, 0.70, 1.48, 5.46, 1.24)	

Figure 7