

STOCHASTIC TRUST REGION RESPONSE SURFACE CONVERGENT METHOD FOR GENERALLY-DISTRIBUTED RESPONSE SURFACE

Kuo-Hao Chang

Dept. of Industrial Engineering and Engineering Management
National Tsing Hua University
Hsinchu 30013, Taiwan ROC

Hong Wan

School of Industrial Engineering
Purdue University
West Lafayette, IN 47907-2023, USA

ABSTRACT

Simulation optimization refers to the iterative procedure in search of the optimal parameter when the objective function can only be evaluated by stochastic simulation. STRONG (Stochastic Trust Region Response Surface Convergent Method) is a newly developed design-of-experiments based simulation optimization method. It incorporates the idea of trust region method (TRM) for deterministic optimization into the traditional response surface methodology (RSM) to eliminate the human intervention required by RSM and to achieve the desired convergence. In the earlier paper, we proved the convergence of STRONG and demonstrated its computational efficiency. The original STRONG assumes the stochastic response follows a normal distribution. This paper relaxes the normal assumption and develops a framework called STRONG-X which is applicable for generally distributed additive noise with bounded variance. The convergence of STRONG-X can be proved and the generality of STRONG-X makes it an appealing method.

1 INTRODUCTION

Stochastic optimization refers to the minimization (or maximization) of a function in the presence of randomness, which in practice has wide applications. For example, a financial manager is interested in determining the optimal portfolio so as to maximize the expected total profits incurred by the stochastic asset prices, and/or a production manager can decide the optimal production plan so as to minimize the expected inventory cost incurred by the stochastic customer demand. Mathematically, the optimization problem can be written as:

$$\text{minimize } g(x), \quad x \in \mathbb{X}^p, \quad (1)$$

where $g(x) = E(G(x, \omega))$ is the objective function, $G(x, \omega)$ is the stochastic response, ω represents the stochastic effect of the system, x is the controllable parameter vector of dimension p , and \mathbb{X}^p is the p -dimensional parameter space, either defined explicitly or implicitly. When $G(x, \omega)$ can only be evaluated through simulation, the iterative procedure in search of the optimal parameter x for problem (1) is known as “simulation-based optimization” (henceforth simulation optimization).

Recent development of methodology in simulation optimization have been discussed in a considerable amount of literature, for example, Banks (1998), Fu (2002), Tekin and Sabuncuoglu (2004), and Fu (2006). Chang et al. (2007, 2009) proposed a new RSM-based method called “Stochastic Trust Region respONse surface converGent method,” abbreviated as STRONG, which applies well-studied statistical techniques such as design of experiments (DOE) and regression analysis that can significantly improve the efficiency of the procedure. The numerical evaluation shows that the computational gain of STRONG is significant, especially for the large-scale simulation optimization problems. The original STRONG assumes normal additive noises for the stochastic response $G(x, \omega)$. In this paper we relax this normal assumption and develop a more general framework we called STRONG-X (the “X” represents a relaxation) which is an algorithmic extension of STRONG for solving the simulation optimization problems with generally distributed response surface.

This article is organized as follows. In section 2, we will state the assumptions, present the algorithm STRONG-X, and comparing it with the original STRONG. The theoretical performance of STRONG-X will be presented in Section 3, and the numerical evaluation is demonstrated in Section 4. We conclude with the future direction in Section 5.

2 The STRONG-X Algorithm

2.1 problem definition

Mathematically, the stochastic response $G(x, \omega)$ can be expressed as:

$$G(x, \omega) = g(x) + \varepsilon_x, \quad (2)$$

where $g(x)$ is an unknown deterministic function (usually complicated) and ε_x is a random error (noise) induced by ω . In this paper, we assume $\varepsilon_x \sim F(\cdot)$, where $F(\cdot)$ represents a general distribution with mean zero and variance σ_x^2 . We further assume σ_x^2 is a continuous function, which can change with the location x , $\inf_{x \in \mathbb{X}^p} \sigma_x^2 > 0$ (therefore the problem will not degenerate to a deterministic problem) and $\sup_{x \in \mathbb{X}^p} \sigma_x^2 < \infty$. Based on our formulation, problem (1) is equivalent to $\min_{x \in \mathbb{X}^p} g(x)$. Notably, in classic RSM, the random error ε_x is typically assumed to have equal variances across the parameter space, i.e., $\sigma_x^2 = \sigma^2$, for all $x \in \mathbb{X}^p$. In our work, the homogeneity assumption is relaxed to heterogeneity. This relaxation is more realistic for simulation experiments. Further assumptions are stated as follows.

Assumption 1. $E(\sup_{x \in \mathbb{X}^p} |G(x, \omega)|) < \infty$.

Assumption 2. *The objective function $g(x)$ is bounded below, twice differentiable and the gradient and Hessian matrix are uniformly bounded in \mathbb{X}^p , i.e., $\forall x \in \mathbb{X}^p$ there exist two constants α_1 and β_1 such that $\|\nabla g(x)\| \leq \alpha_1$, and $\|H(x)\| \leq \beta_1$, where $\|\cdot\|$ denotes the Euclidean norm.*

Assumption 3. *For any x^k , there exists a sufficiently small region Δ_T , such that for $\|x - x^k\| \leq \Delta_T$, the stochastic response $G(x, \omega)$ can be expressed as*

$$G(x, \omega) = g(x^k) + \nabla g^T(x^k)(x - x^k) + \frac{1}{2}(x - x^k)^T H(x^k)(x - x^k) + \varepsilon_x, \quad (3)$$

where ε_x is a normal random variable with $E(\varepsilon_x) = 0$ and $\text{Var}(\varepsilon_x) = \sigma_x^2$.

Assumption 1 is a technical condition requiring the family $\{|G(x, \cdot)|, x \in \mathbb{X}^p\}$ is dominated by an integrable function. Assumption 2 defines the functional nature of the underlying response surface $g(x)$. Consider Assumption 3, at any iteration k , let x^k be the current solution. Based on Taylor Theorem (Trench (2003)), the underlying response surface can be expressed as

$$g(x) = g(x^k) + \nabla g^T(x^k)(x - x^k) + \frac{1}{2}(x - x^k)^T H(x^k)(x - x^k) + o(\|x - x^k\|^2),$$

where $o(\|x - x^k\|^2)$ represents the effect of higher-order terms, which is approximately equal to zero when x is sufficiently close to x^k . Since for any given ω , the induced noise ε_x is a constant, when the trust region is sufficiently small (i.e., x is sufficiently close to x^k), the effect of higher-order terms will be dominated by noise almost surely. Alternatively stated, the effect of higher-order terms is negligible comparing to the noise in this case. Therefore, it is reasonable to assume that when the trust region is sufficiently small (i.e., x is sufficiently close to x^k), the effect of higher-order terms can be ‘‘absorbed’’ into the noise without affecting the noise distribution, which justifies Assumption 3.

In addition, we assume Strong Law of Large Numbers (SLLN) holds for $x \in \mathbb{X}^p$, i.e., the sample mean function

$$\bar{G}_N(x) = \frac{\sum_{i=1}^N G(x, \omega_i)}{N} \rightarrow g(x) \quad \text{w.p.1,} \quad (4)$$

as $N \rightarrow \infty$, for every $x \in \mathbb{X}^p$.

2.2 STRONG-X

The main framework of STRONG-X is presented in Figure 1. The shaded block represents the major difference between STRONG and STRONG-X. The detailed algorithm of STRONG-X is presented in Figure 2. The notations associated with the algorithm are summarized as follows:

N_c : initial sample size of the center point.

N_d : sample size of the non-center design points.

N_c^{ki} : sample size of the center point when the algorithm is at the i th inner-loop of iteration k .

$n(k)$: number of current design points at iteration k .

$n(k_i)$: number of current design points at the i th inner-loop of iteration k .

$\hat{r}^k(\cdot)$: metamodel-predicted response at iteration k .

$\widehat{G}_N(\cdot)$: average observations with sample size N .

$\widehat{g}^k(\cdot)$: Intercept estimate at iteration k .

$\widehat{\nabla g}^k(\cdot)$: gradient estimate at iteration k .

$\widehat{H}^k(\cdot)$: Hessian estimate in iteration k .

$X_{n(k)}$: design matrix at iteration k .

$X'_{n(k)}$: design matrix with interaction and quadratic columns at iteration k .

$Y_{n(k)}$: centralized response vector at iteration k .

Δ_T^0 : initial trust region size.

$\tilde{\Delta}_T$: threshold of trust region size for switching from Stage I to Stage II.

Δ_T^k : size of trust region at iteration k .

ρ^k : calculated ratio at iteration k .

ρ^{ki} : calculated ratio at the i th inner-loop of iteration k .

Similar as STRONG, STRONG-X is also a metamodel-based algorithm. At each iteration, it constructs first-order and/or second-order regression models to approximate the underlying response surface and to find the new solution. The approximation, as well as the new solution, are constrained within a local area (trust region), which is updated at each iteration based on the quality of the new solution. If the new solution is good, the trust region will expand or remain the same; otherwise the trust region will shrink. Specifically, the algorithm has three stages. In Stage I, the algorithm will approximate the underlying response surface with a first-order model. Based on the model, the algorithm determines the optimal solution within the trust region. If the solution is not satisfactory (judged by the ratio test), the trust region will shrink. If the trust region becomes smaller than a user-specified threshold, then the algorithm will switch to Stage II, where a second-order model will be used. If the second-order model can still not find a satisfactory solution, the algorithm will move to Inner-Loop and a series of second-order models will be constructed with accumulated design points. In summary, STRONG-X inherits most of the details of STRONG including the metamodel construction, selection of experimental designs, and trust region updating techniques. Due to the space limitations, we do not repeat every detail in here, but refer the interested readers to Chang et al. (2009) for more information. The major difference between these two methods is that STRONG-X uses only ratio test to determine whether the new solution should be accepted while STRONG uses both the ratio and the hypothesis tests. For convergence to hold, at each iteration in STRONG-X, the sample size used for the center point need to be increased as shown in Figure 1. We will discuss more details in the next section.

2.3 Comparison of STRONG and STRONG-X

As we described before, STRONG uses a ratio and a hypothesis test to determine whether the new solution should be accepted. Since STRONG-X does not presume any specific distribution of the stochastic response, the hypothesis test used in STRONG is no longer applicable. In STRONG-X, we purely count on ratio test to determine whether the new solution should be accepted or not. To ensure the correctness of the ratio test, the sampling error inherent in the ratio test needs to be effectively controlled. This can be accomplished by requiring the sample size used at the center point to increase at a certain rate. Let $N_c^1, N_c^2, \dots, N_c^k, \dots$ be a sequence of integer numbers representing the sample size used at the center point at iteration $1, 2, \dots, k, \dots$. The $\{N_c^k\}$ is called the *schedule* of sample sizes associated with the algorithm. In STRONG-X, we require schedule $\{N_c^k\}$ to satisfy the following Sample Size Allocation Rule (SSAR):

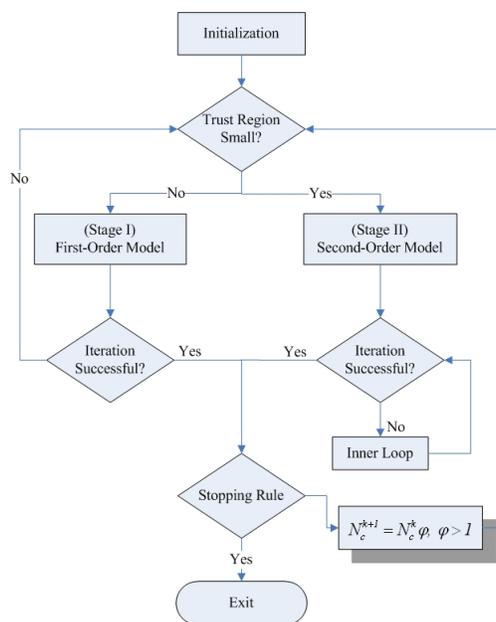


Figure 1: The Main Framework of STRONG-X

Definition 1. The schedule of sample size $\{N_c^k\}$ is required to satisfy $\lim_{k \rightarrow \infty} \left| \frac{N_c^{k+1}}{N_c^k} \right| > \lim_{k \rightarrow \infty} \left| \frac{\beta_k^2}{\beta_{k+1}^2} \right|$, where $\beta_k > 0$ is a convergent sequence that satisfies $\sum_{k=1}^{\infty} \beta_k < \infty$,

If the schedule $\{N_c^k\}$ satisfies this SSAR, we can prove Theorem 1, which shows that the difference between the estimated “accumulated reduction” and actual “accumulated reduction” is bounded by a constant; so is the accumulated differences between the sample mean and the actual value of $g(x)$.

Theorem 1. Suppose $\{N_c^k\}$ follows the SSAR in Definition 1, then w.p.1, $\sum_{k=1}^{\infty} (g(x^k) - g(x^{k+1}))$ only differs from $\sum_{k=1}^{\infty} (\widehat{g}^k(x^k) - \widehat{g}^{k+1}(x^{k+1}))$ by a constant; and $\sum_{k=1}^{\infty} |\widehat{g}^k(x^k) - g(x^k)| < \infty$

In our numerical evaluation, the sample size is increased by: $N_c^{k+1} = \lceil N_c^k \cdot \psi \rceil$, where $\psi = 1.1$. To verify it satisfy SSAR, consider the convergent series $\sum_{k=1}^{\infty} \frac{1}{k^2}$, we can see that

$$\lim_{k \rightarrow \infty} \left| \frac{N_c^{k+1}}{N_c^k} \right| = \psi > \lim_{k \rightarrow \infty} \left| \frac{\beta_k^2}{\beta_{k+1}^2} \right| = 1.$$

As Figure 3 shows, the initial sample size for the center point $\{N_c^k\}$ in STRONG is a fixed constant (the horizontal line) but can be increased to $\{N_c^{k'}\}$ in Inner-Loop. $\{N_c^k\}$ in STRONG-X, however, is required to increase following the SSAR as in Definition 1 (the exponential curve) and can be further increased in Inner-Loop as Figure 4 shows.

It should also be noted that in Inner-Loop, STRONG accumulates all visited solutions at iteration k in a set and always picks the best one as the candidate solution at each loop. STRONG-X, on the other hand, only uses the current solution as the candidate solution. No visited solutions are kept. The machinery of visited-solution set in STRONG is to prevent the hypothesis testing from incorrectly rejecting a good solution. In STRONG-X, however, there is no hypothesis test and therefore this machinery is not employed.

Main Framework

Step 0. Set the iteration count $k = 0$. Select an initial solution x^0 , initial sample size of the center point N_c and of each design point N_d , initial trust region size Δ_T^0 , the switch threshold $\tilde{\Delta}_T$ satisfying $\Delta_T^0 > \tilde{\Delta}_T > 0$, and constants η_0 , η_1 , γ_1 , and γ_2 satisfying $0 < \eta_0 < \eta_1 < 1$ and $0 < \gamma_1 < 1 < \gamma_2$.

Step 1. Let $k = k + 1$. If $\Delta_T^k > \tilde{\Delta}_T$, go to **Stage I**; otherwise, go to **Stage II**.

Step 2. If the termination criterion is satisfied, stop and return the final solution. Otherwise go to **Step 1**.

Stage I. First-order model (for k th iteration)

- a. Employ a proper first-order design. Take N_d observations at each design point and N_c observations at the center point x^k .
- b. Build a first-order model. Solve the subproblem \mathcal{P}_k and obtain a new solution x^{k+1} .
- c. Accept/Reject the new solution and update the trust region:
 Compute $\rho^k = \frac{\text{observed achieved reduction}}{\text{predicted reduction}} = \frac{\bar{G}_{N_c}(x^k) - \bar{G}_{N_c}(x^{k+1})}{r^k(x^k) - r^k(x^{k+1})}$.
 - if $\rho^k < \eta_0$, let $x^{k+1} = x^k$, $\Delta_T^{k+1} = \gamma_1 \Delta_T^k$ and return to **Step 1** in the main framework.
 - if $\eta_0 \leq \rho^k < \eta_1$, move to x^{k+1} , and let $\Delta_T^{k+1} = \Delta_T^k$ and return to **Step 1** in the main framework.
 - if $\rho^k \geq \eta_1$, move to x^{k+1} , and let $\Delta_T^{k+1} = \gamma_2 \Delta_T^k$ and return to **Step 1** in the main framework.

Stage II. Second-order model (for k th iteration)

- a. Employ a proper second-order design with the center point. Take N_d observations at each design point and N_c observations at the center point x^k .
- b. Build a second-order model. Solve the subproblem \mathcal{P}_k and obtain a new solution x^{k+1} .
- c. Accept/Reject the new solution x^{k+1} and update the trust region:
 Compute $\rho^k = \frac{\bar{G}_{N_c}(x^k) - \bar{G}_{N_c}(x^{k+1})}{r^k(x^k) - r^k(x^{k+1})}$.
 - if $\rho^k < \eta_0$, go to Inner-Loop.
 - if $\eta_0 \leq \rho^k < \eta_1$, move to x^{k+1} , $\Delta_T^{k+1} = \Delta_T^k$ and return to **Step 1** in the main framework.
 - if $\rho^k \geq \eta_1$, move to x^{k+1} , let $\Delta_T^{k+1} = \gamma_2 \Delta_T^k$ and return to **Step 1** in the main framework.

Inner-Loop (for i th loop in k th iteration)

1. Set subiteration count $i = 0$. Let $x^{k_0} = x^k$, $\Delta_T^{k_0} = \Delta_T^k$, $N_c^{k_0} = N_c$.
2. Let $i = i + 1$. Construct a new second-order design within the trust region $\Delta_T^{k_i}$. Update $X'_{n(k_i)}$.
3. Build a second-order model with $X'_{n(k_i)}$. Solve the subproblem \mathcal{P}_{k_i} and obtain a new solution x^{k_i} .
4. Accept/Reject the best solution and update the trust region.
 Compute $\rho^{k_i} = \frac{\bar{G}_{N_c^{k_i}}(x^{k_0}) - \bar{G}_{N_c^{k_i}}(x^{k_i})}{r^{k_i}(x^{k_0}) - r^{k_i}(x^{k_i})}$.
 - if $\rho^{k_i} < \eta_0$, $\Delta_T^{k_{i+1}} = \gamma_1 \Delta_T^{k_i}$, and go back to Step 2 in Inner-Loop.
 - if $\eta_0 < \rho^{k_i} < \eta_1$, $\Delta_T^{k_{i+1}} = \Delta_T^{k_0}$ and return to **Step 1** in the main framework.
 - elseif $\rho^{k_i} > \eta_1$, $\Delta_T^{k_{i+1}} = \gamma_2 \Delta_T^{k_0}$ and return to **Step 1** in the main framework.

Figure 2: STRONG-X Algorithm

3 Theoretic Performance

Given that the gradient estimator in metamodel is uniformly convergent, which can be achieved with appropriate experimental design and regression method (Chang et al. 2009), we can prove the following results:

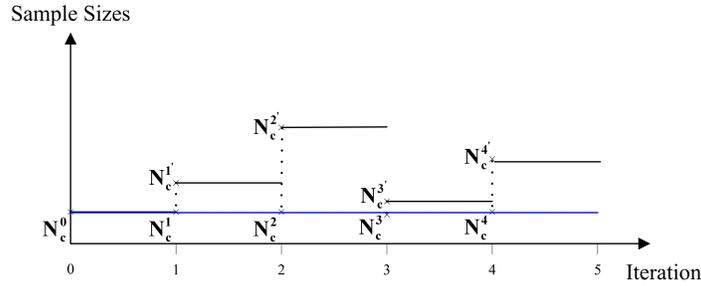


Figure 3: An Illustration of the Schedule of Sample Sizes Used at the Center Point in STRONG

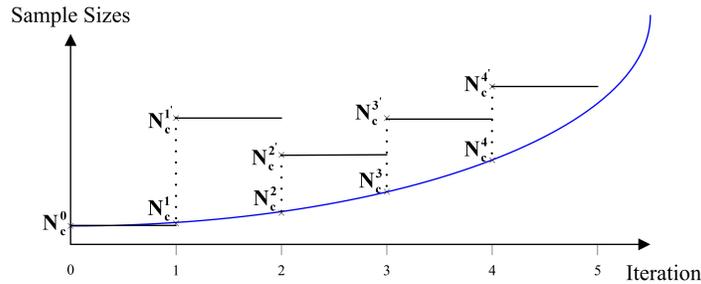


Figure 4: An Illustration of the Schedule of Sample Sizes Used at the Center Point in STRONG-X

Theorem 2. For any x^k , if $\|\nabla g(x^k)\| = \xi > 0$ and the algorithm is in Inner-Loop, for sufficiently large i , there exists a x^{k_i} which will pass the ratio test w.p.1.

Theorem 3. For any initial point x^0 , if STRONG-X has infinitely many successful iterations, then w.p.1

$$\lim_{k \rightarrow \infty} \|\nabla g(x^k)\| = 0.$$

Theorem 2 shows that if the algorithm is not at a stationary point, it can always find a satisfactory solution to move to; in other words, the algorithm will not get stuck at a nonstationary solution. Theorem 3 shows that the algorithm will converge to a stationary point of the underlying response surface with probability one.

4 Numerical Evaluation

The numerical tests presented here is in parallel with that for STRONG as discussed in Chang et al. (2009). Resolution III fractional factorial designs and Central Composite Designs are selected to fit the first-order and second-order metamodels, respectively. We compare the performance of STRONG-X, STRONG-X without using DOE (STRONG-ND), and STRONG-X without using trust region updating technique (STRONG-DT). Specifically, for STRONG-ND, we randomly pick the same number of design points within the trust region instead of employing DOE. For STRONG-DT, we let the size of trust region diminish as $1/(0.5 * \text{iteration})$ instead of using the trust region updating technique. We also compare STRONG-X with modified Nelder-Mead method (MNM), which is a heuristic direct-search-based search method. We code it based on **fminsearch** function in MATLAB (version 7) with proper modifications suggested in Barton and Ivey (1996).

The stochastic response takes the form of $G(x, \omega) = g(x) + \varepsilon_x$, where $g(x)$ is the objective function and $\varepsilon_x \sim \text{Uniform}(-d, d)$, where d has four possible settings: (1) $d = 100$; (2) $d = 200$; (3) $d = 0.1 * g(x)$; and (4) $d = g(x)$. The first two settings assume homogeneous response surfaces as in the classic RSM; and the last two settings give heterogeneous response surfaces, which are typical in simulation experiments. Notice that in setting (3) and (4), when the initial solution is selected far from the optimum, $g(x)$ will be large, hence the variance of the response will be huge. We consider two dimensions for each test

problem: six and fifteen input variables. The fifteen dimensional problems constitute good examples to test the algorithm performance in large-scale cases.

We use three test problems with distinct response surfaces to evaluate the performance of the algorithm. Our first test problem is based on the well-known extended Rosenbrock test function in the optimization literature (More et al. 1981). For general p , the objective function has the fourth-order polynomial form:

$$g(x) = \sum_{i=1}^{p/2} [100(x_{2i} - x_{2i-1}^2)^2 + (1 - x_{2i-1})^2]. \quad (5)$$

When $p = 15$, we can rewrite (5) as follows:

$$g(x) = \sum_{i=1}^7 [100(x_{2i} - x_{2i-1}^2)^2 + (1 - x_{2i-1})^2] + (1 - x_{15})^2. \quad (6)$$

The function $g(x)$ has one global optima at $x^* = (1, 1, 1, 1, 1, 1)$ and $x^* = (1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1)$ for six-dimensional and fifteen-dimensional problem respectively. The optimal functional value is $g(x^*) = 0$ for both dimensional problems.

The second test problem is based on the famous Beale function that has been discussed extensively in nonlinear optimization literature (More, Garbow, and Hillstrom 1981). For general p , the objective function is

$$g(x) = \sum_{i=1}^{p/2} [1.5 - x_{2i-1}(1 - x_{2i})]^2 + [2.25 - x_{2i-1}(1 - x_{2i}^2)]^2 + [2.625 - x_{2i-1}(1 - x_{2i}^3)]^2. \quad (7)$$

When $p = 15$, we modify (7) by adding one term x_{15}^2 in the last summand as follows:

$$g(x) = \sum_{i=1}^7 [1.5 - x_{2i-1}(1 - x_{2i})]^2 + [2.25 - x_{2i-1}(1 - x_{2i}^2)]^2 + [2.625 - x_{2i-1}(1 - x_{2i}^3)]^2 + x_{15}^2.$$

The global optima is located at $x^* = (3, 0.5, 3, 0.5, 3, 0.5)$ and $x^* = (3, 0.5, 3, 0.5, 3, 0.5, 3, 0.5, 3, 0.5, 3, 0.5, 3, 0.5, 0)$ for six-dimensional and fifteen-dimensional problem respectively. The optimal functional value is $g(x^*) = 0$ for both dimensional problems.

The third test problem is a quadratic function. For general p , the objective function is

$$g(x) = \sum_{i=1}^p x_i^2, \quad (8)$$

which has one local optima at $x^* = (0, 0, 0, 0, 0, 0)$ and $x^* = (0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)$ for six-dimensional and fifteen-dimensional problems respectively. The optimal functional value $g(x^*)$ equals to zero for both dimensional problems.

4.1 Parameter Setting

The parameter settings of STRONG-X can improve or degrade the performance of the algorithm for any particular problem as the same situation applies to any other algorithm. The optimal settings depend on the specific problems. In our numerical analysis, we took the same parameter setting as in Chang et al. (2009) (Table 1).

4.2 Results

Table 2 shows twelve scenarios which are combinations of two problem dimensions, three test problems, and two distribution ranges (two variance settings). The four algorithms, including STRONG-X, STRONG-ND, STRONG-DT, and MNM, are implemented on these twelve scenarios. It is remarkable that our initial solutions are set relatively far from the optimum of all tested problems to evaluate the performance of the algorithm in the realistic situation where users have little prior knowledge of the system. The quantity $\log_{10}(g(x) - g(x^*) + 1)$ is selected as a measure of performance, where $g(x^*)$ is the

Table 1: The parameter settings for three test problems

Parameter	Setting
N_c	3
N_d	2
Δ_T^0	2
$\tilde{\Delta}_T$	1.2
η_0	0.01
η_1	0.3
γ_1	0.9
γ_2	1.11

known optimal value of the test problem (in our three test problems, $g(x^*)$ are all equal to zero). The difference between the current and the optimal functional value is added by 1 to avoid the log function going to negative infinity as the algorithm is close to the optimum. Each algorithm was independently implemented for ten times for each scenario. The performance of the algorithm are compared in average sense.

The performance plots for Scenario 1-12 are shown in Figure 5 and 6; and we made the similar observations as the STRONG's performance discussed in Chang et al. (2009). Firstly, the design of experiments can significantly improve the efficiency of the optimization procedure, especially for the large-scale and noisy cases (Scenarios 6, 9, and 10), even when the random error is generally distributed. The application of experimental designs also improve the quality of the final solution as Scenarios 1, 2, 7, 8, 11 and 12 show. Secondly, STRONG-DT has much poorer performance than STRONG-X in all scenarios, which demonstrates the importance of the trust region updating technique. This technique not only improves the efficiency of the algorithm but also plays an important role in convergence of the algorithm as Scenarios 5 and 6 show. Lastly, compared with the Nelder-Mead method (Barton and Ivey 1996), which is a direct-search-based method, STRONG-X has significant better performance in heterogenous case such as Scenarios 2, 4, 6, 8, and 10. In homogenous case as Scenarios 1, 3, 5, 7, and 9, NMM can progress quickly initially, but fail to converge later.

Table 2: Twelve Scenarios

Scenario	Dimension	Test Function	Initial Solution	d
1	6	Rosenbrock	(10,10,10,10,10,10)	100
2	6	Rosenbrock	(10,10,10,10,10,10)	$g(x)$
3	6	Beale	(10,10,10,10,10,10)	100
4	6	Beale	(10,10,10,10,10,10)	$g(x)$
5	6	Quadratic	(30,30,30,30,30,30)	100
6	6	Quadratic	(30,30,30,30,30,30)	$g(x)$
7	15	Rosenbrock	10*(-1,1,1,1,-1,1,1,-1,1,1,-1,1,1,1,-1)	200
8	15	Rosenbrock	10*(-1,1,1,1,-1,1,1,-1,1,1,-1,1,1,1,-1)	$0.1 * g(x)$
9	15	Beale	10*(-1,1,1,1,-1,1,1,-1,1,1,-1,1,1,1,-1)	200
10	15	Beale	10*(-1,1,1,1,-1,1,1,-1,1,1,-1,1,1,1,-1)	$0.1 * g(x)$
11	15	Quadratic	30*(-1,1,1,1,-1,1,1,-1,1,1,-1,1,1,1,-1)	200
12	15	Quadratic	30*(-1,1,1,1,-1,1,1,-1,1,1,-1,1,1,1,-1)	$0.1 * g(x)$

5 Concluding Remarks

In this paper, we propose STRONG-X, an algorithmic extension of STRONG. STRONG-X relaxes the normality assumption of STRONG and can be applied to a wider problem spectrum. Compared with the traditional response surface method, STRONG-X applies the trust region updating technique to achieve the desired automation and convergence property. The framework allows the application of a wide variety of experimental designs. Numerical evaluation shows that the application of design of experiments can significantly improve the efficiency of the optimization procedure. It is worth mentioning that while STRONG-X can also handle the case when the stochastic response is normally distributed, it usually requires more observations than STRONG. Therefore it should not be considered as a replacement of the original STRONG framework, but as an extension.

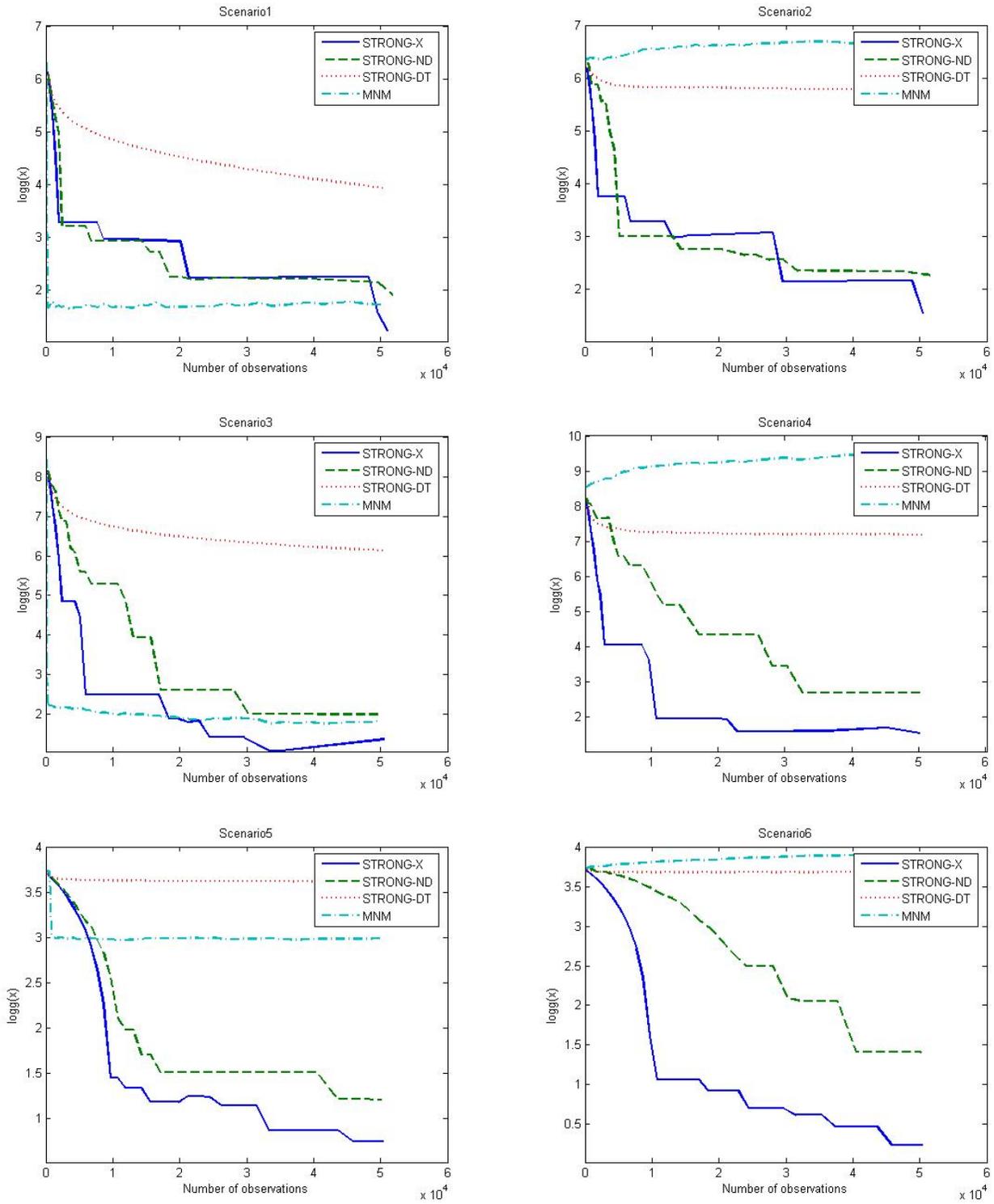


Figure 5: Numerical Results for Scenario 1-6

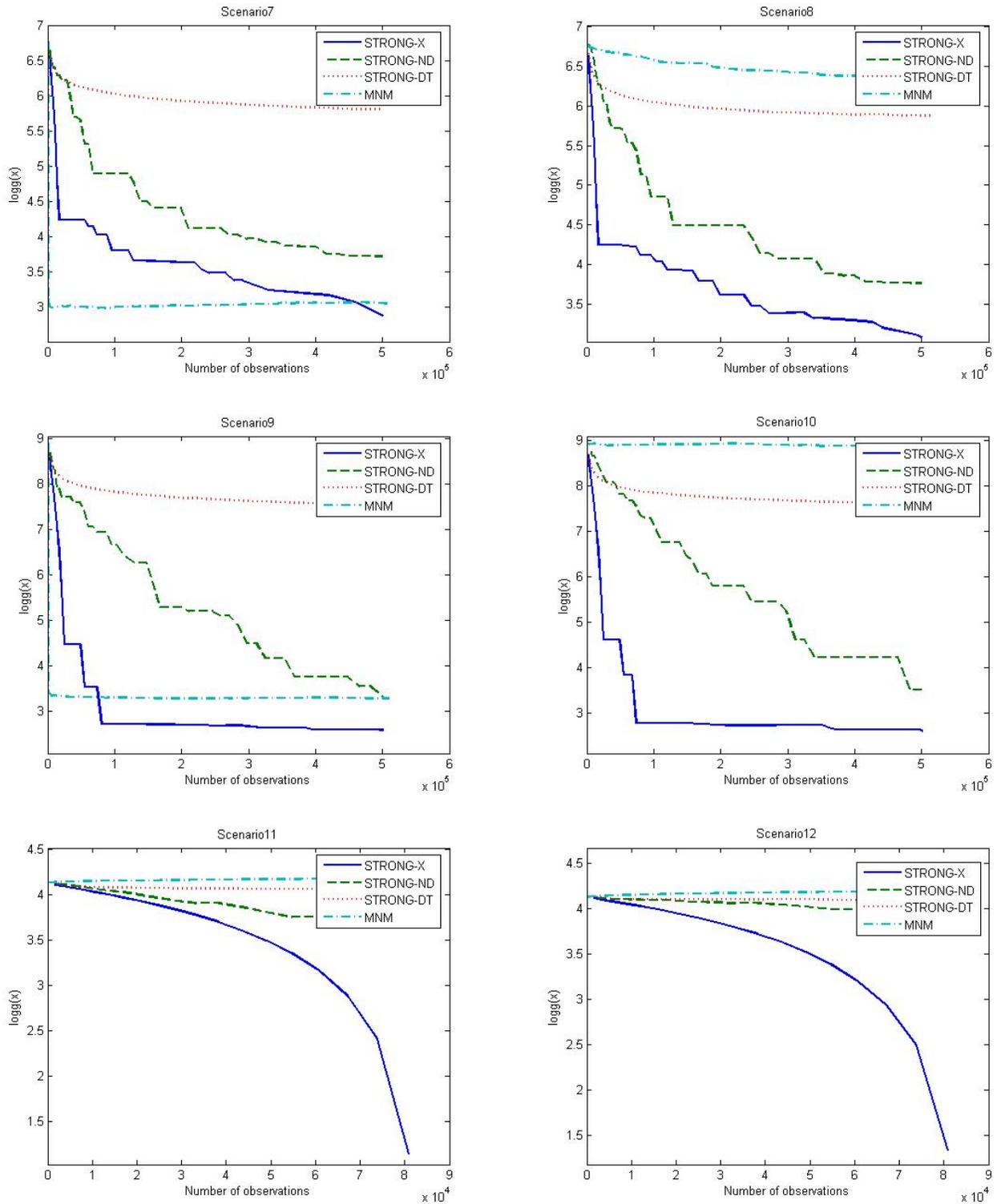


Figure 6: Numerical Results for Scenario 7-12

ACKNOWLEDGMENT

This research was partially supported by Purdue Research Foundation (PRF), Naval Postgraduate School Award N00244-06-C-0002, and Hong Kong Research Grants Council grant CERG 613305. Some materials in this article have been presented in Dr. Kuo-Hao Chang's dissertation (Chang 2008).

REFERENCES

- Banks, J. (Ed.) 1998. *Handbook of simulation*. New York: John Wiley and Sons.
- Barton, R., and J. Ivey. 1996. Nelder-Mead simplex modifications for simulation optimization. *Management Science*, 42 (7): 954–973.
- Chang, K.-H. 2008, December. *Stochastic trust region response surface convergent method for continuous simulation optimization*. Ph. D. thesis, Purdue University, West Lafayette, Indiana.
- Fu, M. 2002. Optimization for simulation: Theory vs. practice. *INFORMS Journal on Computing* 14:192–227.
- Fu, M. 2006. Gradient estimation. *Handbooks in Operations Research and Management Science* 13:575–616.
- More, J., B. Garbow, and K. Hillstom. 1981. Testing unconstrained optimization software. *ACM Transactions on Mathematical Software* 7:17–41.
- Tekin, E., and I. Sabuncuoglu. 2004. Simulation optimization: A comprehensive review on theory and applications. *IIE Transactions* 36:1067–1081.
- Trench, W. 2003. *Introduction to real analysis*. New Jersey: Pearson Education.

AUTHOR BIOGRAPHY

KUO-HAO CHANG is an assistant professor in Industrial Engineering and Engineering Management at National Tsing Hua University. He received his PhD in Industrial Engineering from Purdue University. His research interests include simulation optimization, stochastic modeling and financial engineering. His email address is <chang@mx.nthu.edu.tw>.

HONG WAN is an assistant professor in the School of Industrial Engineering at Purdue University. Her research interests include design and analysis of simulation experiments, simulation optimization, applied statistics, quality management, and healthcare engineering. She has taught a variety of courses and is a member of INFORMS and ASA. She currently serves as the associate editor of ACM Transactions on Modeling and Computer Simulation. Her e-mail address is <hwan@purdue.edu>.