Efficient Simulation Budget Allocation for Selecting an Optimal Subset

Chun-Hung Chen, Donghai He
Department of Systems Engineering and Operations Research, George Mason University, Fairfax, Virginia 22030
{cchen9@gmu.edu, dhe1@gmu.edu}

Michael Fu
Robert H. Smith School of Business and Institute for Systems Research, University of Maryland, College Park, Maryland 20742, mfu@umd.edu

Loo Hay Lee
Department of Industrial and Systems Engineering, The National University of Singapore, Kent Ridge, 119260, Singapore, iselee@nus.edu.sg

We consider a class of the subset selection problem in ranking and selection. The objective is to identify the top $m$ out of $k$ designs based on simulated output. Traditional procedures are conservative and inefficient. Using the optimal computing budget allocation framework, we formulate the problem as that of maximizing the probability of correctly selecting all of the top-$m$ designs subject to a constraint on the total number of samples available. For an approximation of this correct selection probability, we derive an asymptotically optimal allocation and propose an easy-to-implement heuristic sequential allocation procedure. Numerical experiments indicate that the resulting allocations are superior to other methods in the literature that we tested, and the relative efficiency increases for larger problems. In addition, preliminary numerical results indicate that the proposed new procedure has the potential to enhance computational efficiency for simulation optimization.

Key words: simulation optimization; computing budget allocation; ranking and selection

History: Accepted by Marvin Nakayama, Area Editor for Simulation; received August 2006; revised August 2007, October 2007, and November 2007; accepted December 2007. Published online in Articles in Advance May 30, 2008.

1. Introduction

We consider the problem of selecting the top $m$ out of $k$ designs, where the performance of each design is estimated with noise (uncertainty). The primary context is simulation, where the goal is to determine the best allocation of simulation replications among the various designs to maximize the probability of selecting all top-$m$ designs. This problem setting falls under the well-established branch of statistics known as ranking and selection or multiple comparison procedures (see Bechhofer et al. 1995). In the context of simulation, Goldsman and Nelson (1998) provide an overview of this field; see also Andradóttir et al. (2005) and Swisher et al. (2003).

Most of the ranking-and-selection research has focused on identifying the best design. Typical of these are two-stage or sequential procedures that ultimately return a single choice as the estimated optimum, e.g., Dudewicz and Dalal (1975) and Rinott (1978). Even the traditional “subset selection” procedures aim at identifying a subset that contains the best design, dating back to Gupta (1965), who presented a single-stage procedure for producing a subset (of random size) containing the best design with a specified probability. Extensions of this work relevant to the simulation setting include Sullivan and Wilson (1989), who derive a two-stage subset selection procedure that determines a subset of maximum size $m$ that, with a specified probability, contains at least one design whose mean response is within a prespecified distance from the optimal mean response. This indifference zone procedure approach also results in a subset of random size, and the designs are assumed to follow a normal distribution, with independence between designs assumed and unknown and unequal moments. The primary motivation for such procedures is screening, whereby the selected subset can be scrutinized further to find the single optimum.

To reiterate, instead of selecting the very best design from a given set or finding a subset that is highly likely to contain the best design, the objective in this paper is to find all top-$m$ designs. The only substantive work we are aware of addressing this problem is Koenig and Law (1985), who along the lines of the procedure in Dudewicz and Dalal (1975), develop a two-stage procedure for selecting all the $m$ best...
designs (see also §10.4 of Law and Kelton 2000 for an extensive presentation of the problem and procedure). The number of additional simulation replications for the second stage is computed based on a least favorable configuration, resulting in very conservative allocations, so that the required computational cost is much higher than actually needed.

In this paper, we develop an efficient approach for such a class of ranking-and-selection problems. Unlike traditional frequentist approaches constrained with least favorable configuration, our procedure is developed using a Bayesian model. The rationale for the adoption of the Bayesian model is the ease of derivation of the solution approach. Conservative least-favorable configuration is no longer required and so the efficiency can be enhanced. Further comparison of the Bayesian model with the frequentist model can be found in Inoue and Chick (1998) and Inoue et al. (1999).

To improve efficiency for ranking and selection, several approaches have been explored for problems of selecting a single best design. Intuitively, to ensure a high probability of correct selection, a larger portion of the computing budget should be allocated to those designs that are critical in the process of identifying the best design. A key consequence is the use of both the means and variances in the allocation procedures, rather than just the variances, as in Dudewicz and Dalal (1975) and Rinott (1978). Among examples of such approaches, the optimal computing budget allocation (OCBA) approach by Chen et al. (1997, 2000) maximizes a simple heuristic approximation of the correct selection probability; extensions of the OCBA approach include Lee et al. (2004), who consider multiple objective functions; Trailovic and Pao (2004), who consider the objective of minimizing variance; and Fu et al. (2007), who consider correlated sampling. The approach by Chick and Inoue (2001a, b) estimates the correct selection probability with Bayesian posterior distributions, and allocates further samples using decision-theory tools to maximize the expected value of information in those samples. The procedure by Kim and Nelson (2006) allocates samples to provide a guaranteed lower bound for the frequentist probability of correct selection integrated with ideas of early screening. More recently, Branke et al. (2007) provide a nice overview and extensive comparison for some of the aforementioned selection procedures. These procedures are developed to remedy the drawbacks of inefficiency for traditional two-stage procedures by allocating simulation samples in a more efficient manner. However, all of this work has focused on selecting the single best, whereas no such research results exist for efficiently selecting the top-m designs since the Koenig and Law (1985) paper appeared. This paper aims to fill this gap by providing an efficient allocation procedure for selecting the m best designs.

Development of such an efficient procedure for selecting the m best designs is also beneficial to some recent developments in global optimization that, when applied to the simulation setting, require the selection of an “elite” subset of good candidate solutions in each iteration of the algorithm. Examples of these include the cross-entropy method (CE; see Rubinstein and Kroese 2004), the population-based incremental learning method (PBIL; see Rudlof and Köppen 1996), the model reference adaptive search method (Hu et al. 2007a, b), genetic algorithms (Holland 1975, Chambers 1995), and more generally, evolutionary population-based algorithms that require the selection of an “elite” population in the evolutionary process (see Fu et al. 2006). Instead of trying to find a subset that contains the single best among a currently generated set of candidate solutions, the objective is to find an optimal subset such that all members are among the best performers in that candidate set. The reason for this requirement is that this entire subset is used to update the subsequent population or sampling distribution that drives the search for additional candidates. A subset with poor performing solutions will result in an update that leads the search in a possibly misleading direction. The overall efficiency of these types of simulation optimization algorithms depends on how efficiently we simulate the candidates and correctly select the top-m designs. The algorithm developed herein is generic enough so that it can be integrated with any such evolutionary population-based search methods. Note that among the selected m designs, there is no further ranking done within the set. Again, this is consistent with the requirements of the CE and PBIL methods, as well as other evolutionary population-based methods that require an “elite” population of some type.

The contribution of this paper is threefold. From a ranking-and-selection perspective, we offer a heuristic for selecting all top-m designs out of k, where our empirical studies suggest that it can be more efficient than existing methods. From the computing budget allocation perspective, our heuristic illustrates how the previous OCBA method for identifying a single best design can be modified to instead select an optimal subset. From a simulation optimization perspective, we illustrate one possible way of efficiently allocating simulation replications for those evolutionary population-based search methods that require an elite set to guide the search; however, further research is required to fully realize the benefits of such integration, which is clearly highly dependent on the search method adopted. This paper is organized as follows.
In the next section, we formulate the optimal computing budget allocation problem for selecting the top-\(m\) designs. Section 3 derives an allocation based on approximating the correct selection probability and an asymptotic analysis and, based on the analysis, proposes a heuristic sequential allocation scheme. The performance of the resulting allocations is illustrated with a series of numerical examples in §4, including numerical results that indicate that the proposed new procedure has the potential to enhance computational efficiency for simulation optimization. Section 5 concludes the paper.

2. Problem Statement

We introduce the following notation:

\[
T = \text{total number of simulation replications (budget),} \\
k = \text{total number of designs,} \\
m = \text{number of top designs to be selected in the optimal subset,} \\
S_m = \text{set of } m \text{ (distinct) indices indicating designs in selected subset,} \\
N_i = \text{number of simulation replications allocated to design } i, \\
X_{ij} = j\text{th simulation replication for design } i, \\
\bar{X}_i = (1/N_i) \sum_{j=1}^{N_i} X_{ij}, \text{ sample mean for design } i, \\
\bar{\mu}_i = \text{mean for design } i, \\
\sigma_i^2 = \text{variance for design } i, \\
\Phi(x) = (1/\sqrt{2\pi})e^{-x^2/2}, \text{ standard normal probability density function,} \\
\Phi(x) = \int_{-\infty}^{x} \Phi(t)\, dt, \text{ standard normal cumulative distribution function.}
\]

The objective is to find a simulation budget allocation that maximizes the probability of selecting the optimal subset, defined as the set of \(m\) \((< k)\) best designs, for \(m\) a fixed number. Note that rank order within the subset is not part of the objective. In this paper, we will take \(S_m\) to be the \(m\) designs with the smallest sample means. Let \(\bar{F}_r\) be the \(r\)th smallest (order statistic) of \(\{\bar{X}_1, \bar{X}_2, \ldots, \bar{X}_k\}\), i.e., \(\bar{F}_1 \leq \bar{F}_2 \leq \cdots \leq \bar{F}_k\). Then, the selected subset is given by

\[
S_m \equiv \{i_1, i_2, \ldots, i_m\}.
\]

Without loss of generality, we will take the \(m\) best designs as those designs with the \(m\) smallest means, so that in terms of our notation, the correct selection (CS) event is defined by \(S_m\) containing all of the \(m\) smallest mean designs:

\[
\text{CS}_m \equiv \left\{ \bigcap_{i \in S_m} \bigcap_{j \notin S_m} (\bar{X}_i \leq \bar{X}_j) \right\} = \left\{ \max_{i \in S_m} \bar{X}_i \leq \min_{i \notin S_m} \bar{X}_i \right\}. \tag{1}
\]

The OCBA problem is given by

\[
\max_{N_1, \ldots, N_k} P[\text{CS}_m] \\
\text{s.t. } N_1 + N_2 + \cdots + N_k = T. \tag{2}
\]

Here, \(N_1 + N_2 + \cdots + N_k\) denotes the total computational cost assuming the simulation execution times for different designs are roughly the same. This formulation implicitly assumes that the computational cost of each replication is constant across designs. The simulation budget allocation problems given in Chen et al. (2000) are actually special cases of (2) with \(m = 1\). For notational simplification, we will drop the “\(m\)” in \(P[\text{CS}_m]\) in the remaining discussion.

We assume that the simulation output samples \(\{X_{ij}\}\) are normally distributed and independent from replication to replication (with mean \(\bar{X}_i\) and variance \(\sigma_i^2\)), as well as independent across designs. The normality assumption is typically satisfied in simulation because the output is obtained from an average performance or batch means, so that central limit theorem effects usually hold.

3. Approximate Asymptotically Optimal Allocation Scheme

Our approach is developed based on Bayesian setting. We estimate \(P[\text{CS}]\) using the Bayesian model presented in Chen (1996) and He et al. (2007). The mean of the simulation output for each design, \(\bar{X}_i\), is assumed unknown and treated as a random variable. After the simulation is performed, a posterior distribution for the unknown mean \(\bar{X}_i\), \(p(\bar{X}_i \mid X_{ij}, j = 1, \ldots, N_i)\), is constructed based on two pieces of information: (i) prior knowledge of the system’s performance, and (ii) current simulation output. Thus, in the Bayesian framework, the probability of correct selection defined by (1) is given by

\[
P[\text{CS}] = P[\bar{X}_i \leq \bar{F}_r \text{ for all } i \in S_m \text{ and } j \notin S_m], \tag{3}
\]

where \(\bar{F}_r, r = 1, \ldots, k\), denotes the random variable whose probability distribution is the posterior distribution of design \(i\). As in Chen (1996), we assume that the unknown mean \(\bar{X}_i\) has a conjugate normal prior distribution and consider noninformative prior distributions, which implies that no prior knowledge is available about the performance of any design before conducting the simulations, in which case the posterior distribution of \(\bar{X}_i\) is (see DeGroot 1970)

\[
\bar{X}_i \sim N\left(\bar{X}_i, \sigma_i^2/N_i\right).
\]

After the simulation is performed, \(\bar{X}_i\) can be calculated, \(\sigma_i^2\) can be approximated by the sample variance, and the \(P[\text{CS}]\) given by Equation (3) can then be estimated using Monte Carlo simulation. However, because estimating \(P[\text{CS}]\) via Monte Carlo simulation is time-consuming and the purpose of budget allocation is to improve simulation efficiency, we
adopt an approximation of $P[CS]$ using a lower bound.

3.1. Approximating the Probability of Correct Selection

For a constant $c$,

$$P[CS] = P\{\bar{J}_i \leq \bar{J}_j \text{ for all } i \in S_m \text{ and } j \notin S_m\} \geq P\{\bar{J}_i \leq c \text{ and } \bar{J}_j \geq c \text{ for all } i \in S_m \text{ and } j \notin S_m\} = \prod_{i \in S_m} P\{\bar{J}_i \leq c\} \prod_{i \notin S_m} P\{\bar{J}_i \geq c\} \equiv APCSm,$$ \hspace{1cm} (4)

where the last line is due to independence across designs. We refer to this lower bound for $P[CS]$, which can be computed easily and eliminates the need for extra Monte Carlo simulation, as the approximate probability of correct selection for $m$ best (APCSm). The value for $c$ will be between $\bar{J}_m$ and $\bar{J}_{m+1}^*$ and the rationale for this will be explained in §3.3. Using the approximation given by Equation (4), the OCBA problem (2) becomes

$$\begin{align*}
\max_{N_1, \ldots, N_k} & \quad \prod_{i \in S_m} P\{\bar{J}_i \leq c\} \prod_{i \notin S_m} P\{\bar{J}_i \geq c\} \\
\text{s.t.} & \quad N_1 + N_2 + \cdots + N_k = T.
\end{align*}$$ \hspace{1cm} (5)

Now we solve OCBA problem (5), assuming the variables $\{N_i\}$ are continuous.

3.2. Asymptotically Optimal Solution

For notation simplification, we define the variable $\delta_i = \bar{J}_i - c$, $i = 1, 2, \ldots, k$.

For $i \in S_m$,

$$P(\bar{J}_i \leq c) = \int_{-\infty}^{0} \frac{1}{\sqrt{2\pi} (\sigma_i/\sqrt{N_i})} e^{-(x-\delta_i)^2/(2(\sigma_i^2/N_i))} \, dx = \int_{\delta_i/(\sigma_i/\sqrt{N_i})}^{\infty} \frac{1}{\sqrt{2\pi}} e^{-t^2/2} \, dt = \Phi\left(\frac{-\delta_i}{\sigma_i/\sqrt{N_i}}\right),$$

and for $i \notin S_m$,

$$P(\bar{J}_i \geq c) = \int_{0}^{\infty} \frac{1}{\sqrt{2\pi} (\sigma_i/\sqrt{N_i})} e^{-(x-\delta_i)^2/(2(\sigma_i^2/N_i))} \, dx = \Phi\left(\frac{\delta_i}{\sigma_i/\sqrt{N_i}}\right).$$

Now let $F$ be the Lagrangian relaxation of (5), with Lagrange multiplier $\lambda$:

$$F = \prod_{i \in S_m} P\{\bar{J}_i \leq c\} \prod_{i \notin S_m} P\{\bar{J}_i \geq c\} - \lambda \left(\sum_{i=1}^{k} N_i - T\right)$$

$$= \prod_{i \in S_m} \Phi\left(\frac{-\delta_i}{\sigma_i/\sqrt{N_i}}\right) \prod_{i \notin S_m} \Phi\left(\frac{\delta_i}{\sigma_i/\sqrt{N_i}}\right) - \lambda \left(\sum_{i=1}^{k} N_i - T\right).$$

Furthermore, the Karush-Kuhn-Tucker (KKT) (Walker 1999) conditions of this problem can be stated as follows:

For $i \in S_m$,

$$\frac{\partial F}{\partial N_i} = \prod_{j \in S_m, j \neq i} P\{\bar{J}_j \leq c\} \prod_{j \notin S_m} P\{\bar{J}_j \geq c\} \cdot \left[-\frac{1}{2} \varphi\left(\frac{\delta_i}{\sigma_i/\sqrt{N_i}}\right) \frac{\delta_i}{\sigma_i} \frac{N_i^{-1/2} - \lambda}{\delta_i N_i^{-1/2} - \lambda}\right] = 0.$$ \hspace{1cm} (6)

For $i \notin S_m$,

$$\frac{\partial F}{\partial N_i} = \prod_{r \in S_m, r \neq i} P\{\bar{J}_r \leq c\} \prod_{r \notin S_m} P\{\bar{J}_r \geq c\} \cdot \left[\frac{1}{2} \varphi\left(\frac{\delta_i}{\sigma_i/\sqrt{N_i}}\right) \frac{\delta_i}{\sigma_i} \frac{N_i^{-1/2} - \lambda}{\delta_i N_i^{-1/2} - \lambda}\right] = 0.$$ \hspace{1cm} (7)

Also, $\frac{\partial F}{\partial \lambda} = 0$ returns the budget constraint $\sum_{i=1}^{k} N_i - T = 0$.

To examine the relationship between $N_i$ and $N_j$ for $i \neq j$, we consider three cases:

Case 1. $i \in S_m$ and $j \notin S_m$. Equating the expressions in Equations (6) and (7),

$$\prod_{r \in S_m, r \neq i} P\{\bar{J}_r \leq c\} \prod_{r \notin S_m} P\{\bar{J}_r \geq c\} \cdot \left[-\frac{1}{2} \varphi\left(\frac{\delta_i}{\sigma_i/\sqrt{N_i}}\right) \frac{\delta_i}{\sigma_i} \frac{N_i^{-1/2} - \lambda}{\delta_i N_i^{-1/2} - \lambda}\right] = \prod_{r \in S_m, r \neq j} P\{\bar{J}_r \leq c\} \prod_{r \notin S_m} P\{\bar{J}_r \geq c\} \cdot \left[\frac{1}{2} \varphi\left(\frac{\delta_j}{\sigma_j/\sqrt{N_j}}\right) \frac{\delta_j}{\sigma_j} \frac{N_j^{-1/2} - \lambda}{\delta_j N_j^{-1/2} - \lambda}\right].$$

Simplifying,

$$P\{\bar{J}_j \geq c\} \cdot e^{-\delta_i^2/(2(N_i \sigma_i^2))} \frac{\delta_i}{\sigma_i} N_i^{-1/2} = P\{\bar{J}_j \leq c\} \cdot e^{-\delta_j^2/(2(N_j \sigma_j^2))} \frac{\delta_j}{\sigma_j} N_j^{-1/2}.$$ \hspace{1cm} (8)

Taking the log on both sides,

$$\log(P\{\bar{J}_j \geq c\}) - \log(P\{\bar{J}_j \leq c\}) = -\frac{\delta_i^2 N_i}{2\sigma_i^2} + \log\left(\frac{\delta_i}{\sigma_i}\right) - \frac{1}{2} \log(N_i)$$

$$= \log(P\{\bar{J}_j \leq c\}) - \log(P\{\bar{J}_j \geq c\}) = -\frac{\delta_j^2 N_j}{2\sigma_j^2} + \log\left(\frac{\delta_j}{\sigma_j}\right) - \frac{1}{2} \log(N_j).$$ \hspace{1cm} (9)

Now, we consider the asymptotic limit $T \to \infty$ with $N_i = \alpha_i T$, $\sum_{i=1}^{k} \alpha_i = 1$. Substituting for $N_i$ and dividing by $T$ yields

$$\frac{1}{T} \log(P\{\bar{J}_j \geq c\}) - \frac{\delta_i^2 N_i}{2\sigma_i^2} + \log\left(\frac{\delta_i}{\sigma_i}\right) - \frac{1}{2T} \log(\alpha_i T)$$

$$= \frac{1}{T} \log(P\{\bar{J}_j \leq c\}) - \frac{\delta_j^2 N_j}{2\sigma_j^2} + \log\left(\frac{\delta_j}{\sigma_j}\right) - \frac{1}{2T} \log(\alpha_j T),$$ \hspace{1cm} (10)
and then taking $T \to \infty$ yields
\[
\frac{\delta_i^2}{\sigma_i^2} = \frac{\delta_j^2}{\sigma_j^2} = \frac{\alpha_i}{\alpha_j}.
\]

Therefore, we obtain the ratio between $\alpha_i$ and $\alpha_j$ or between $N_i$ and $N_j$ as
\[
\frac{N_j}{N_i} = \frac{\alpha_j}{\alpha_i} = \left( \frac{\sigma_i/\delta_i}{\sigma_j/\delta_j} \right)^2.
\]

Because $\delta_i = \bar{J}_i - c$, $i = 1, 2, \ldots, k$, it is clear that the optimal allocation depends on the value of $c$ (see §3.3 for determining a good value), and thus the $m = 1$ case does not in general reduce to the original OCBA allocation for selecting the best design.

Case 2. Both $i, j \in S_m$ and $i \neq j$. From Equation (6), $\partial F/\partial N_i = \partial F/\partial N_j = 0$ yields
\[
\prod_{r \in S_m} P[\bar{J}_r \leq c] \cdot \prod_{r \neq i} P[\bar{J}_r \geq c]
\]
\[
\cdot \frac{-1}{2} \varphi \left( \frac{\delta_i}{\sigma_i/\sqrt{N_i}} \right) \frac{\delta_i}{(\sigma_i^2/N_i)^{1/2}} - \lambda
\]
\[
= \prod_{r \in S_m} P[\bar{J}_r \leq c] \cdot \prod_{r \neq j} P[\bar{J}_r \geq c]
\]
\[
\cdot \frac{-1}{2} \varphi \left( \frac{\delta_j}{\sigma_j/\sqrt{N_j}} \right) \frac{\delta_j}{(\sigma_j^2/N_j)^{1/2}} - \lambda.
\]

Then,
\[
P[\bar{J}_i \leq c] \cdot e^{-\delta_i^2/(2\sigma_i^2/N_i)} \frac{-\delta_i}{\sigma_i} N_i^{-1/2}
\]
\[
= P[\bar{J}_i \leq c] \cdot e^{-\delta_i^2/(2\sigma_i^2/N_i)} \frac{-\delta_i}{\sigma_i} N_i^{-1/2}.
\]

Following the analogous derivation that led to Equation (8) yields the same result.

Case 3. $i, j \in S_m$, and $i \neq j$. Again, following the same derivation procedures as in the previous two cases leads to Equation (8) again, so it holds for any $i, j \in \{1, 2, \ldots, k\}$, and $i \neq j$.

In conclusion, if a solution satisfies Equation (8), then the KKT sufficient conditions must hold asymptotically, so that the corresponding solution is a locally optimal solution to the Lagrangian relaxation of the OCBA problem (5). We therefore have the following result.

**Theorem 1.** The allocation given by (8) is asymptotically (as $T \to \infty$) a locally optimal solution for the OCBA problem (5), where $\delta_i = \bar{J}_i - c$ for $c$ a constant, and the variances $\sigma_1^2, \sigma_2^2, \ldots, \sigma_k^2$ are finite; i.e., APCSm is asymptotically maximized by the allocation given by (8).

### 3.3. Determination of $c$ Value

The parameter $c$ impacts the quality of the approximation APCSm to $P[CS]$. Because APCSm is a lower bound of $P[CS]$, choosing $c$ to make APCSm as large as possible is likely to provide a better approximation of APCSm to $P[CS]$. Figure 1 is provided to help explain our choice of $c$ by giving an example of probability density functions for $\bar{J}_i$, $i = 1, 2, \ldots, k$.

Note that APCSm is a product of $P[\bar{J}_i \leq c]$ for $i \in S_m$ and $P[\bar{J}_i \geq c]$ for $i \notin S_m$. Consider the equal variance case $\text{Var}(\bar{J}_i) = \text{Var}(\bar{J}_j) = \cdots = \text{Var}(\bar{J}_k)$, where for any value of $c$, $P[\bar{J}_i \leq c] \times P[\bar{J}_i \leq c] \times \cdots \times P[\bar{J}_i \leq c]$, and $P[\bar{J}_i \leq c] < P[\bar{J}_i \leq c] < \cdots < P[\bar{J}_i \leq c]$. To prevent APCSm from being small, we want to choose $c$ to avoid any of the product terms being too small, especially $P[\bar{J}_i \leq c]$ and $P[\bar{J}_i \leq c]$, because one of these two terms will be the smallest in the product, depending on the value of $c$. Thus, a good choice of $c$ lies between $\bar{J}_m$ and $\bar{J}_{m+1}$ because

(i) if $c = c' < \bar{J}_m$, then $P[\bar{J}_i < c'] < 0.5$, and this term decreases with decreasing $c'$, resulting in a negative impact on APCSm;

(ii) similarly, if $c = c'' > \bar{J}_{m+1}$, then $P[\bar{J}_i > c''] < 0.5$, and this term decreases with increasing $c''$.

With these considerations, one would like to maximize both $P[\bar{J}_i \leq c]$ and $P[\bar{J}_i \leq c]$. In this paper, we choose to maximize the product of $P[\bar{J}_i \leq c]$ and $P[\bar{J}_i \leq c]$. Define $\tilde{\sigma}_i \equiv \sigma_i/\sqrt{N_i}$. Then,
\[
P[\bar{J}_i \leq c] \cdot P[\bar{J}_i \leq c] = \Phi \left( \frac{c - \bar{J}_m}{\tilde{\sigma}_i} \right) \Phi \left( \frac{\bar{J}_{m+1} - c}{\tilde{\sigma}_{i+1}} \right).
\]

Following the same approach as used to establish Theorem 1, this quantity is asymptotically maximized when
\[
c = \frac{\tilde{\sigma}_{m+1} \bar{J}_m + \tilde{\sigma}_m \bar{J}_{m+1}}{\tilde{\sigma}_m + \tilde{\sigma}_{m+1}},
\]
and we use this value of $c$ in our implementation, which in numerical testing results in good performance while requiring negligible computation cost.

### 3.4. Heuristic Sequential Allocation Scheme

The allocation given by (8) assumes known variances and independence of estimated sample means across designs. In practice, a sequential algorithm is used to estimate these quantities using the updated sample variances. Furthermore, the “constant” $c$ and sample means are also updated during each iteration. Each
design is initially simulated with \( n_0 \) replications in the first stage, and additional replications are allocated incrementally with \( \Delta \) replications to be allocated in each iteration. By utilizing the allocation given in Theorem 1, we present the following heuristic sequential algorithm.

**OCBA-\( m \) Allocation Procedure**

**INPUT** \( k, m, T, \Delta, n_0 \) (\( T - kn_0 \) a multiple of \( \Delta \) and \( n_0 \geq 5 \));

**initialize** \( l \leftarrow 0 \);

**Perform** \( n_0 \) simulation replications for all designs; \( N_1' = N_2' = \cdots = N_k' = n_0 \).

**Loop while** \( \sum_{j=1}^{k} N_j' < T \) **do**

**Update** Calculate sample means \( \bar{J}_i = (1/N_i') \sum_{j=1}^{N_i'} X_{ij} \), and sample standard deviation \( s_i = \sqrt{(1/(N_i') - 1) \sum_{j=1}^{N_i'} (X_{ij} - \bar{J}_i)^2} \), \( i = 1, \ldots, k \), using the new simulation output; compute \( \bar{s}_i = s_i/\sqrt{N_i'} \), \( i = 1, \ldots, k \), and \( c = (\bar{s}_i' \bar{J}_i' + \bar{s}_i \bar{J}_i)/ (\bar{s}_i' + \bar{s}_i) \); update \( \delta_i = \bar{J}_i - c \), \( i = 1, \ldots, k \).

**Allocate** Increase the computing budget by \( \Delta \) and calculate the new budget allocation, \( N_1'^{+1}, N_2'^{+1}, \ldots, N_k'^{+1} \), according to

\[
\frac{N_1'^{+1}}{(s_1/\delta_1)2} = \frac{N_2'^{+1}}{(s_2/\delta_2)2} = \cdots = \frac{N_k'^{+1}}{(s_k/\delta_k)2}. \tag{10}
\]

**Simulate** Perform additional \( \max(N_i'^{+1} - N_i', 0) \) simulations for design \( i \), \( i = 1, \ldots, k \); \( l \leftarrow l + 1 \).

**End of loop**

The approximations made in this OCBA-\( m \) procedure are further discussed below.

**3.4.1. Variances.** The allocation given in Theorem 1 assumes known variances. The above sequential algorithm estimates these quantities using the updated sample variances. As more simulation replications are iteratively allocated to each design, the variance estimation improves. To avoid poor estimation at the beginning, \( n_0 \) should not be too small (we suggest \( n_0 \geq 5 \)). Also, it is wise to avoid large \( \Delta \) (we suggest \( \Delta < 100 \)) to prevent a poor allocation before a correction can be made in the next iteration, which is particularly important in the early stages. Our numerical testing indicates that the performance of the OCBA-\( m \) procedure is not sensitive to the choice of \( n_0 \) and \( \Delta \) if these guidelines are followed, and the impact of approximating variance by sample variance is not significant.

**3.4.2. Sequential Allocation.** The OCBA-\( m \) procedure is a sequential algorithm. The sequential nature of the estimation introduces dependence in the sample estimates themselves so that the independence across designs assumed in (4) does not hold. However, the sequential approach is an effective way to estimate the unknown variance, and several studies have demonstrated significant efficiency gains in using sequential allocation versus one-time or two-stage allocation (cf. Inoue et al. 1999, Chen et al. 2006).

**3.4.3. Asymptotically Large Computing Budget.** Although the allocation given by Equation (8) in Theorem 1 is derived by taking \( T \to \infty \), the numerical results presented in the next section indicate that the corresponding allocation in the OCBA-\( m \) procedure works very efficiently for small \( T \), as well.

**3.4.4. Continuous \( N_i \).** The resulting \( N_i \) in the ALLOCATE step based on Equation (8) is a continuous number that must be rounded to an integer. In the numerical experiments in the next section, \( N_i \) is rounded to the nearest integer such that the summation of additional simulation replications for all designs equals \( \Delta \). This is simply for ease of computing budget management in numerical testing, providing a fair comparison with other allocation procedures. We have found numerically that the OCBA-\( m \) performance is not sensitive to how we round \( N_i \), probably due to the robustness of a sequential procedure.

**3.4.5. Determination of \( c \) Value.** The parameter \( c \) impacts the quality of the approximation APCS\( m \) to \( P[CS] \). Section 3.3 provides a simple approach to determine \( c \) by maximizing the product of \( P[\bar{J}_i \leq c] \) and \( P[\bar{J}_{i+1} \geq c] \), which are the critical terms for most cases and provide a proxy for maximizing APCS\( m \). For the purpose of determining the computing budget allocation, our numerical testing shows that such a simple approach performs very well. However, for the purpose of estimating the probability of correct selection, one would want to choose \( c \) by considering more than these two terms or even all the terms in APCS\( m \).

Note that this OCBA-\( m \) procedure is designed to select all of the top-\( m \) designs when \( m \geq 2 \). For the \( m = 1 \) case, the original OCBA procedure given in Chen et al. (2000) is different from the OCBA-\( m \) due to different approximations made. In this case, the original OCBA procedure is slightly preferred, even though this OCBA-\( m \) procedure still works effectively.

**4. Numerical Testing and Comparison with Other Allocation Procedures**

In this section, we test the OCBA-\( m \) algorithm by comparing it on several numerical experiments with different allocation procedures: equal allocation, which simulates all design alternatives equally; the Koenig and Law (1985) procedure denoted by KL;
proportional to variance (PTV), which is a modification of KL that allocates replications proportional to the estimated variances; and the original OCBA allocation algorithm for selecting only the best design (Chen et al. 2000). For notational simplicity, we assume that \( J_{[1]} < J_{[2]} < \cdots < J_{[m]} \), so design \([1]\) is the best, and correct selection would be \( S_m = \{[1], [2], \ldots, [m]\} \) (but this is unknown a priori).

We also test the OCBA-\(m\) algorithm under the simulation optimization setting, in which OCBA-\(m\) is integrated with three different optimization methods. The performance of each of the three methods integrating the OCBA-\(m\) allocation is compared with the performance of the same method using equal allocation.

4.1. Computing Budget Allocation Procedures

4.1.1. Equal Allocation. The simulation budget is allocated equally to all designs; i.e., \( N_i = T/k \) for each \( i \). The performance of equal allocation will serve as a benchmark for comparison.

4.1.2. KL (Koenig and Law 1985). The two-stage procedure of Koenig and Law (1985) selects a subset of specified size \( m \), with probability at least \( P^* \), so that the selected subset is exactly the actual subset with the best (smallest) expected values, provided that \( J_{[m+1]} - J_{[m]} \) is no less than an indifference zone, \( d \). As in our setting, the ordering within the selected subset does not matter.

In the first stage, all designs are simulated for \( n_i \) replications. Based on the sample variance estimate \( \hat{s}_i^2 \) obtained from the first stage and given the desired correct selection probability \( P^* \), the number of additional simulation replications for each design in the second stage is determined by

\[
N_i = \max(n_i + 1, \lceil h_3(\hat{s}_i^2/d^2) \rceil) \quad \text{for } i = 1, 2, \ldots, k,
\]

where \( \lceil \cdot \rceil \) is the integer “round-up” function, and \( h_3 \) is a constant that depends on \( k, m, P^* \), and \( n_i \).

4.1.3. Proportional to Variance (PTV). This is a sequential modified version of the KL procedure, based on the observation that (11) implies that \( N_i \) is proportional to the estimated sample variance \( s_i^2 \). Thus, the PTV procedure sequentially determines \( \{N_i\} \) based on the newly updated sample variances by replacing Equation (10) in the ALLOCATE step of the OCBA-\(m\) algorithm by

\[
\frac{N_i^{i+1}}{s_i^2} = \frac{N_1^{i+1}}{s_1^2} = \cdots = \frac{N_k^{i+1}}{s_k^2}.
\]

Thus, the number of replications for each design grows in proportion to the sample variance. Note that the indifference-zone parameter has been removed in this modification to make it comparable to the other procedures.

4.1.4. OCBA (Chen et al. 2000). The original sequential OCBA procedure of Chen et al. (2000) allocates the computing budget with the objective of selecting only the best design, i.e., \( m = 1 \), for which extensive numerical testing has demonstrated its efficiency. Although it is not designed for \( m > 1 \), we test this procedure here for benchmarking purposes, and denote it by OCBA. Specifically, the budget allocation in Equation (10) of the OCBA-\(m\) algorithm is replaced by

\[
(1) \quad N_i^{i+1} = (s_i(\bar{j}_b - j_i)/s_i(\bar{j}_b - j_b))^2 \quad \text{for all } i \neq j \neq b,
\]

\[
(2) \quad N_b^{i+1} = s_b\sqrt{\sum_{i=1}^{k} \left( N_i^{i+1}/s_i \right)^2},
\]

where \( b = \arg \min \bar{j}_i \).

4.2. Numerical Results for Different Allocation Procedures

To compare the performance of the procedures, we carried out numerical experiments for several typical selection problems. In comparing the procedures, the measurement of effectiveness used is the \( P[CS] \) estimated by the fraction of times the procedure successfully finds all the true \( m \)-best designs out of 100,000 independent experiments. Because this penalizes incorrect selections equally—e.g., a subset containing the top-1, top-2, \ldots, and top-(\(m-1\)) designs and missing only the top-\(m\) design is treated no differently than a subset containing not a single one of the top-\(m\) designs—in our numerical experiments, we also include a second measure of selection quality, the so-called expected opportunity cost \( E[OC] \), where

\[
OC \equiv \sum_{j=1}^{m} (j_i - \bar{j}_i).
\]

This measure penalizes particularly bad choices more than mildly bad choices. For example, when \( m = 3 \), a selection of \{top-1, top-2, top-4\} is better than \{top-1, top-2, top-5\}, and both are better than \{top-1, top-3, top-5\}. Note that OC returns a minimum value of zero when all the top-\(m\) designs are correctly selected. The estimated \( E[OC] \) is the average of the OC estimates over the 100,000 independent experiments.

Each of the procedures simulates each of the \( k \) designs for \( n_i = 20 \) replications initially (following recommendations in Koenig and Law 1985, Law and Kelton 2000). KL allocates additional replications in a second stage (so the total number is not fixed a priori), whereas the other procedures allocate replications incrementally by \( \Delta = 50 \) each time until the total budget, \( T \), is consumed. For each level of computing budget, we estimate the achieved \( P[CS] \) and \( E[OC] \).

Because KL is a two-stage indifference-zone procedure, we must specify the values for the desired probability of correct selection, \( P^* \), and the indifference
zone $d$ to satisfy the condition that $I_{m+1} - I_m \geq d$, where a smaller $d$ implies a higher required computation cost based on Equation (10). In practice, the value of $I_{m+1}$ or $I_m$ is unknown beforehand, but for benchmarking purposes, we set $d = I_{m+1} - I_m$, which leads to the minimum computational requirement (or maximum efficiency) for the procedure. As is done for the other procedures, the resulting $P[CS]$ and $E[OC]$ can be estimated over the 100,000 independent experiments. Because the required computation cost also varies from one experiment to another, we will indicate the average number of total replications based on the 100,000 independent experiments.

Example 1 (Equal Variance). There are 10 alternative designs, with distribution $N(i, i^2)$ for design $i = 1, 2, \ldots, 10$. The goal is to identify the top-3 designs via simulation samples, i.e., $m = 3$ in this example.

To characterize the performance of different procedures as a function of $T$, we vary $T$ between 200 and 8,000 for all of the procedures other than KL, and the estimated achieved $P[CS]$ and $E[OC]$ as a function of $T$ are shown in Figures 2(a) and 2(b), respectively. For KL, we test two cases, $P^* = 0.9$ and $P^* = 0.95$, and the corresponding estimated $P[CS]$ and $E[OC]$ versus the average total simulation replications are shown as two single points (the triangle and circle) in Figures 2(a) and 2(b), respectively.

We see that all procedures obtain a higher $P[CS]$ and a lower $E[OC]$ as the available computing budget increases. However, OCBA-$m$ achieves the highest $P[CS]$ and the lowest $E[OC]$ for the same amount of computing budget. It is interesting to observe that OCBA, which performs significantly better than equal allocation and PTV when the objective is to find the single best design, fares worse in this example than these two allocations when the objective is changed to finding all the top-3 designs. Equal allocation performs almost identically to PTV, which makes sense because the variance is constant across designs. Specifically, the computation costs to attain $P[CS] = 0.95$ for OCBA-$m$, OCBA, Equal, and PTV are 800, 3,200, 1,950, and 2,000, respectively.

Not surprisingly, the performance of KL is along the performance curve of PTV because KL basically allocates the computing budget based on design variance. However, KL achieves a substantially higher $P[CS]$ than the desired level (e.g., exceeding 0.99 for the target minimum of $P^* = 0.9$) by spending a much higher computing budget than actually needed, consistent with the fact that typical two-stage indifference-zone procedures are conservative.

Example 2 (Variance Increasing in Value of Mean). This is a variant of Example 1. All settings are preserved except that the variance is increasing in the design index, so good designs have smaller variances. Specifically, the designs are distributed $N(i, i^2)$ for design $i = 1, 2, \ldots, 10$. Again, $m = 3$.

The test results shown in Figures 3(a) and 3(b) are qualitatively similar to those in Example 1. OCBA-$m$ achieves the highest $P[CS]$ for the same amount of computing budget. However, PTV (and KL) performs poorly in this example because good designs receive relatively less computing budget due to their smaller variances, which tends to slow down the process of distinguishing good designs. Specifically, the computation costs to attain $P[CS] = 0.95$ for OCBA-$m$, OCBA, Equal, and PTV are 350, 750, 700, and 2,250, respectively.

Example 3 (Variance Decreasing in Value of Mean). The third example is another variant of Examples 1 and 2, but this time the variance is decreasing in the design index; i.e., the distribution is $N(i, (11 - i)^2)$ for design $i = 1, 2, \ldots, 10$. Under this setting, good designs have larger variance. Again, $m = 3$.

The test results shown in Figures 4(a) and 4(b) are similar to those in the previous examples, with again OCBA-$m$ performing the best. However, in contrast to Example 2, PTV (and KL) performs relatively well in this example because good designs receive much more computing budget due to their higher variances. On the other hand, OCBA performs poorly because it spends an excessive amount of the computing budget to distinguish between the very top designs because its objective is to find the best. In this example, the computation costs to attain $P[CS] = 0.95$ for OCBA-$m$, OCBA, Equal, and PTV are 1,400, 7,900, 3,050, and 2,200, respectively.

Example 4 ((s, S) Inventory Problem). The fourth example is an (s, S) inventory policy problem based on the example given in §1.5.1 of Law and Kelton (2000). The system involves a single item under periodic review, full backlogging, and random lead times (uniformly distributed between the 0.5 and 1.0 period), with costs for ordering (including a fixed setup cost of $32 per order and an incremental cost of $3 per item), on-hand inventory ($1 per item per period), and backlogging (fixed shortage cost of $5 per item per period). The times between demands are independent and identically distributed (i.i.d.) exponential random variables with a mean of 0.1 period. The sizes of demands are i.i.d. random variables taking values 1, 2, 3, and 4, with probabilities 1/6, 1/3, 1/3, and 1/6, respectively. The (s, S) policy specifies that if the on-hand inventory at the review point is at or below the level $s$, then an order is placed of an amount that would bring the inventory up to level $S$. The 10 inventory policies are defined by the parameters $(s_1, s_2, \ldots, s_{10}) = (20, 20, 20, 40, 40, 40, 60, 60, 60, 80)$ and $(s_1, s_2, \ldots, s_{10}) = (30, 40, 50, 50, 60, 70, 70, 80, 90, 90)$, respectively. The objective is to find the top-3 ($m = 3$) policies with minimum expected average inventory cost over 120 periods.
Figure 2(a) \( P(CS) \) vs. \( T \) Using Four Sequential Allocation Procedures and KL (Triangle for \( P^* = 90\% \) and Circle for \( P^* = 95\% \)) for Example 1

Figure 2(b) \( 
\mathbb{E}[OC] \) vs. \( T \) Using Four Sequential Allocation Procedures and KL (Triangle for \( P^* = 90\% \) and Circle for \( P^* = 95\% \)) for Example 1
Figure 3(a) $P(\text{CS})$ vs. $T$ Using Four Sequential Allocation Procedures and KL (Triangle for $P^* = 90\%$ and Circle for $P^* = 95\%$) for Example 2

Figure 3(b) $\mathbb{E}(\text{OC})$ vs. $T$ Using Four Sequential Allocation Procedures and KL (Triangle for $P^* = 90\%$ and Circle for $P^* = 95\%$) for Example 2
Figure 4(a) \( P(\text{CS}) \) vs. \( T \) Using Four Sequential Allocation Procedures and KL (Triangle for \( P^* = 90\% \) and Circle for \( P^* = 95\% \)) for Example 3

Figure 4(b) \( E(\text{OC}) \) vs. \( T \) Using Four Sequential Allocation Procedures and KL (Triangle for \( P^* = 90\% \) and Circle for \( P^* = 95\% \)) for Example 3
Figure 5(a) \( P(\text{CS}) \) vs. \( T \) Using Four Sequential Allocation Procedures and KL (Triangle for \( P^* = 90\% \) and Circle for \( P^* = 95\% \)) for Example 4

Figure 5(b) \( E(\text{OC}) \) vs. \( T \) Using Four Sequential Allocation Procedures and KL (Triangle for \( P^* = 90\% \) and Circle for \( P^* = 95\% \)) for Example 4
The test results shown in Figures 5(a) and 5(b) are similar to those in previous examples, in that OCBA-\(m\) is clearly the top performer again; however, this time, OCBA is the runner-up by a slight margin. The computation costs to attain \(P[CS] = 0.95\) for OCBA-\(m\), OCBA, Equal, and PTV are 500, 1,200, 1,650, and 1,350, respectively.

**Example 5 (Larger-Scale Problem).** This is a variant of Example 1 (constant variance), with the number of designs increased to 50. The alternatives have distribution \(N(i, 10^2)\) for design \(i = 1, 2, \ldots, 50\), and \(m = 5\). Because KL’s performance basically follows that of PTV, but its required computing budget is far beyond the range we are considering here, we exclude KL from the numerical testing.

Figures 6(a) and 6(b) depict the simulation results. As in earlier examples, OCBA-\(m\) achieves the highest \(P[CS]\) and the lowest \(E[OC]\) with the same amount of computing budget; however, the performance gap between OCBA-\(m\) and other procedures is substantially greater. This is because a larger design space allows the OCBA-\(m\) algorithm more flexibility in allocating the computing budget, resulting in even better performance. On the other hand, OCBA performs poorly because it spends a lot of computing budget on distinguishing the very top designs, a tendency that is penalized even more for larger \(m\). Again, because the variance is constant across designs, the performance of Equal and PTV are nearly indistinguishable. In this example, the computation costs to attain \(P[CS] = 0.95\) for OCBA-\(m\), OCBA, Equal, and PTV are 4,050, 31,050, 27,050, and 27,200, respectively.

### 4.3. Numerical Results for Simulation Optimization

In these numerical examples, we combine the OCBA-\(m\) allocation procedure with three iterative optimization search algorithms, requiring the selection of an “elite” subset of good candidate solutions in each iteration. The OCBA-\(m\) procedure is integrated into each of the three optimization algorithms, and the resulting performance of the algorithm is compared with the same algorithm using equal simulation of all candidate solutions. The purpose of these examples is not to find the best optimization search algorithm, but rather to explore whether the OCBA-\(m\) procedure can enhance the efficiency of any simulation optimization algorithm. The three optimization algorithms considered are the following.

**4.3.1. Cross-Entropy Method.** The cross-entropy (CE) method (see Rubinstein and Kroese 2004) searches the underlying variable space by adaptively updating a parameterized sampling distribution. The basic principle is to minimize the Kullback-Leibler divergence between the unknown optimal sampling distribution and the parameterized distribution. The algorithm is summarized as follows.

**Step 1.** Initialize a sampling distribution \(P\).
**Step 2.** Sample \(k\) candidate solutions using \(P\).
**Step 3.** Simulate these sampled \(k\) alternatives and select a subset containing the top-\(m\).
**Step 4.** Update \(P\) based on the selected top-\(m\) and the CE principle.
**Step 5.** Go back to Step 2 if the stopping criterion is not met.

### 4.3.2. Population-Based Incremental Learning (PBIL).

The PBIL algorithm was originally developed for binary search problems (Baluja 1994). We test one of its new developments for continuous optimization problems due to Rudlof and Köppen (1996). The PBIL updates the sampling distribution \(P\) with a probabilistic learning technique using the estimated mean of an “elite” subset of good candidate solutions in each iteration. This algorithm is almost the same as the CE algorithm, except in Step 4, the PBIL learning principle is applied.

### 4.3.3. Neighborhood Random Search (NRS).

We also test a simple random search method. In each iteration, \(k\) alternative designs are simulated and then the top-\(m\) solutions are selected. For the next iteration, a large proportion of candidate solutions are generated by sampling the neighborhood of the elite solutions. The longer the distance, the smaller the probability a solution is sampled. The remaining (smaller) portion of samples are taken from the entire variable space to ensure convergence to the global optimum. The algorithm is summarized as follows.

**Step 1.** Uniformly sample \(k\) candidate solutions over the design variable space.
**Step 2.** Stop, if the stopping criterion is met.
**Step 3.** Simulate the sampled \(k\) alternatives and select a subset containing the top \(m\).
**Step 4.** Sample 80% of the new \(k\) candidate solutions in the neighborhood of the top-\(m\) elite solutions. Another 20% is taken uniformly from the entire variable space.
**Step 5.** Go back to Step 2.

In each of the above three optimization algorithms, Step 3 is a ranking-and-selection problem in which the top-\(m\) designs must be identified. The overall efficiency of these types of simulation optimization algorithms depends on how efficiently we simulate the candidates and correctly select the top-\(m\) designs.

**Example 6 (Griewank Function).** The Griewank function is a common example in the global optimization literature (see Fu et al. 2006), given in two-dimensional (2-D) form by

\[
f(x_1, x_2) = \frac{1}{40} (x_1^2 + x_2^2) - \cos(x_1) \cos\left(\frac{x_2}{\sqrt{2}}\right) + 1,
\]

where \(x_1\) and \(x_2\) are continuous variables and \(-10 \leq x_1, x_2 \leq 10\). The unique global minimum...
Figure 6(a) $P(\text{CS})$ vs. $T$ Using Four Sequential Allocation Procedures for Example 5

Figure 6(b) $E(\text{OC})$ vs. $T$ Using Four Sequential Allocation Procedures for Example 5
of this function is at \((x_1^*, x_2^*) = (0, 0)\) and \(f(x_1^*, x_2^*) = 0.\) The additive noise incurred in stochastic simulation is \(N(0, 1^2)\). Figure 7 gives an illustration of this function without noise.

In numerical implementation, the stopping criterion for the ranking-and-selection problem in Step 3 is when the posterior APCS given by Equation (4) is no less than \(1 - 0.2 \times \exp(-q/50)\), where \(q\) is the iteration number. We set \(k = 100\) and \(m = 5\). In comparing the procedures, the measurement of effectiveness used is the average error between the best design thus far and the true optimal solution over 200 independent experiments. The results are shown in Figure 8. The thick lines indicate the performance with OCBA-\(m\) for different optimization algorithms, whereas the thin lines show the performances without OCBA-\(m\). Lines with different shades/patterns represent the use of different optimization search algorithms.

We see that the optimality gap decreases for all procedures as the available computing budget increases. In this example, NRS performs better than PBIL, which does better than CE. However, OCBA-\(m\) significantly enhances the efficiency for all three search methods. For example, with integration of OCBA-\(m\), NRS can achieve an average error of 0.1 using a computation cost of 22,800. Without OCBA-\(m\), NRS spends a computation cost of 68,500 to achieve the same level of error. Similarly, the computation costs for PBIL to reduce the average error to 0.12 with and without OCBA-\(m\) are 11,500 and 53,700, respectively. The speedup factor of using OCBA-\(m\) is even larger if the target level of optimality gets higher.

**Example 7 (Rosenbrock Function).** The Rosenbrock function is another common example in the global optimization literature (e.g., Fu et al. 2006). It is a nonconvex function with a “banana-shaped” valley given in 2-D by

\[
f(x_1, x_2) = 100(x_2 - x_1^2)^2 + (x_1 - 1)^2,
\]

where \(x_1\) and \(x_2\) are continuous variables and \(-5 \leq x_1 \leq 5, -5 \leq x_2 \leq 5\). The global minimum of this function is at \((x_1^*, x_2^*) = (1, 1)\) and \(f(x_1^*, x_2^*) = 0.\) The additive noise incurred in stochastic simulation is \(N(0, 1^2)\).
5. Conclusions

We present an efficient allocation procedure for a class of ranking-and-selection problems in which the objective is to identify the top-\(m\) designs out of \(k\) (simulated) competing designs. The goal is to maximize the simulation efficiency, expressed as the probability of correct selection within a given computing budget. We propose a heuristic to approximate the associated correct selection probability, and then derive an asymptotically optimal allocation procedure for an approximation to the approximate probability. Numerical testing indicates that the allocation procedure is significantly more efficient and robust than other methods in the literature, with the relative efficiency increasing in problem size. Furthermore, although the procedure was derived based on an asymptotic derivation, the numerical results indicate that the procedure is effective even when the computing budget is small. The numerical results illustrate that the allocation specified by the original OCBA algorithm (Chen et al. 2000), designed for selecting the single best design, does not perform well in selecting the top-\(m\) designs, providing another motivation for the need of a new methodology when the objective is extended beyond selecting just the best design. Our allocation method has the potential to improve the efficiency of population-based global optimiza-

Figure 9 2-D Rosenbrock Function Tested in Example 7

Figure 9 gives an illustration of this function without noise.

The numerical setting is the same as that in Example 6. The test results are shown in Figure 10. Unlike Example 6, the PBIL method has the best performance in this example. Although the order of optimization methods are different from that in Example 6, the level of efficiency enhancement using OCBA-\(m\) is very similar.

Figure 10 Performance Comparison for Three Optimization Search Algorithms With and Without OCBA-\(m\) for Example 7
tion methods that require the selection of an “elite” subset of good candidate solutions in each iteration of the algorithm. While different optimization algorithms perform differently in our two preliminary numerical examples, the new OCBA-$m$ allocation significantly enhances the computational efficiency for each individual optimization algorithm.

Acknowledgments

This work was supported in part by the National Science Foundation Grants IIS-0325074, DMI-0540312, and DMI-0323220, by NASA Ames Research Center Grant NNA05CV26G, by the Federal Aviation Administration Grant 00-G-016, and by Air Force Office of Scientific Research Grants FA9550-04-1-0210 and FA9550-07-1-0366.

References


