Published online ahead of print October 17, 2011

INFORMS Journal on Computing
Articles in Advance, pp. 1–14
ISSN 1091-9856 | EISSN 1526-5528

http://dx.doi.org/10.1287/ijoc.1110.0481
© 2011 INFORMS

An Adaptive Hyperbox Algorithm for High-Dimensional Discrete Optimization via Simulation Problems

Jie Xu
Department of Systems Engineering and Operations Research, George Mason University, Fairfax, Virginia 22030, jxu13@gmu.edu

Barry L. Nelson
Department of Industrial Engineering and Management Sciences, Northwestern University, Evanston, Illinois 60208, nelsonb@northwestern.edu

L. Jeff Hong
Department of Industrial Engineering and Logistics Management, Hong Kong University of Science and Technology, Clear Water Bay, Hong Kong, China, hongl@ust.hk

We propose an adaptive hyperbox algorithm (AHA), which is an instance of a locally convergent, random search algorithm for solving discrete optimization via simulation problems. Compared to the COMPASS algorithm, AHA is more efficient in high-dimensional problems. By analyzing models of the behavior of COMPASS and AHA, we show why COMPASS slows down significantly as dimension increases, whereas AHA is less affected. Both AHA and COMPASS can be used as the local search algorithm within the Industrial Strength COMPASS framework, which consists of a global search phase, a local search phase, and a final cleanup phase. We compare the performance of AHA to COMPASS within the framework of Industrial Strength COMPASS and as stand-alone algorithms. Numerical experiments demonstrate that AHA scales up well in high-dimensional problems and has similar performance to COMPASS in low-dimensional problems.

Key words: optimization via simulation; random search; ranking and selection
History: Accepted by Marvin Nakayama, Area Editor for Simulation; received February 2010; revised November 2010, June 2011; accepted June 2011. Published online in *Articles in Advance*.

1. Introduction

Both academic researchers and industrial practitioners have designed various algorithms for optimizing the performance of simulated systems. We refer to such problems as *optimization via simulation* (OvS). See Fu (2002) and Fu et al. (2005) for a review of OvS. In many applications, such as multiproduct inventory management, the system design variables are discrete valued, and thus the optimization problems are *discrete optimization via simulation* (DOvS) problems.

There have been a number of approaches to DOvS problems proposed in the research literature. When the number of feasible solutions is not too large (at most a few hundred), ranking-and-selection procedures attempt to choose the best solution by simulating all of them in such a way that a probability of correct selection is maximized or guaranteed. The indifference-zone procedure of Nelson et al. (2001), the Bayesian procedure of Chick and Inoue (2001), and the optimal budget allocation procedure of Chen et al. (2000) fall into this category. The ordinal optimization approach of Ho et al. (1992) relaxes the problem of finding the global optimum to instead finding one of the top-n solutions. This is done by first sampling k solutions from the set of feasible solutions, simulating them, and then selecting the best from those k solutions. The critical decision is choosing k such that at least one of the simulated solutions is a top-n solution.

When the solution space is large, adaptive random search has been the dominant paradigm for designing DOvS algorithms. Many algorithms achieve asymptotic convergence to the global optimum; these include the stochastic ruler algorithm of Yan and Mukai (1992), the simulated annealing algorithm of Alrefaei and Andradóttir (1999), the stochastic comparison algorithm of Gong et al. (1999), and the nested partitions algorithm of Shi and Ólafsson (2000) and Pichitlamken and Nelson (2003). However, to guarantee global convergence without any structural information about the problem, these algorithms essentially have to visit every solution; thus they do not scale up efficiently for practical problems with a large number of feasible solutions. More recently, Andradóttir and Prudius (2009) and Hu et al. (2007) designed two OvS algorithms that combine the efficiency of local search and the guarantee of converging to a global optimum by balancing the efficiency of intensive local search and the requirement to keep sampling solutions from the entire solution space for global convergence.

Another class of adaptive random search DOvS algorithms guarantee to converge to a locally optimal solution. Andradóttir (1995) proposed a locally convergent algorithm for one-dimensional DOvS problems. The COMPASS algorithm of Hong and Nelson (2006) is another instance of a locally convergent, random search algorithm, but it is not restricted to one dimension. By focusing on finding a local optimum, these algorithms can efficiently search the solution space and deliver good finite-time performance because they only need to visit a small fraction of the feasible solutions.

To bridge the divide between the academic community and industrial practitioners, Xu et al. (2010) proposed a framework for locally convergent DOvS algorithms and implemented it in a software package called Industrial Strength COMPASS (ISC). ISC solves linearly constrained DOvS problems with a finite solution space. The framework consists of three phases: global search, local search, and final cleanup. The global search phase explores the feasible solution space and identifies promising regions for intensive local search. The local search phase investigates these regions and may return multiple locally optimal solutions. The cleanup phase then selects the best among these local optima and estimates the objective value with controlled error. In ISC, a niching genetic algorithm performs the global phase, the local phase is COMPASS with constraint pruning, and the cleanup phase is a two-stage ranking-and-selection procedure. Numerical experiments show that ISC has finite-time performance comparable to or better than one of the best and most popular commercial optimization via simulation software products, OptQuest, for problems up to dimension 10. However, ISC dramatically slows down when dimensionality increases beyond 10.

The slowdown of ISC is due to COMPASS' behavior in higher-dimensional spaces: COMPASS closes in on a locally optimal solution by progressively adding linear constraints that define the most promising area for exploration. As dimension increases, the number of constraints that COMPASS needs to define the most promising area quickly increases. As a result, there is more algorithm overhead for sampling new solutions from the most promising area. To manage the number of constraints, COMPASS can employ constraint pruning. Constraint pruning involves solving linear programs (LPs), and it is needed much more frequently in high-dimensional problems. Although constraint pruning is time consuming, it is essential to keep COMPASS from slowing down in problems where it visits many solutions. Our primary goal in this paper is to identify alternative locally convergent algorithms without this shortcoming and thus make the ISC software applicable to higher-dimensional problems.

COMPASS and most other adaptive random search algorithms construct a most promising area (MPA) at each iteration and focus on sampling solutions within the MPA. We propose an adaptive hyperbox algorithm (AHA), where the MPA takes the form of a hyperbox. AHA has two advantages. First, it scales up well as problem dimension increases; we model the behavior of the algorithm and show why this is so. Using the model, we also provide insight into the impact of dimensionality on COMPASS, which sheds light on why COMPASS slows down as dimension increases. Second, AHA does not require a time-consuming constraint pruning step to identify the set of *active* solutions at each iteration (i.e., the solutions that define the MPA and thus need to be simulated at that iteration). The procedure to identify the set of active solutions is much faster for AHA.

We first briefly describe the COMPASS algorithm with constraint pruning in §2. In §3, we introduce AHA, prove its local convergence for fully constrained DOvS problems, and model its performance. We then report experiment results in §4, and we conclude the paper in §5. Proofs for the convergence property of AHA and derivations of other results presented in §3 can be found in the Online Supplement (available at http://joc.pubs.informs.org/ecompanion.html).

2. Background

The problem we face is to find an integer-valued D-dimensional decision variable $\mathbf{x}$ to optimize the expected value of the performance of a system modeled by a stochastic simulation. The performance of the system is a random variable denoted by $G(\mathbf{x})$, which can be observed by running simulation experiments at $\mathbf{x}$. We assume the sample mean of observations of $G(\mathbf{x})$ is a strongly consistent estimator of $g(\mathbf{x}) = \mathrm{E}[G(\mathbf{x})]$. In this paper, we also assume that the optimization problem is a minimization problem on $\Theta = \Phi \cap \mathcal{Z}^D$, where Φ is convex and compact, and $\mathcal{Z}^D$ denotes the D-dimensional integer lattice. Therefore Θ is finite. Stated formally, we want to solve the problem

$$\begin{aligned} &\text{Minimize} \quad g(\mathbf{x}) = \mathrm{E}[G(\mathbf{x})] \\ &\text{subject to} \quad \mathbf{x} \in \Theta = \Phi \cap \mathcal{Z}^D. \end{aligned} \tag{1}$$

Following Hong and Nelson (2006, 2007), we define a *local minimum* of problem 1 as follows.

Definition. Let $\mathcal{N}(\mathbf{x}) = \{\mathbf{y}\colon \mathbf{y} \in \Theta \text{ and } \|\mathbf{x}-\mathbf{y}\| = 1\}$ be the local neighborhood of $\mathbf{x} \in \Theta$, where $\|\mathbf{x}-\mathbf{y}\|$ denotes the Euclidean distance between $\mathbf{x}$ and $\mathbf{y}$. Then $\mathbf{x}$ is a *local minimum* if $\mathbf{x} \in \Theta$ and either $\mathcal{N}(\mathbf{x}) = \varnothing$ or $g(\mathbf{x}) \le g(\mathbf{y})$ for all $\mathbf{y} \in \mathcal{N}(\mathbf{x})$. Let $\mathcal{M}$ denote the set of local minimizers of the function g in Θ.

Suppose that we have an iterative random search algorithm that reports $\hat{\mathbf{x}}_k^*$ as the sample best solution at the end of iteration k. Following Hong and Nelson (2006, 2007) and Xu et al. (2010), we define local convergence of the algorithm as follows.

Definition. An algorithm is a *locally convergent random search* (LCRS) algorithm if the infinite sequence $\{\hat{\mathbf{x}}_0^*, \hat{\mathbf{x}}_1^*, \ldots\}$ generated by the algorithm converges with probability 1 (w.p.1) to the set $\mathcal{M}$ in the sense that $\Pr\{\hat{\mathbf{x}}_k^* \notin \mathcal{M}$ *infinitely often* $(i.o.)\} = 0$.

An iterative random search algorithm for DOvS problems has two essential components: a sampling scheme and an estimation scheme. On iteration k, *sampling* a solution means selecting a feasible solution $\mathbf{x}$ from the MPA $\mathscr{C}_k \subseteq \Theta$, which is usually done by randomly choosing a solution according to a sampling distribution F_k defined on $\mathscr{C}_k$. One example of a sampling distribution is the uniform distribution defined on $\mathscr{C}_k$. When the algorithm samples a solution multiple times within $\mathscr{C}_k$, duplicates are possible, and we denote the set of unique sampled solutions on iteration k as $\mathscr{S}_k$. We use $\mathscr{S}(k)$ to denote the set of all sampled solutions through iteration k. The estimation scheme chooses a subset of solutions $\mathscr{E}_k \subseteq \mathscr{S}(k)$ and allocates $a_k(\mathbf{x})$ additional simulation observations to all $\mathbf{x} \in \mathscr{E}_k$. Let $a_k(\mathbf{x}) = 0$ for all $\mathbf{x} \notin \mathscr{E}_k$. Then, the total number of simulation observations $\mathbf{x}$ has received up to iteration k is $N_k(\mathbf{x}) = \sum_{i=0}^{k} a_i(\mathbf{x})$. The cumulative sample mean of solution $\mathbf{x}$ is $\bar{G}_k(\mathbf{x}) = \sum_{i=1}^{N_k(\mathbf{x})} G_i(\mathbf{x})/N_k(\mathbf{x})$ if $N_k(\mathbf{x}) > 0$, where $G_i(\mathbf{x})$ is the ith observation of $G(\mathbf{x})$.

Hong and Nelson (2007) proposed a generic LCRS algorithm and gave sufficient conditions on the sampling scheme and the estimation scheme to guarantee local convergence. We present their generic algorithm with some minor modifications so that the presentation is consistent with the remainder of this paper.

Algorithm 1 (The generic LCRS algorithm)

Step 0. Let $\mathbf{x}_0$ be the starting solution provided by the user. Set the iteration counter $k = 0$. Let $\mathscr{S}_0 = \mathscr{S}(0) = \{\mathbf{x}_0\}$ and $\hat{\mathbf{x}}_0^* = \mathbf{x}_0$. Set $\mathscr{E}_0 = \{\mathbf{x}_0\}$. Determine $a_0(\mathbf{x}_0)$. Take $a_0(\mathbf{x}_0)$ observations from $\mathbf{x}_0$, set $N_0(\mathbf{x}_0) = a_0(\mathbf{x}_0)$, and calculate $\bar{G}_0(\mathbf{x}_0)$.

Step 1. Let $k = k+1$. Determine the MPA $\mathscr{C}_k$ and the sampling distribution F_k on $\mathscr{C}_k$ according to the sampling scheme. Sample m_k solutions $\mathbf{x}_{k1}, \mathbf{x}_{k2}, \ldots, \mathbf{x}_{km_k}$ from $\mathscr{C}_k$ using F_k. Remove any duplicates from $\mathbf{x}_{k1}, \mathbf{x}_{k2}, \ldots, \mathbf{x}_{km_k}$, and let $\mathscr{S}_k$ be the remaining set. Let $\mathscr{S}(k) = \mathscr{S}(k-1) \cup \mathscr{S}_k$.

Step 2. Determine $\mathscr{E}_k \subset \mathscr{S}(k)$ according to the estimation scheme. For all $\mathbf{x} \in \mathscr{E}_k$, take $a_k(\mathbf{x})$ simulation observations. Update $N_k(\mathbf{x})$ and $\bar{G}_k(\mathbf{x})$. For all $\mathbf{x} \notin \mathscr{E}_k$, let $N_k(\mathbf{x}) = N_{k-1}(\mathbf{x})$ and $\bar{G}_k(\mathbf{x}) = \bar{G}_{k-1}(\mathbf{x})$.

Step 3. Let $\hat{\mathbf{x}}_k^* = \arg\min_{\mathbf{x}\in\mathscr{E}_k} \bar{G}_k(\mathbf{x})$. Go to Step 1.

The convergent optimization via most promising area stochastic search (COMPASS) algorithm is an instance of this generic LCRS algorithm. It lets $\mathscr{C}_1 = \Theta$, $\mathscr{C}_k = \{\mathbf{x}\colon \mathbf{x} \in \Theta \text{ and } \|\mathbf{x} - \hat{\mathbf{x}}_{k-1}^*\| \le \|\mathbf{x}-\mathbf{y}\|,\ \forall\, \mathbf{y} \in \mathscr{S}(k-1)\}$ for $k > 1$, uses a uniform distribution defined on $\mathscr{C}_k$ as the sampling distribution F_k, and sets $\mathscr{E}_k = \mathscr{S}(k)$, $m_k = m$ for all k, where m is an algorithm design parameter.

COMPASS converges to a local optimum under mild conditions. When Θ is finite, which is the scope of this paper, COMPASS is convergent as long as each sample mean $\bar{G}_k(\mathbf{x})$ satisfies a strong law of large numbers and the estimation scheme guarantees that $N_k(\mathbf{x})$ goes to infinity as $k \to \infty$ for all $\mathbf{x} \in \mathscr{S}(k)$.

The constraints defining the COMPASS MPA have the form

$$(\hat{\mathbf{x}}_{k-1}^* - \mathbf{x}_i)'\left(\mathbf{x} - \frac{\hat{\mathbf{x}}_{k-1}^* + \mathbf{x}_i}{2}\right) \ge 0, \quad \mathbf{x}_i \in \mathscr{S}(k-1). \tag{2}$$

Not all constraints are required to define $\mathscr{C}_k$. Hong and Nelson (2007) noted that to guarantee local convergence, it is sufficient to drive $N_k(\mathbf{x})$ to infinity only for those solutions $\mathbf{x}_i \in \mathscr{S}(k-1)$ that yield active constraints defining the MPA $\mathscr{C}_k$, which greatly reduces the computational effort.

To determine whether solution $\mathbf{x}_i \in \mathscr{S}(k-1)$ defines an active constraint, Xu et al. (2010) showed that one can solve the following LP:

$$\begin{aligned} \min_{\mathbf{x}} \quad & (\hat{\mathbf{x}}_{k-1}^* - \mathbf{x}_i)'\left(\mathbf{x} - \frac{\hat{\mathbf{x}}_{k-1}^* + \mathbf{x}_i}{2}\right) \\ \text{s.t.} \quad & (\hat{\mathbf{x}}_{k-1}^* - \mathbf{x}_j)'\left(\mathbf{x} - \frac{\hat{\mathbf{x}}_{k-1}^* + \mathbf{x}_j}{2}\right) \ge 0 \\ & \forall\, \mathbf{x}_j \in \mathscr{S}(k-1) \setminus \{\hat{\mathbf{x}}_{k-1}^*\},\ j \ne i. \end{aligned} \tag{3}$$

The solution $\mathbf{x}_i$ defines an active constraint if and only if the objective function value is negative.

Xu et al. (2010) referred to those solutions defining active constraints as *active solutions* and used the LP (3) for *constraint pruning*. In practice, we do not necessarily want to run the constraint pruning procedure at the end of each iteration. Xu et al. (2010) used a *constraint pruning parameter* in the ISC software to control the frequency of constraint pruning. However, it is difficult to know a priori what is a good choice for this parameter.

Hong and Nelson (2006) compared COMPASS to random search, simulated annealing, and OptQuest.

Numerical experiments showed that COMPASS performs much better than random search and simulated annealing and is competitive with OptQuest. Xu et al. (2010) compared the ISC software to OptQuest in more extensive numerical experiments and showed that ISC is able to achieve competitive finite-time performance and yet deliver convergence guarantees. However, they also noted that as dimensionality increases beyond 10, ISC exhibits a dramatic slowdown, even with constraint pruning.

One factor causing the slowdown is the increasingly large LPs the constraint pruning procedure needs to solve. However, constraint pruning is essential to keeping the amount of simulation effort low, because otherwise, we would need to simulate all visited solutions, which can be as many as $m \cdot k$ solutions at iteration k. In this paper, we show that the geometry of the COMPASS MPA is also an impediment to solving large-dimension problems, and we propose an alternative.

3. Adaptive Hyperbox

In this section, we propose using a hyperbox geometry for constructing the MPA for Algorithm 1. We describe the algorithm and state its convergence property in §3.1. Through a model characterizing the behavior of both algorithms, we then gain some insight into why AHA more easily scales up to higher dimensions than COMPASS in §3.2.

3.1. The Adaptive Hyperbox Algorithm

For a visited solution $\mathbf{x}$, let $x^{(d)}$ be its dth coordinate, $1 \le d \le D$. Let $l_k^{(d)} = \max_{\mathbf{x}\in\mathcal{S}(k),\, \mathbf{x}\neq\hat{\mathbf{x}}_k^*}\{x^{(d)}\colon x^{(d)} < \hat{x}^{*(d)}\}$ if it exists; otherwise, let $l_k^{(d)} = -\infty$. Similarly, let $u_k^{(d)} = \min_{\mathbf{x}\in\mathcal{S}(k),\, \mathbf{x}\neq\hat{\mathbf{x}}_k^*}\{x^{(d)}\colon x^{(d)} > \hat{x}_k^{*(d)}\}$ if it exists; otherwise, let $u_k^{(d)} = \infty$. Note that $u_k^{(d)}$ and $l_k^{(d)}$ may be $\pm\infty$ because $\hat{\mathbf{x}}^*$ may be on the boundary, or we may not have visited enough solutions yet. The hyperbox containing $\hat{\mathbf{x}}_k^*$ is $\mathcal{H}_k = \{\mathbf{x}\colon l_k^{(d)} \le x^{(d)} \le u_k^{(d)},\ 1 \le d \le D\}$. Put another way, $u_k^{(d)}$ and $l_k^{(d)}$ specify the positions of the two edges of the hyperbox along the dth coordinate direction if such edges exist. These two edges cross two solutions that have already been visited and are closest to $\hat{\mathbf{x}}_k^*$ along coordinate direction d among all visited solutions, but not in the same position as $\hat{\mathbf{x}}_k^*$. Let $\mathcal{L}_k = (l_k^{(1)}, \ldots, l_k^{(D)})$ and $\mathcal{U}_k = (u_k^{(1)}, \ldots, u_k^{(D)})$.

AHA constructs its MPA $\mathcal{C}_k$ at iteration k by finding $\mathcal{H}_k$ and letting $\mathcal{C}_k = \mathcal{H}_k \cap \Theta$. Notice that it is much easier to identify $\mathcal{H}_k$ than to identify the set of active solutions for the COMPASS algorithm. Again, we define the local neighborhood of a solution $\mathbf{x}$ as $\mathcal{N}(\mathbf{x})$. The algorithm is described below.

Algorithm 2 (Adaptive hyperbox)

Step 0. Let $\mathbf{x}_0$ be the starting solution provided by the user. Set the iteration counter $k = 0$. Let $\mathcal{S}_0 = \mathcal{S}(0) = \{\mathbf{x}_0\}$ and $\hat{\mathbf{x}}_0^* = \mathbf{x}_0$. Set $\mathcal{E}_0 = \{\mathbf{x}_0\}$. Determine $a_0(\mathbf{x}_0)$. Take $a_0(\mathbf{x}_0)$ observations from $\mathbf{x}_0$, set $N_0(\mathbf{x}_0) = a_0(\mathbf{x}_0)$, and calculate $\bar{G}_0(\mathbf{x}_0)$.

Step 1. Let $k = k+1$. Identify $\mathcal{U}_k$ and $\mathcal{L}_k$ and thus $\mathcal{H}_k$ (for $k = 1$, $\mathcal{U}_k = \varnothing$, $\mathcal{L}_k = \varnothing$, and $\mathcal{C}_k = \Theta$). Let $\mathcal{C}_k = \mathcal{H}_k \cap \Theta$. Sample $\mathbf{x}_{k1}, \mathbf{x}_{k2}, \ldots, \mathbf{x}_{km}$ independently from $\mathcal{C}_k$ using a sampling distribution F_k defined on $\mathcal{C}_k$. Remove any duplicates from $\mathbf{x}_{k1}, \mathbf{x}_{k2}, \ldots, \mathbf{x}_{km}$, and let $\mathcal{S}_k$ be the remaining set. Let $(k) = \mathcal{S}(k-1) \cup \mathcal{S}_k$.

Step 2. Let $\mathcal{E}_k = \mathcal{S}_k \cup \{\hat{\mathbf{x}}_{k-1}^*\}$. For all $\mathbf{x} \in \mathcal{E}_k$, take $a_k(\mathbf{x})$ simulation observations and update $N_k(\mathbf{x})$ and $\bar{G}_k(\mathbf{x})$. For all $\mathbf{x} \notin \mathcal{E}_k$, let $N_k(\mathbf{x}) = N_{k-1}(\mathbf{x})$ and $\bar{G}_k(\mathbf{x}) = \bar{G}_{k-1}(\mathbf{x})$.

Step 3. Let $\hat{\mathbf{x}}_k^* = \arg\min_{\mathbf{x}\in\mathcal{E}_k} \bar{G}_k(\mathbf{x})$. Go to Step 1.

To better understand the differences in MPA construction between COMPASS and AHA, it is helpful to examine a simple two-dimensional example. In Figure 1, the feasible solution space Θ is the integer points in a rectangle. The current sample best solution is $\hat{\mathbf{x}}_k^*$, and there are four other visited solutions: $\mathbf{x}_1$, $\mathbf{x}_2$, $\mathbf{x}_3$, and $\mathbf{x}_4$. COMPASS places a hyperplane halfway between $\hat{\mathbf{x}}_k^*$ and each solution $\mathbf{x}_1$, $\mathbf{x}_2$, $\mathbf{x}_3$, and $\mathbf{x}_4$ (which is a line in two dimensions). In this particular example, there are four COMPASS constraints. In contrast, the AHA MPA is defined by $\mathbf{x}_1$ and $\mathbf{x}_2$. Notice that the four COMPASS constraints do not form a closed MPA by themselves and rely on the boundaries of Θ to define $\mathcal{C}_k$. Because of this, the volume of the COMPASS MPA may be much larger than that of the AHA MPA for the same set of visited solutions, especially when D is large.

Although the MPA of AHA and COMPASS differ, we can use the same rule as in COMPASS (Hong and Nelson 2006) to determine the number of additional simulation observations $a_k(\mathbf{x})$ for each solution $\mathbf{x} \in \mathcal{E}_k$. For example, we can set the total number of observations for $\mathbf{x} \in \mathcal{E}_k$ through iteration k:

$$N_k(\mathbf{x}) = \min\{5, \lceil 5(\log k)^{1.01}\rceil\}. \tag{4}$$

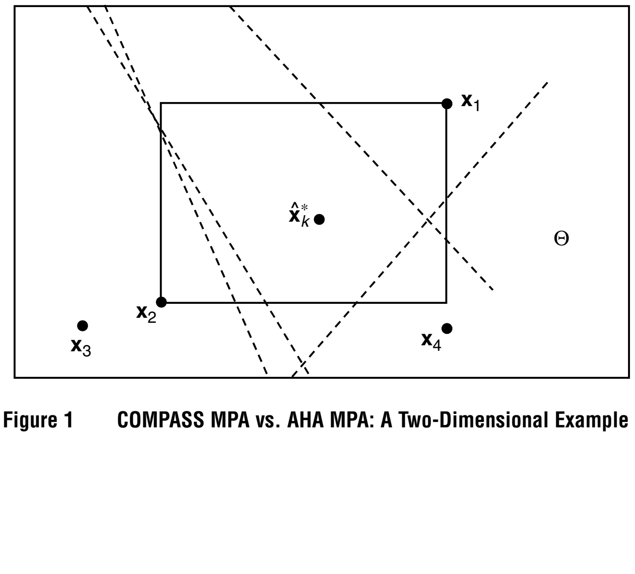

Figure 1 COMPASS MPA vs. AHA MPA: A Two-Dimensional Example

To prove the local convergence of AHA, we first need the following assumption, which states that the sample mean of $G(\mathbf{x})$ is a strongly consistent estimator of $g(\mathbf{x})$.

Assumption 1. *For all* $\mathbf{x} \in \Theta$,

$$\lim_{r\to\infty} \frac{1}{r}\sum_{i=1}^{r} G_i(\mathbf{x}) = g(\mathbf{x}) \quad \text{w.p. } 1.$$

When we generate independent and identically distributed replications of a terminating simulation, Assumption 1 is simply the strong law of large numbers. Under certain conditions, steady-state simulation also satisfies this assumption.

We modify the conditions on the sampling scheme and the estimation scheme for the generic LCRS algorithm proposed by Hong and Nelson (2007), so that we can have a more efficient implementation of AHA. We show that when the solution space is finite, the new conditions also guarantee the convergence of Algorithm 1. We then verify that AHA satisfies these conditions. Because AHA is a particular instance of Algorithm 1, it is an LCRS algorithm.

Condition 1. The sampling scheme satisfies the following requirement:

The sampling distribution F_k guarantees that $\Pr\{\mathbf{x} \in \mathcal{S}_k\} \ge \epsilon$ for all $\mathbf{x} \in \mathcal{N}(\hat{\mathbf{x}}^*_{k-1})$ for some $\epsilon > 0$ that is independent of k.

Condition 2. The estimation scheme satisfies the following requirements:

1. $\mathcal{E}_k$ is a subset of $\mathcal{S}(k)$;
2. $\mathcal{E}_k$ contains $\hat{\mathbf{x}}^*_{k-1}$ and $\mathcal{S}_k$; and
3. $a_k(\mathbf{x})$ is allocated such that $\min_{\mathbf{x}\in\mathcal{E}_k} N_k(\mathbf{x}) \ge 1$ for all $k = 1, 2, \ldots,$ and $\min_{\mathbf{x}\in\mathcal{E}_k} N_k(\mathbf{x}) \to \infty$ w.p. 1 as $k \to \infty$.

The first condition requires that the sampling scheme gives a nonzero sampling probability to all feasible neighbors of the current sample best solution. By comparison, Hong and Nelson (2007) only required that the feasible neighbors of the current sample best solution *that have not yet been visited* have a nonzero probability to be sampled. So the new sampling condition is stronger than the original sampling condition. The second condition requires the algorithm to estimate the sample best solution from the previous iteration as well as the newly sampled solutions, and as the iteration count goes to infinity, all solutions in the estimation set receive infinitely many simulation observations. Hong and Nelson (2007) gave a stronger estimation condition by requiring the algorithm to estimate *all* visited neighbors of the sample best solution from the previous iteration, in addition to the sample best solution of the previous iteration and the newly sampled solutions. Hong and Nelson (2007) also imposed additional conditions to guarantee convergence for unbounded DOvS problems ($|\Theta| = \infty$). These conditions are not necessary in the context of this paper, and thus we drop them.

We first state the result that gives the convergence of Algorithm 1 when Conditions 1 and 2 hold. The proof is provided in the Online Supplement.

Proposition 1. *Let* $\hat{\mathbf{x}}^*_k$, $k = 0, 1, 2, \ldots$ *be a sequence of solutions generated by Algorithm* 1 *when applied to problem* (1). *Suppose that Assumption* 1 *is satisfied. If Conditions* 1 *and* 2 *hold, then* $\Pr\{\hat{\mathbf{x}}^*_k \notin \mathcal{M} \text{ i.o.}\} = 0$.

Because Proposition 1 guarantees the convergence of any particular design of the generic algorithm, it is sufficient to verify that AHA satisfies Conditions 1 and 2 to establish AHA's local convergence property for some sampling distribution F_k. Throughout the remainder of this paper, we let F_k be the uniform distribution defined on the MPA $\mathcal{C}_k$.

Proposition 2. *AHA is an instance of the general LCRS algorithm when solutions are uniformly randomly sampled within the MPA* $\mathcal{C}_k$ *at each iteration* k.

Proof. We verify that AHA satisfies Conditions 1 and 2. To verify Condition 1, we need to compute $\Pr\{\mathbf{x} \in \mathcal{S}_k\}$ for all $\mathbf{x} \in \mathcal{N}(\hat{\mathbf{x}}^*_{k-1})$. We first notice that $\mathcal{N}(\hat{\mathbf{x}}^*_{k-1}) \subseteq \mathcal{H}_k \cap \mathbf{\Theta} = \mathcal{C}_{k-1}$ by construction. Denote the m solutions independently and uniformly sampled within $\mathcal{C}_{k-1}$ as $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_m$. For all $\mathbf{x} \in \mathcal{N}(\hat{\mathbf{x}}^*_{k-1})$, we have $\Pr\{\mathbf{x} \in \mathcal{S}_k\} = 1 - \Pr\{\mathbf{x} \notin \mathcal{S}_k\} = 1 - \Pr\{\mathbf{x}_1 \ne \mathbf{x}, \ldots, \mathbf{x}_m \ne \mathbf{x}\} = 1 - \Pr\{\mathbf{x}_1 \ne \mathbf{x}\}^m$. Then we have $\Pr\{\mathbf{x}_1 \ne \mathbf{x}\} = (|\mathcal{C}_{k-1}| - 1)/|\mathcal{C}_{k-1}|$. So

$$\Pr\{\mathbf{x} \in \mathcal{S}_k\} = 1 - \left(1 - \frac{1}{|\mathcal{C}_{k-1}|}\right)^m \ge 1 - \left(1 - \frac{1}{|\Theta|}\right)^m > 0.$$

Thus Condition 1 is satisfied.

Next, we check Condition 2. It is clear that $\mathcal{E}_k$ is a subset of $\mathcal{S}(k)$. So the first requirement is satisfied. By construction, Algorithm 2 also satisfies the second part of Condition 2. The third requirement is on the sample allocation schedule, and thus we can use (4), which satisfies this requirement. Therefore, Condition 2 is also satisfied. □

When implementing the algorithm, we have different ways to identify $\mathcal{U}_k$ and $\mathcal{L}_k$. A straightforward implementation is to use a data structure such as the multimap provided by the standard C++ container library to record the positions of all visited solutions for each coordinate, and then for each $\hat{\mathbf{x}}^*_k$, search the entire map for $u^{(d)}_k$ and $l^{(d)}_k$. The algorithmic complexity is $O(|\mathcal{S}(k)| \log(|\mathcal{S}(k)|))$. In practice, where simulation is much more time consuming than a map lookup, this overhead is not significant. Compared with the constraint pruning in the original COMPASS algorithm, this overhead is also quite small.

3.1.1. The Local Optimality Stopping Test for AHA. Like COMPASS, AHA is an asymptotically convergent algorithm; i.e., the algorithm is locally convergent as the number of iterations goes to infinity. However, in practice, the algorithm will be terminated at some point. A reasonable approach is to compare the current sample best solution and all of its neighbors and decide whether to accept it as the local optimum with a statistical guarantee. Xu et al. (2010) developed one such test, which takes the form of

$$H_0\colon g(\hat{\mathbf{x}}_k^*) \le \min_{\mathbf{y}\in\mathcal{N}(\hat{\mathbf{x}}_k^*)} g(\mathbf{y}) \quad \text{versus} \quad H_1\colon g(\hat{\mathbf{x}}_k^*) > \min_{\mathbf{y}\in\mathcal{N}(\hat{\mathbf{x}}_k^*)} g(\mathbf{y}).$$

The Type I error of the test is set to α. If $g(\hat{\mathbf{x}}_k^*) \ge \min_{\mathbf{y}\in\mathcal{N}(\hat{\mathbf{x}}_k^*)} g(\mathbf{y}) + \delta$, the power of the test is at least $1-\alpha$, where δ is the tolerance level provided by the user. There are two outcomes of the test: either $\hat{\mathbf{x}}_k^*$ passes the test and is accepted as the local optimum or some other solution in $\mathcal{N}(\hat{\mathbf{x}}_k^*)$ is returned as the current sample best solution and the search continues.

The construction of $\mathscr{C}_k$ means that AHA will always have more than one solution inside the MPA. There are two obvious options for the local optimality test. In option 1, as soon as $\hat{\mathbf{x}}_k^*$ is the only *interior* solution of the MPA $\mathscr{C}_k$, the algorithm hands $\hat{\mathbf{x}}_k^*$ and all of its neighbors (some of which may not have been visited yet) to the stopping test procedure. However, this may lead to too many premature tests and consume a lot of simulation replications unnecessarily. In option 2, when $\hat{\mathbf{x}}_k^*$ is the only interior solution of the MPA $\mathscr{C}_k$, the algorithm checks whether all of $\hat{\mathbf{x}}_k^*$'s neighbors have been visited. If not, the algorithm proceeds to the next iteration; if so, the algorithm hands $\hat{\mathbf{x}}_k^*$ and all of its neighbors to the stopping test procedure. However, waiting for all neighbors to be visited by uniform random sampling can take a long time for high-dimensional problems and hence will unnecessarily slow down AHA.

Therefore, it is of great practical importance to achieve a proper balance between applying the stopping test prematurely and waiting for all neighbors to be visited by uniform random sampling. We modify AHA so that all unvisited neighbors of $\hat{\mathbf{x}}_k^*$ can be visited sooner than using uniform random sampling of the MPA when $\hat{\mathbf{x}}_k^*$ is the only interior solution of the MPA. The following steps are executed after Step 3 in Algorithm 2.

Algorithm 3 (AHA stopping test)

Step 3.1. Check if $\hat{\mathbf{x}}_k^*$ is the only interior solution in $\mathscr{C}_k$. If not, continue with Step 1 in Algorithm 2.

Step 3.2. Let $k = k+1$. For all $\mathbf{x} \in \mathcal{N}(\hat{\mathbf{x}}_{k-1}^*)$, check if $\mathbf{x} \in \mathscr{S}(k-1)$. Let $\mathscr{D}_k \subseteq \{1, 2, \ldots, D\}$ be the set of coordinate directions along which there are unvisited neighbors. If $\mathscr{D}_k = \varnothing$, invoke stopping test.

Step 3.3. Randomly pick a coordinate direction from $\mathscr{D}_k$ and sample one neighbor along that direction. Repeat the process $m-2$ times to generate $\mathbf{x}_{k1}, \mathbf{x}_{k2}, \ldots, \mathbf{x}_{k(m-1)}$. Sample randomly within $\mathscr{C}_{k-1}$ to generate $\mathbf{x}_{km}$. Remove any duplicates from $\mathbf{x}_{k1}, \mathbf{x}_{k2}, \ldots, \mathbf{x}_{km}$, and let $\mathscr{S}_k$ be the remaining set. Let $\mathscr{S}(k) = \mathscr{S}(k-1) \cup \mathscr{S}_k$ and $\mathscr{C}_k = \mathscr{C}_{k-1}$. Continue with Step 2 in Algorithm 2.

In Step 3.3, we randomly sample $\mathbf{x}_{km}$ from $\mathscr{C}_{k-1}$ to satisfy Condition 1 and thus maintain the convergence property of AHA. This procedure performs well in our numerical experiments. Alternative procedures may further improve the finite-time performance of AHA, but they are not the focus of this article.

3.2. Analysis of AHA vs. COMPASS

Intuitively, AHA should scale well in high-dimensional problems because at most $2D$ solutions and as few as two solutions are needed to construct a hyperbox containing $\mathbf{x}_k^*$, where D is the dimension of the search space. Figure 1 illustrates this. In this two-dimensional example, two solutions define the hyperbox, which is a rectangle. However, this plane will be a hyperplane in an arbitrary high-dimensional problem, and only two solutions can define the entire hyperbox. In contrast, COMPASS needs to have more and more solutions to define an enclosing MPA containing $\mathbf{x}_k^*$ as dimension increases. Numerical experiments also support this intuitive belief. In this subsection, we develop a model to characterize the behavior of AHA and COMPASS and provide insights into the performance differences between them.

It is virtually impossible to model, in a mathematically tractable and comprehensive way, anything but the asymptotic behavior of adaptive random search DOvS algorithms. However, it is possible to analyze one aspect of the algorithm's behavior: how it cuts down the MPA for a deterministic problem. We undertake such an analysis here. Of course, more than the volume of the MPA matters in overall algorithm performance. However, the effectiveness of a locally convergent DOvS algorithm depends on its ability to focus in on a locally optimal solution. An algorithm whose ability to do so degrades as dimension increases in the best case (deterministic output) will certainly struggle in a stochastic problem. Thus, the following analysis helps explain why AHA is more effective than COMPASS in our empirical study in §4 and suggests that we can expect it to be effective more generally.

We look at a deterministic optimization problem with continuous decision variables and model how each algorithm reduces the volume of the MPA from iteration to iteration. The feasible region is a hyperbox of volume 1 in a D-dimension solution space. The optimization problem has a unique locally optimal solution, $\mathbf{x}^* = (0, 0, \ldots, 0)^T$. We study two cases. In the first case, the feasible solution, space is $\Theta_1 = [0, 1]^D$. In this case, $\hat{\mathbf{x}}^*$ is located at the corner of the feasible solution space. The second case has $\Theta_2 = [-1/2, 1/2]^D$, and thus $\mathbf{x}^*$ is located at the center of the feasible solution space. These two cases represent the two extremes among all possible locations of $\mathbf{x}^*$. We shall refer to them as "Corner Case" and "Center Case" hereafter.

We assume that $\mathbf{x}^*$ is given as the initial solution of the optimization problem. In one iteration, we sample m solutions $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_m$ uniformly from Θ_1 or Θ_2. We then construct the MPA with these solutions using AHA or COMPASS. Because the problem is deterministic, all solutions are estimated without noise, and thus $\mathbf{x}^*$ is the sample best solution. The measure of the algorithms we consider is $\mathrm{E}(V)$, where V is the volume of MPA in each case. The smaller $\mathrm{E}(V)$ is, the more efficient the algorithm is in terms of reducing the search area. Derivations of all analytical results in this section are given in the Online Supplement.

3.2.1. The AHA MPA. For the Corner Case, the expected volume of the MPA constructed according to Algorithm 2 using m solutions $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_m$ uniformly randomly sampled within Θ_1 is

$$\mathrm{E}(V) = \left(\frac{1}{m+1}\right)^D. \tag{5}$$

For the Center Case, the expected volume of the MPA constructed according to Algorithm 2 using m solutions $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_m$ uniformly randomly sampled within Θ_2 is given by

$$\mathrm{E}(V) = \left\{\frac{2}{m+1}\left[1 - \left(\frac{1}{2}\right)^{m+1}\right]\right\}^D. \tag{6}$$

As the number of sampled solutions m increases, (6) can be very accurately approximated by

$$\mathrm{E}(V) \approx \left(\frac{2}{m+1}\right)^D, \tag{7}$$

which is also an upper bound for $\mathrm{E}(V)$ in the Center Case.

We can extend the analysis above to other locations for $\mathbf{x}^*$. But since the Corner Case is the most favorable position because the problem constraints (i.e., the boundaries of Θ) help reduce the size of the MPA, and the Center Case is the least favorable because we have to rely on sampling solutions to close in on $\mathbf{x}^*$ from all directions, we may conclude that at each iteration, AHA reduces the volume of the MPA at a rate of $(C/(m+1))^D$, where C is a constant between 1 and 2 and varies from iteration to iteration.

The formula for $\mathrm{E}(V)$ can also provide guidance on how to allocate simulation budget to most efficiently reduce the size of the MPA and thus minimize the number of simulation replications needed to find the local optimum. Suppose we have a fixed budget of M simulation replications, and it takes one replication to evaluate each sampled solution. If we sample m solutions at each iteration, then we will use up the simulation budget at iteration $k \approx M/m$. At that time, the expected volume is $\mathrm{E}(V_k) \sim (C/(m+1))^{DM/m}$. We would like to minimize $\mathrm{E}(V_k)$ by choosing m. By simple analysis, we can find that $\mathrm{E}(V_k)$ is minimized by letting $m = 2.8$ (when $C = 1$) or $m = 5.5$ (when $C = 2$). Therefore we may conclude that it is reasonable to sample three to six solutions at each iteration.

3.2.2. The COMPASS MPA. Unlike AHA, the geometry of COMPASS is much more complicated, and thus we do not have simple closed-form expressions for $\mathrm{E}(V)$. However, we are able to derive asymptotic lower bounds when D is large. For the Corner Case, the expected volume of the MPA constructed by COMPASS using m solutions $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_m$ uniformly randomly sampled within $\Theta_1 = [0, 1]^D$ has an asymptotic lower bound given by

$$\mathrm{E}(V) \geq \Phi(-0.49\sqrt{D})^m, \tag{8}$$

where Φ is the cumulative distribution function of the standard normal distribution. For the Center Case, we have

$$\mathrm{E}(V) \geq \Phi(0.46\sqrt{D})^m. \tag{9}$$

To verify the accuracy of the approximation, we ran simulation experiments with different m values, and the results show that the approximation is quite accurate for dimension D as small as 3. The maximum relative error is 5.96% for Corner Case and 5.63% for Center Case when m is between 2 and 20. The accuracy improves as m increases.

Unlike AHA, the MPA formed by COMPASS is no longer a hyperbox. As a result, we are not able to extend the analysis to multiple iterations.

Again, the volume of the MPA alone is not a complete characterization of the performance of an LCRS algorithm. Shrinking too quickly could also be inefficient. At a minimum, our analysis sheds light on how AHA and COMPASS differ; we believe that it also helps explain the empirical results in §4.

3.2.3. Comparison. From the previous analyses of AHA and COMPASS, we see that the location of the optimal solution $\mathbf{x}^*$, the number of solutions sampled at each iteration m, and dimension D influence the volume-shrinking efficiency of the algorithms in very different ways. In Figure 2, we plot $\mathrm{E}(V)$ for both AHA and COMPASS using the approximations.

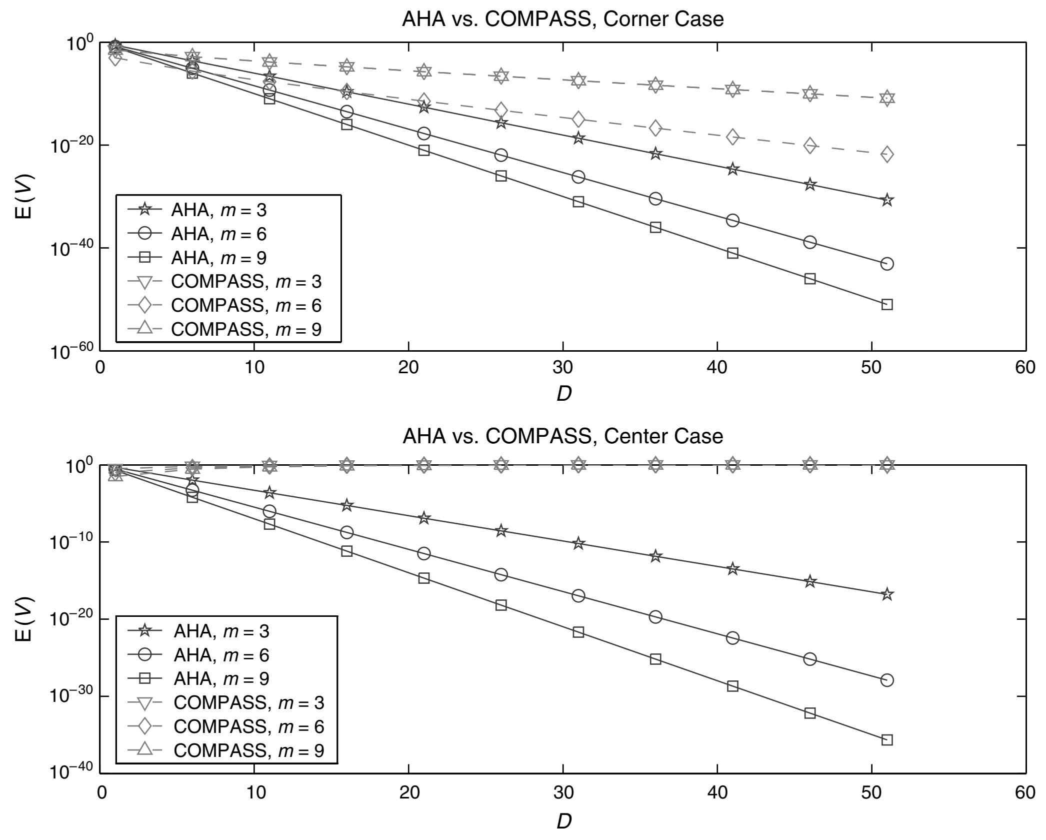

Figure 2 E(V) as a Function of D for Different m for AHA and COMPASS

Although it is not surprising that both algorithms perform better for the Corner Case than the Center Case, it is interesting to see that COMPASS is much more sensitive to the location of $\mathbf{x}^*$ than AHA. For COMPASS, when $\mathbf{x}^*$ is at the center of Θ, $\mathrm{E}(V)$ increases as D increases, whereas $\mathrm{E}(V)$ is decreasing in D when $\mathbf{x}^*$ is at the corner of Θ. In contrast, for AHA the effect of the location of $\mathbf{x}^*$ is only through the constant in the numerator, which is 1 for the Corner Case and 2 for the Center Case. So AHA's ability to shrink the MPA is more robust than COMPASS with respect to the location of $\mathbf{x}^*$.

Even though COMPASS may be more efficient at shrinking the MPA than AHA when D is small and m is large for the Corner Case (e.g., $D < 10, m = 9$; as can be seen in Figure 2), as D increases the efficiency of AHA quickly catches up and surpasses that of COMPASS.

Finally, since D appears as the exponent in AHA's formula for $\mathrm{E}(V)$, we see that AHA's ability to shrink the volume of the MPA keeps up with the exponential increase in the number of feasible solutions as D increases. We thus expect AHA to scale up well as dimensionality increases. By comparison, in the Corner Case the ability of COMPASS to shrink the volume of the MPA clearly falls far behind the exponential increase in the number of feasible solutions as dimensionality increases (note that the vertical axis has log scale). For the Center Case, as D increases, inequality (9) shows that $\mathrm{E}(V)$ actually increases, which helps to explain why COMPASS slows down dramatically as the dimension of Θ increases.

The analysis in this section is far from characterizing every aspect of the behavior of AHA and COMPASS in general DOvS problems. Nor does it establish the superiority of AHA over COMPASS in terms of the quality of the final solution, given the same computation budget. However, it does illuminate the fundamental impact of MPA geometry on the ability of the algorithm to converge to a locally optimal solution.

4. Numerical Experiments

Hong and Nelson (2006) compared the performance of COMPASS to pure random search and simulated annealing. Xu et al. (2010) further compared the performance of ISC to one of the most popular commercial OvS software packages, OptQuest, and demonstrated that ISC's finite-time performance was comparable to that of OptQuest and, in many cases, even outperformed OptQuest. Because those two studies have established the competitiveness of ISC and stand-alone COMPASS, we focus on comparing the performance of AHA to that of COMPASS, both as the local search phase algorithm of ISC and as a stand-alone algorithm.

4.1. Test Problems

We use the same set of test problems as in Xu et al. (2010). In this section, we briefly describe these test problems. Detailed descriptions can be found in Xu et al. (2010).

The first test problem is the *multimodal function* $g_1(x_1, x_2)$:

$$F_2(x) = \frac{\sin^6(0.05\pi x)}{2^{2((x-10)/80)^2}}, \quad 0 \le x \le 100,$$

$$g_1(x_1, x_2) = -[F_2(x_1) + F_2(x_2)], \quad 0 \le x_1, x_2 \le 100. \quad (10)$$

There are 25 local optima and the global optimum is located at (10, 10) with objective value −2. The second-best local optima are (10, 30) and (30, 10), both with an objective function value of −1.834. We add normally distributed noise with 0 mean and standard deviation of 0.3 to $g_1(x_1, x_2)$.

The second test problem is the *singular function* of Hong (2004):

$$g_2(x_1, x_2, x_3, x_4) = (x_1 + 10x_2)^2 + 5(x_3 - x_4)^2 + (x_2 - 2x_3)^4 + 10(x_1 - x_4)^4 + 1. \quad (11)$$

When only integer solutions are considered, this function has three local minima: $(0, 0, 0, 0)$ with $g_2(0, 0, 0, 0) = 1$; $(1, 0, 0, 1)$ with $g_2(1, 0, 0, 1) = 7$; and $(-1, 0, 0, -1)$ with $g_2(-1, 0, 0, -1) = 7$. We add normally distributed noise with 0 mean and standard deviation 30 (substantial noise near the global optimum) to make it a stochastic optimization problem. The feasible solution space is $-100 \le x_i \le 100$, $x_i \in \mathscr{Z}^+$, $i = 1, 2, 3, 4$.

The third test problem is a three-stage *flowline* with finite buffer storage space in front of stations 2 and 3 (denoted by x_4 and x_5) and an infinite number of jobs in front of station 1 (see Buzacott and Shantikumar 1993, Pichitlamken and Nelson 2003). There is a single server at each station, and the service time at station i is exponentially distributed with service rate x_i, $i = 1, 2, 3$. If the buffer of station i is full, then station $i - 1$ is blocked, and a finished job cannot be released from station $i - 1$. The total buffer space and the service rates are limited. The objective is to find a buffer allocation and service rates such that the steady-state throughput is maximized. The constraints are $x_1 + x_2 + x_3 \le 20$, $x_4 + x_5 = 20$, $1 \le x_i \le 20$, and $x_i \in \mathscr{Z}^+$ for $i = 1, 2, \ldots, 5$. This gives 21,660 feasible solutions. The local optima happen to also be global optima: $(6, 7, 7, 12, 8)$ and $(7, 7, 6, 8, 12)$ with steady-state throughput 5.776. The average throughput from time 50 to time 1,000 constitutes a replication.

The fourth test problem is an *inventory management* problem with dynamic consumer substitution. The model that we optimize is adapted from Mahajan and van Ryzin (2001), and it represents a difficult problem for which the optimal solutions are not known. The model considers a one-shot inventory stocking decision faced by a retailer for v product variants at the beginning of a season; no inventory replenishment happens in the model, and there is no salvage value for the products. Pricing is assumed to be an exogenous decision. Variant j's unit price is given as p_j, and the unit cost is c_j. The number of customers is Poisson with mean 10,000, and the customer's choice behavior is modeled by the widely used multinomial logit model. Briefly, variant j's utility to customer t is $U_{tj} = a_j - p_j + \xi_{tj}$, $j = 1, 2, \ldots, v$, where a_j is variant j's quality factor, and ξ_{tj} is a random variable having an extreme value (Type I) distribution. It is assumed that each individual consumer chooses an available product with the highest utility, which may be a no-purchase option. There are six products and one no-purchase option. The stock levels for these six products range from 0 to 500. In Mahajan and van Ryzin (2001), a continuous relaxation of the problem was numerically solved via a sample path algorithm, which is essentially a continuous-variable OvS method.

The fifth problem is to test the performance of the algorithms in *high-dimensional* spaces:

$$g_5(x_1, x_2, \ldots, x_D) = -\beta \exp\left\{-\gamma \sum_{d=1}^{D} (x_d - \xi^*)^2\right\}, \quad (12)$$

where we set $\gamma = 0.001$, $\beta = 10{,}000$, and $\xi^* = 0$. The response surface has the shape of an inverted multivariate normal density function with a single globally optimal solution at $\mathbf{x} = (\xi^*, \xi^*, \ldots, \xi^*)$. The feasible region is the hyperbox defined by

$$x_d \in \left\{-\frac{m^{1/D}}{2}, \frac{m^{1/D}}{2}\right\}$$

for $d = 1, 2, \ldots, D$ with $m = 10^{20}$. We round the bounds to the nearest integer if necessary. Normally distributed noise with standard deviation $0.3 \times |g_5(\mathbf{x})|$ is added. The number of the feasible solutions is kept constant for all D, and thus we can focus on the impact of dimensionality without worrying about the increase in the size of search space.

The last problem is to test the performance of AHA and COMPASS when solving *high-dimensional multimodal* functions:

$$g_6(x_1, x_2, \ldots, x_D) = -\sum_{d=1}^{D} \left(\beta_1 \exp\{-\gamma_1 (x_d - \xi_1^*)^2\} + \beta_2 \exp\{-\gamma_2 (x_d - \xi_2^*)^2\}\right), \quad (13)$$

where we set $\gamma_1 = 0.001, \beta_1 = 300, \gamma_2 = 0.005, \beta_2 = 500$, $\xi_1^* = -38$, and $\xi_2^* = 56$. The response surface along any coordinate d has the shape of two inverted normal density functions with two locally optimal solutions at $x_d \in \{\xi_1^*, \xi_2^*\}$, with the globally optimal solution at $x_d = \xi_2^*$. The combined D-dimensional function has 2^D locally optimal solutions, of the form $x_d = \xi_1^*$ or ξ_2^*, for $d = 1, 2, \ldots, D$, with a unique globally optimal solution at $\mathbf{x} = (\xi_2^*, \xi_2^*, \ldots, \xi_2^*)$. The feasible region is the hyperbox defined by $x_d \in \{-100, 100\}$ for $d = 1, 2, \ldots, D$. Normally distributed noise with standard deviation 1 is added.

4.2. Empirical Results

We plot the performance of the algorithms being compared against the number of simulation replications as the search progresses. When we know the true objective value of a solution, we use the true objective value of the current sample best solution as the performance measure because that is what the user will obtain if the search is terminated at that moment. For the first three test problems, we run COMPASS and AHA as stand-alone algorithms as well as the local phase algorithm for ISC. For the inventory problem, we only run them with ISC because it is closer to what users will do in practice for such a complex problem. For the single-mode, high-dimensional test problem, we only run COMPASS and AHA in stand-alone mode, so that we can focus on the effect of dimensionality. For the high-dimensional multimodal test problem, we only run COMPASS and AHA in conjunction with ISC, because neither algorithm can find the better solutions without it. We use the local optimality test in both stand-alone and ISC mode.

Figures 3, 4, and 5 are the performance plots for the multimodal function, singular function, and flowline problem, respectively (color versions of all performance plots can be found in the Online Supplement). The curves are the average of 50 independent runs for the multimodal function and the singular function and the average of 25 runs for the flowline problem. Note that the local optimality test may stop COMPASS and AHA when local optimal solutions are found. In that case, the curves are the average of the remaining runs.

For the multimodal function, we see that stand-alone AHA is more prone to converging to inferior local optima than stand-alone COMPASS because of its quick convergence behavior. However, in practice, we recommend using AHA for the local search phase of the ISC framework. As we see from the results, the global search phase was able to alleviate the problem of AHA being trapped in inferior local optima by identifying starting points close to

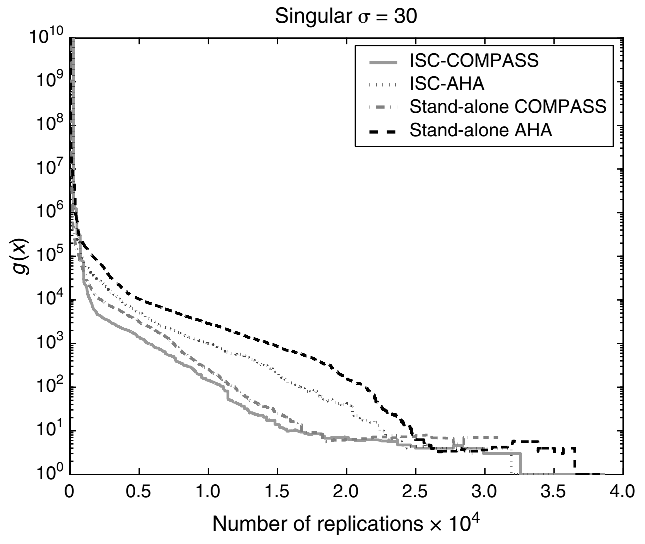

Figure 4 **Performance Plot for the Singular Function**

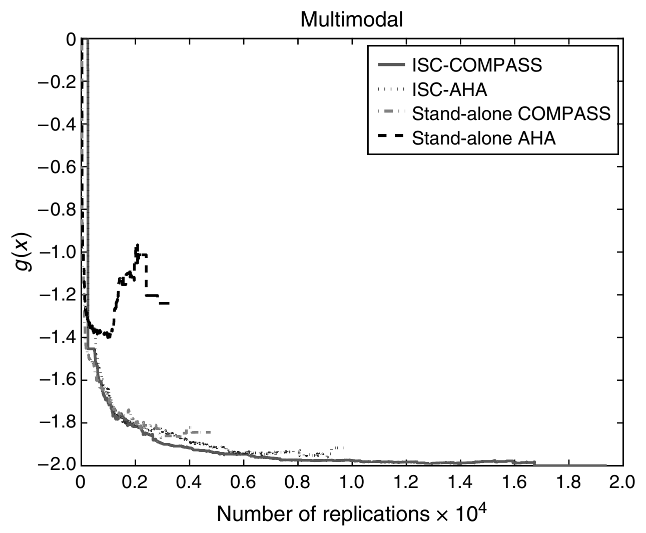

Figure 3 **Performance Plot for the Multimodal Function**

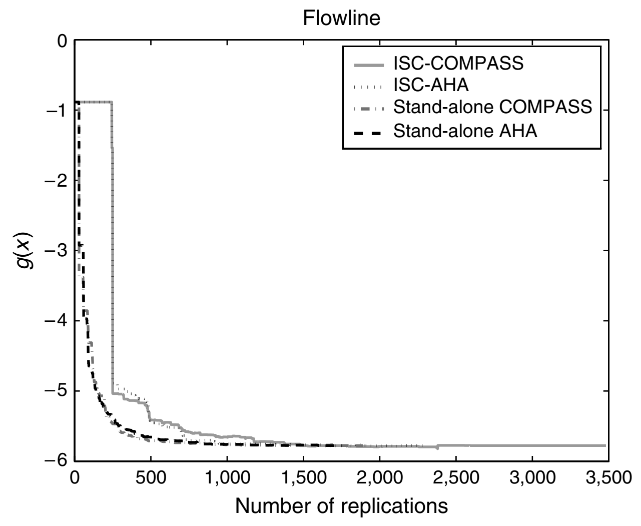

Figure 5 **Performance Plot for the Flowline Problem**

the global optimum that lead to good final results. In both cases, AHA converges to local optima much faster than COMPASS. We should also point out that because different runs can take quite different numbers of simulation replications to finish, the actual terminating performance of AHA is usually better than the curve suggests because the tail of the ISC-AHA curve comes from the few runs that return inferior local optima.

For the singular function, COMPASS performs better than AHA both as a stand-alone algorithm and as the local phase algorithm for ISC. Because AHA quickly shrinks the MPA and the stochastic noise is significant, AHA tends to converge prematurely and then spends a large number of simulation replications doing the stopping test, only to find that the current sample best solution is not a locally optimal solution; it therefore resumes the search. Nevertheless, the performance of AHA is still acceptable on this low-dimensional problem and can be improved with fine tuning of the local optimality stopping test step.

For the flowline problem, the performance curves of COMPASS and AHA are very similar, with AHA being able to stop much earlier because of its ability to quickly reduce the search space.

For the inventory management problem, we do not know the true objective value. To compare ISC-COMPASS and ISC-AHA, we ran each for 10 trials, and at the end of each trial, we ran extensive simulation of the chosen best solution to accurately estimate the objective value. Table 1 shows the results (notice that this problem is a maximization problem). For each algorithm, the $\bar{G}(\hat{\mathbf{x}}^*)$ column shows the estimated value of the sample best solution when the algorithm terminated, and the column labeled "True" is an estimate of this solution's true expected value based on 10,000 replications (the standard error of this estimate is shown in the SE column). The N column gives the number of simulation replications used by the algorithm when it stops.

We again observe similar patterns. In general, ISC-AHA is more aggressive than ISC-COMPASS in looking for locally optimal solutions. Usually, the number of simulation replications that ISC-AHA spends is less than half of what ISC-COMPASS uses. However, ISC-COMPASS searches the space more thoroughly than ISC-AHA, and the performance, is on average, 2.2% better than ISC-AHA. In some sense, for low-dimensional problems, ISC-COMPASS and ISC-AHA present a trade-off between speed and quality for the user to choose.

For the high-dimensional test problems, we try $D =$ 5, 10, 15, and 20 and plot the results in Figure 6. We take the average of 25 trials for $D = 5$, 10, and 15. Because COMPASS is extremely slow when $D = 20$, we only plot one sample path. Also note that the horizontal axis in the $D = 20$ case has a log scale. We see from the plots that both COMPASS and AHA eventually find the optimal solution in all cases. However, AHA is much faster than COMPASS.

To illustrate the performance of AHA in higher dimensional cases, we also apply AHA to $D = 50$ and $D = 100$, which are virtually impossible for COMPASS to handle. For these two experiments, we keep the range of x_j the same as the $D = 20$ case to show how AHA is able to handle the exponential growth in the volume of the search space. AHA are able to find the global optimal in all trials. Table 2 compares the average CPU time (in seconds) of each COMPASS and AHA run for high-dimensional problems.

Recall the analysis in §§3.2.1 and 3.2.2. The high-dimensional test problems are the Center Case in the analysis. According to (6), the volume of the MPA shrinks exponentially with a rate of D and thus scales up well as D increases. In contrast, Equation (9) tells us after the first iteration, COMPASS's ability to shrink the volume of MPA decreases as D increases. As a result, we see the dramatic differences in convergence speed between COMPASS and AHA. The tremendous savings in CPU time achieved by AHA also comes from the fact that AHA's algorithmic overhead is much smaller than that of COMPASS. AHA's biggest overhead is the map lookup step to determine the position of the hyperbox when constructing the MPA. In contrast, COMPASS has a computationally intensive constraint pruning step whose performance deteriorates quickly as dimension increases.

Because AHA converges to locally optimal solutions much more aggressively than COMPASS does, a valid concern is whether this will lead to a deterioration of solution quality when the response surface has many inferior locally optimal solutions. Our first test problem showed that AHA by itself tends to be trapped into inferior locally optimal solutions, but the

Table 1 Ten Trials of ISC-COMPASS and ISC-AHA on the Inventory Management Problem

	ISC-COMPASS				ISC-AHA			
Trial	$\bar{G}(\hat{\mathbf{x}}^*)$	"True"	SE	N	$\bar{G}(\hat{\mathbf{x}}^*)$	"True"	SE	N
1	4,670	4,662	1.9	352,644	4,648	4,626	1.9	209,289
2	4,678	4,658	2.3	411,891	4,588	4,577	2.6	139,882
3	4,679	4,660	2.1	261,829	4,628	4,616	2.4	17,146
4	4,659	4,637	2.8	138,313	4,542	4,533	2.1	8,239
5	4,680	4,664	2.5	301,031	4,516	4,503	1.8	3,588
6	4,683	4,665	2.5	351,666	4,596	4,589	1.0	159,145
7	4,688	4,670	0.8	603,401	4,634	4,627	2.4	418,221
8	4,680	4,665	2.1	411,092	4,606	4,586	2.6	225,085
9	4,668	4,654	2.0	340,945	4,488	4,471	2.6	125,447
10	4,670	4,660	2.5	439,345	4,480	4,458	2.5	227,815
Average	4,676	4,660		361,216	4,573	4,559		153,386

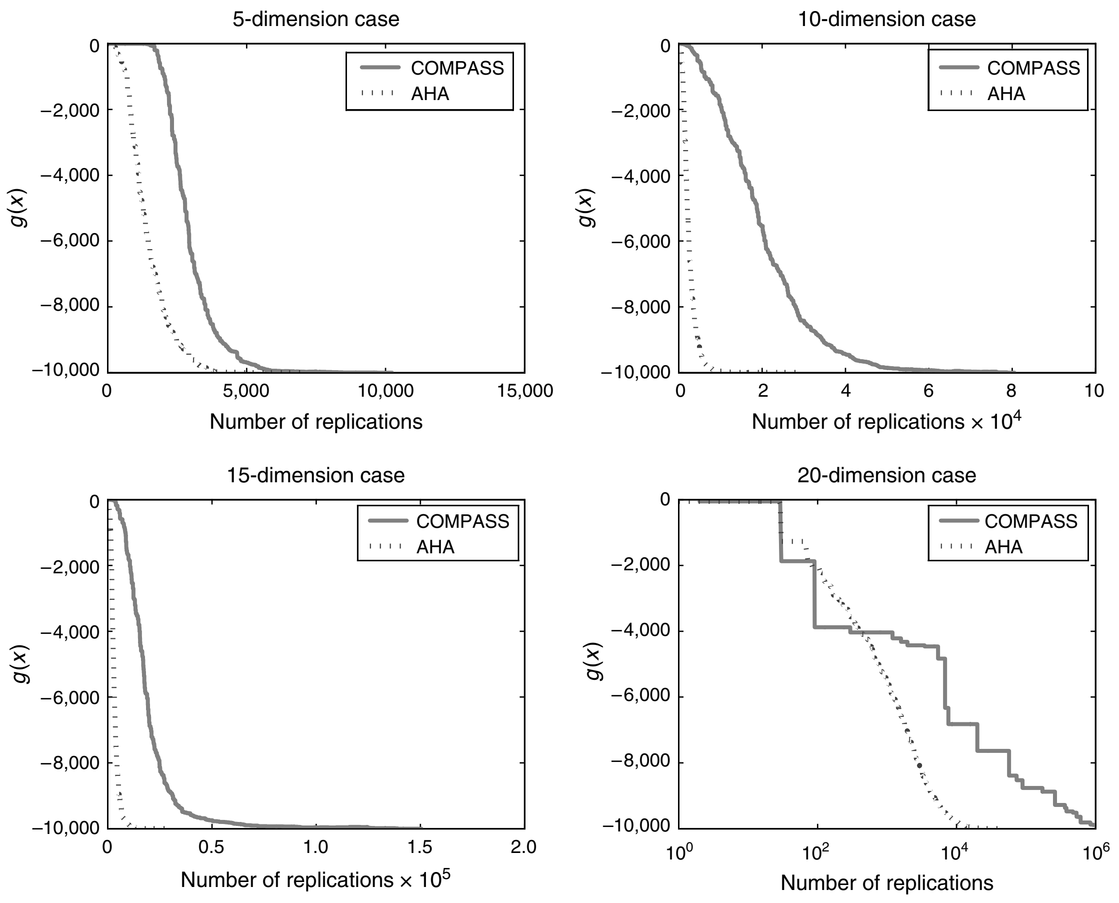

Figure 6 **Performance Plot for the High-Dimensional Test Problem:** $D = 5, 10, 15$, **and** 20

global phase of ISC alleviates the problem. The last test problem examines this concern in higher dimensions. We use the global phase of the ISC framework for this test problem, trying $D = 5$, 10, 15, and 20. The results are shown in Figure 7. We take the average of 20 trials for AHA and COMPASS when $D = 5$, 10, and 15; however, because COMPASS is extremely slow when $D = 20$, we only plot one sample path. Notice that we use a log scale for the horizontal axis when $D = 15$ and $D = 20$. The global optimal solution has an objective value of $-500D$.

We see from the plots that except for $D = 5$, where COMPASS is able to find the global optimal solution eventually, neither COMPASS nor AHA finds the global optimum within the given simulation budget of one million replications. COMPASS tends to have a better average solution quality than AHA in a typical run. But in practice, within the given computation budget, instead of running COMPASS/AHA only once, the decision maker can run the algorithm multiple times and adopt the *best* solution found from multiple runs of the algorithm as the final solution.

Table 3 compares the best solutions ($g(\hat{\mathbf{x}}^*)$) found by ISC-COMPASS and ISC-AHA (stand-alone COMPASS/AHA for high-dimensional problems) among all trials for test problems with known optimal solutions. We also report the average numbers of simulation replications used in each run (N), the mean objective value of solutions ($\bar{g}(\hat{\mathbf{x}}^*)$) returned by COMPASS/AHA on termination for all runs, and the percentages of returned solutions being true local optimum and global optimum. From the table, we clearly see that except for high-dimensional multimodal $D = 5$ problem, the best solutions that ISC-AHA can find are at least as good as ISC-COMPASS. When $D = 20$, AHA is able to achieve about 5% improvement in solution quality. Furthermore, ISC-AHA is able to achieve this with far fewer simulation replications, which is less than 5% of what ISC-COMPASS used for $D = 20$ cases. Both COMPASS and AHA are able to find local optimal solutions with very high probability, and the global phase of the ISC framework largely

Table 2 **The Average CPU Time (Seconds) for High-Dimensional Test Problems**

Variable	$D = 5$	$D = 10$	$D = 15$	$D = 20$	$D = 50$	$D = 100$
COMPASS	12.5600	432	2,716.6	289,122	NA	NA
AHA	0.5856	3.42	8.541	17.3556	463.12	6,456.94

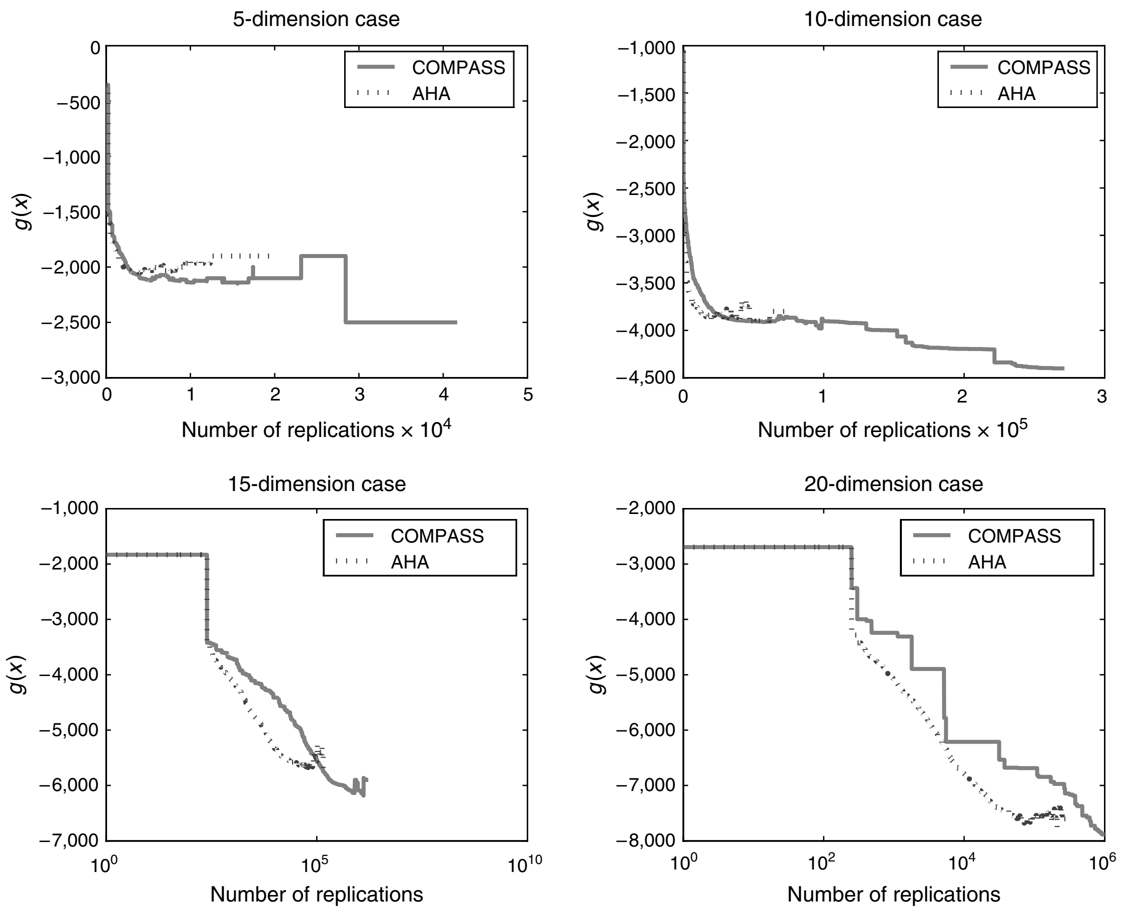

Figure 7 **Performance Plot for the High-Dimensional Multimodal Test Problem:** $D = 5, 10, 15,$ **and** 20

alleviates AHA's tendency to converge too early to poor local optimal solutions, as suggested by the reasonable probabilities to find global optimal solutions. We should point out the low percentages of finding true global/local optima in the Flowline case are due to the relatively larger tolerance level δ used to speed up the search process, as the response surface is quite flat around the optimal solutions. Nevertheless, the average solution quality is almost identical to that of the true global optimum.

In summary, from the results we obtained for six test problems with dimensions from 2 to 100 and varying

Table 3 **The Performance Metrics of AHA and COMPASS**

	ISC-COMPASS					ISC-AHA				
Variable	$g(\hat{\mathbf{x}}^*)$	$\bar{g}(\hat{\mathbf{x}}^*)$	N	Local (%)	Global (%)	$g(\hat{\mathbf{x}}^*)$	$\bar{g}(\hat{\mathbf{x}}^*)$	N	Local (%)	Global (%)
Multimodal	−2.0	−1.990	1.35E4	96	90	−2.0	−1.990	1.33E4	100	82
Singular	1.0	3.160	1.86E4	100	64	1.0	3.160	1.93E4	100	64
Flowline	−5.776	−5.766	1,512	40	40	−5.776	−5.772	1,336	44	44
HD5	-10^4	-10^4	7,295	100	100	-10^4	-10^4	5,350	100	100
HD10	-10^4	-10^4	6.40E4	100	100	-10^4	-10^4	1.59E4	100	100
HD15	-10^4	-10^4	1.26E5	100	100	-10^4	-10^4	2.24E4	100	100
HD20	−9,890	−9,890	10^6	0	0	-10^4	-10^4	2.89E4	100	100
MD5	−2,500	−2,210	1.26E4	100	10	−2,300	−2,090	7,611	100	0
MD10	−4,400	−3,970	9.14E4	100	0	−4,600	−3,850	3.27E4	100	0
MD15	−6,700	−6,140	8.62E5	100	0	−6,700	−5,705	7.80E4	95	0
MD20	−8,000	−8,000	2.75E6	100	0	−8,400	−7,760	1.48E5	100	0

Notes. HDn, high-dimensional test problem with dimension $= n$; MDn, multimodal high-dimensional test problem with dimension $= n$. $g(\hat{\mathbf{x}}^*)$, the objective value of the best solution found; $\bar{g}(\hat{\mathbf{x}}^*)$, the mean objective value of solutions found; N, the average number of simulation replications used; Local/Global, the percentage of returned solutions being true local/global optimum.

difficulty in terms of noise and number of locally optimal solutions, we have the following observations on AHA's performance compared with that of COMPASS:

• AHA achieves performance comparable to that of COMPASS for low-dimensional problems ($D \leq 10$) and is much more efficient in solving high-dimensional problems ($D > 10$).

• AHA converges to locally optimal solutions very quickly and therefore risks being trapped in inferior ones in the presence of multiple locally optimal solutions. However, the global phase of the ISC software largely alleviates this problem.

Therefore, AHA as a locally convergent algorithm is a viable choice as the local search phase of the ISC software when solving high-dimensional DOvS problems.

5. Conclusion

In this paper, we propose an LCRS algorithm for high-dimensional DOvS problems. AHA has performance similar to or slightly inferior to that of COMPASS when dimension is low, say, less than 10. For high-dimensional problems, AHA outperforms COMPASS by a significant margin. Through analyzing the behavior of AHA and COMPASS in simplified settings, we illustrate why AHA is much less affected by dimensionality than COMPASS is.

Numerical experiments have also identified some places where AHA can be further improved. Because AHA is very aggressive in closing toward locally optimal solutions, it may invoke more stopping tests than COMPASS does. It is of vital importance to maintain a proper balance between premature local optimality testing and the time it takes AHA to sample all neighbors of the current sample best solution. The current scheme has reasonable performance but is certainly improvable.

AHA makes use of the LCRS framework proposed by Hong and Nelson (2007). It adopts a hyperbox-shaped MPA geometry but still uses the same uniform random sampling as COMPASS does to sample solutions inside the MPA. Future research will look into other sampling distributions to improve the algorithm's performance.

Acknowledgments

This work was supported by the National Science Foundation under Grant DMI-052185 and the Hong Kong Research Grant Council under Grants GRF613907, GRF613410, and N_HKUST626/10.

References

Alrefaei, M. H., S. Andradóttir. 1999. A simulated annealing algorithm with constant temperature for discrete stochastic optimization. *Management Sci.* **45**(5) 748–764.

Andradóttir, S. 1995. A method for discrete stochastic optimization. *Management Sci.* **41**(12) 1946–1961.

Andradóttir, S., A. A. Prudius. 2009. Balanced explorative and exploitative search with estimation for simulation optimization. *INFORMS J. Comput.* **21**(2) 193–208.

Buzacott, J. A., J. G. Shantikumar. 1993. *Stochastic Models of Manufacturing Systems*. Prentice-Hall, Englewood Cliffs, NJ.

Chen, C.-H., J. Lin, E. Yücesan, S. E. Chick. 2000. Simulation budget allocation for further enhancing the efficiency of ordinal optimization. *Discrete Event Dynam. System* **10**(3) 251–270.

Chick, S. E., K. Inoue. 2001. New two-stage and sequential procedures for selecting the best simulated system. *Oper. Res.* **49**(5) 732–743.

Fu, M. C. 2002. Optimization for simulation: Theory vs. practice. *INFORMS J. Comput.* **14**(3) 192–215.

Fu, M. C., F. W. Glover, J. April. 2005. Simulation optimization: A review, new developments, and applications. *Winter Simulation Conf.*, IEEE Press, New York, 83–95.

Gong, W.-B., Y.-C. Ho, W. Zhai. 1999. Stochastic comparison algorithm for discrete optimization with estimation. *SIAM J. Optim.* **10**(2) 384–404.

Ho, Y. C., R. Sreenivas, P. Vakili. 1992. Ordinal optimization of discrete event dynamic systems. *Discrete Event Dynam. System* **2**(1) 61–88.

Hong, L. J. 2004. Discrete otimization via simulation: Algorithms and error control. Doctoral dissertation, Northwestern University, Evanston, IL.

Hong, L. J., B. L. Nelson. 2006. Discrete optimization via simulation using COMPASS. *Oper. Res.* **54**(1) 115–129.

Hong, L. J., B. L. Nelson. 2007. A framework for locally convergent random search algorithms for discrete optimization via simulation. *ACM Trans. Model. Comput. Simulation* **17**(4) Article 19.

Hu, J., M. C. Fu, S. I. Marcus. 2007. A model reference adaptive search method for global optimization. *Oper. Res.* **55**(3) 549–568.

Mahajan, S., G. van Ryzin. 2001. Stocking retail assortments under dynamic consumer substitution. *Oper. Res.* **49**(3) 334–351.

Nelson, B. L., J. Swann, D. Goldsman, W. Song. 2001. Simple procedures for selecting the best simulated system when the number of alternatives is large. *Oper. Res.* **49**(6) 950–963.

Pichitlamken, J., B. L. Nelson. 2003. A combined procedure for optimization via simulation. *ACM Trans. Model. Comput. Simulation* **13**(2) 155–179.

Shi, L., S. Ólafsson. 2000. Nested partitions method for stochastic optimization. *Method. Comput. Appl. Probab.* **2**(3) 271–291.

Xu, J., B. L. Nelson, J. L. Hong. 2010. Industrial strength COMPASS: A comprehensive algorithm and software for optimization via simulation. *ACM Trans. Model. Comput. Simulation* **20**(1) 1–29.

Yan, D., H. Mukai. 1992. Stochastic discrete optimization. *SIAM J. Control Optimiz.* **30**(3) 594–612.