Accelerated Random Search for Constrained Global Optimization Assisted by Radial Basis Function Surrogates

Luigi Nuñez[a], Rommel G. Regis[a,*], Kayla Varela[b]

[a]*Department of Mathematics, Saint Joseph's University, Philadelphia, PA 19131, USA*
[b]*US Census Bureau, Suitland, MD 20746, USA*

Abstract

The Accelerated Random Search (ARS) algorithm (Appel et al. 2004) is a stochastic algorithm for the global optimization of black-box functions subject only to bound constraints. ARS iteratively samples uniformly within a box centered at the current best point. If the sample point is worse than the current best point, then the size of the box is reduced, thereby making it more likely to generate an improving solution if the current best point is not yet a local minimum. Otherwise, if the sample point is an improvement over the current best point, then the size of the box is reset to the initial value so that the box covers the entire search space. ARS has been shown theoretically and numerically to converge to the global minimum faster than the Pure Random Search (PRS) algorithm on bound-constrained problems. This article develops the Constrained ARS (CARS) algorithm, which is an extension of ARS for optimization problems with black-box inequality constraints. Under some mild conditions on the constrained optimization problem, we prove the convergence to the global minimum in a probabilistic sense of a class of stochastic search algorithms called Scattered Uniform Random Search (SCAT-URS) algorithms, which includes CARS as a special case. To improve performance of CARS on computationally expensive simulation-based problems, we incorporate Radial Basis Function (RBF) surrogate models to approximate the objective and constrained functions, and refer to the resulting algorithm as CARS-RBF. Numerical experiments show that CARS consistently and substantially outperforms both Pure Random Search (PRS) and the Accelerated Particle Swarm Optimization (APSO) algorithm (Yang 2010) on 31 test problems with dimensions ranging from 2 to 20. Moreover, CARS-RBF is an improvement over CARS, outperforms a sequential penalty derivative-algorithm, and competes with ConstrLMSRBF (Regis 2011) on the same test problems. Finally, sensitivity analysis of CARS-RBF to the type of RBF model used shows that the cubic RBF model generally yields the best results on the test problems among six types of RBF models considered, including the thin plate spline, Gaussian and multiquadric models.

Keywords: Constrained global optimization, random search, expensive optimization, surrogate model, radial basis function

*corresponding author (rregis@sju.edu)

1. Introduction

In many practical engineering optimization problems, the objective function and the constraint functions are black-box in the sense that their mathematical expressions and their gradients are not explicitly available. In many cases, these function values are obtained by running computer simulations, which in some cases are computationally expensive. Among the most widely used approaches for these problems are stochastic algorithms, including simulated annealing, evolutionary algorithms, and swarm intelligence approaches (Yang [1], Zabinsky [2], Zhigljavsky and Žilinskas [3]). The most basic of these stochastic algorithms is the Pure Random Search (PRS) algorithm, which simply involves iteratively generating sample points uniformly at random throughout the search space and keeping track of the best solution obtained. It is well-known that PRS converges to the global minimum in a probabilistic sense (e.g., see Solis and Wets [4], Zabinsky [2], Spall [5]). However, it is also well-known to be inefficient (i.e., the number of iterations increases exponentially with the dimension of the problem). Thus, PRS is typically not the method of choice of practitioners.

Appel et al. [6] developed the Accelerated Random Search (ARS) algorithm for bound-constrained global optimization and proved that, under certain mild conditions, it converges to the global optimum faster than PRS. ARS also uses a uniform distribution to select sample points, but selects them within a box centered at the current best point. The initial size of this box is chosen so that it covers the entire search space. Then, if the chosen sample point is *not* an improvement over the current best point, the size of the box is reduced. The idea is that with a smaller box, the chances of obtaining an improving solution is increased if the current best point is not yet a local minimum. On the other hand, if the sample point yields an improvement, then the size of the box is reset to the initial size to maintain the ability to reach any region of the search space.

Many stochastic algorithms for constrained black-box global optimization have been proposed, including heuristics, but relatively few come with convergence guarantees. Examples of provably convergent stochastic search algorithms for constrained global optimization are found in Baba [7] and Pinter [8]. One of the main goals of this article is to extend the ARS algorithm to problems with black-box inequality constraints. Specifically, this article proposes the *Constrained Accelerated Random Search Algorithm (CARS)* for the following constrained optimization problem:

$$\begin{array}{c} \min\ f(x) \\ \text{s.t.} \\ G(x) = (g_1(x), \ldots, g_m(x)) \leq 0 \\ \ell \leq x \leq u \end{array} \tag{1}$$

Here, $\ell, u \in \mathbb{R}^d$ and the region $[\ell, u]$ defined by the bounds is referred to as the *search space* for problem (1). Moreover, the functions $f : \mathbb{R}^d \to \mathbb{R}$ and $g_j : \mathbb{R}^d \longrightarrow \mathbb{R},\ j = 1, \ldots, m$ are black-box in that their mathematical forms are not explicitly available and instead their values are obtained via a deterministic computer simulation. Also, one *simulation* for a given input vector $x \in [\ell, u]$ yields the value $f(x)$ and all the components of $G(x)$. In addition, the derivatives of the objective and constraint functions are assumed to be unavailable. This problem is referred to as $CBOP(f, G, [\ell, u])$ in Regis [9].

For simplicity, assume also that there are no equality constraints in (1) and that the feasible region $\mathcal{D} := \left\{x \in \mathbb{R}^d : \ \ell \le x \le u, \ G(x) \le 0\right\}$ has a nonempty interior. Note that if $\mathcal{D}$ is compact and f is continuous over $\mathcal{D}$, then an optimal solution to the above $CBOP(f, G, [\ell, u])$ problem is guaranteed to exist. For example, the compactness of $\mathcal{D}$ follows if the constraint functions $g_1, \ldots, g_m$ are all continuous. Future work will deal with black-box equality constraints and the presence of noise in the objective and/or constraint functions. Under certain reasonable assumptions, CARS can also be proved to converge to the global minimum of problem (1) using one of the results in Regis [9].

Another characteristic of many real-world simulation-based optimization problems is that these simulations are often computationally expensive. As a result, the running time of a typical optimization algorithm on these problems is completely dominated by the total time spent on the simulations. These problems are found in many engineering applications, particularly those involving finite element or computational fluid dynamics simulations (e.g., Husain et al. [10], Kontoleontos et al. [11]). In this situation, a widely-used approach is to use surrogate models or metamodels (or response surface models) to approximate the objective and constraint functions (e.g., Ky et al. [12]). Hence, another main goal of this paper is to use surrogate models in combination with the proposed CARS algorithm to improve its performance on computationally expensive problems. In particular, we develop the CARS-RBF algorithm, which uses Radial Basis Function (RBF) surrogates to approximate the objective and constraint functions in the proposed CARS algorithm.

The proposed CARS and CARS-RBF algorithms are compared with various alternative methods on 31 test problems involving 2 to 20 decision variables and 1 to 38 inequality constraints. The alternative methods include a PRS algorithm for constrained problems (CPRS), a particle swarm optimization (PSO) algorithm for constrained problems called APSO (Yang [1]), a sequential penalty derivative-free algorithm called SDPEN (Liuzzi et al. [13]), an RBF-assisted CPRS (CPRS-RBF), and ConstrLMSRBF (Regis [14]). The results show that CARS is consistently and substantially better than CPRS and APSO on the test problems considered. Moreover, CARS-RBF is a substantial improvement over CARS. Moreover, it is much better than both CPRS-RBF and SDPEN, and it is competitive with ConstrLMSRBF. In all, both CARS and CARS-RBF are promising methods for constrained black-box optimization, and there is an added benefit that CARS converges to the global minimum in a probabilistic sense under some mild assumptions.

The rest of the paper is organized as follows. Section 2 gives a review of the literature while Section 3 presents the ARS method for bound constrained problems. Section 4 develops the CARS method while Section 5 uses surrogate models, particularly RBF models, to improve the performance of CARS. Section 6 provides a theoretical guarantee of convergence to the global minimum in a probabilistic sense for the class of Scattered Uniform Random Search algorithms, which includes CARS as a special case. Next, Section 7 explains the setup of the numerical experiments, and then, Section 8 provides numerical results and discussions. Finally, Section 9 gives a summary and some conclusions.

2. Review of Literature

Numerous algorithms have been proposed for constrained optimization problems where the objective function and the constraint functions are all black-box. Among these are direct search methods such as Mesh Adaptive Direct Search (MADS) and its variants (Audet and Dennis [15, 16], Le Digabel [17]) and SDPEN (sequential penalty derivative-free algorithm) (Liuzzi et al. [13]. Another class of methods for constrained optimization are trust region algorithms (Conn et al. [18], Powell [19]). Many of these methods can be proved to converge to a first order critical point. However, since the goal is global optimization, one needs to run these algorithms from multiple starting points to improve the chances or to guarantee that the global minimum will eventually be found.

Among the most popular methods for constrained black-box optimization include heuristic methods such as evolutionary algorithms and swarm intelligence algorithms (Yang [1]). In the context of heuristics, one of the most common technique for constraint handling uses penalty functions (e.g., [20], [21], [22]). Other constraint handling methods involve multi-objective optimization [23], a combination of a bi-objective optimization and a penalty approach ([24], [25]), the epsilon constrained method [26], cultural algorithms [27], and those that distinguish between feasible and infeasible solutions [28]. One limitation of these heuristic approaches is that there are seldom any theoretical convergence guarantees to finding an optimal solution.

When the objective and/or constraint functions are computationally expensive, surrogate models (also known as metamodels or response surface models) have been used to approximate these functions as part of a global optimization algorithm (Forrester et al. [29], Ky et al. [12]). In particular, the objective and/or constraint functions are evaluated at an initial set of points from a space-filling design for the given search space. Then, initial surrogates for the objective and/or constraints are built using the data points obtained. These surrogates are used to guide the selection of new points where the objective and constraints are evaluated. Next, the surrogates are updated after the new data points are obtained, and then the method iterates. Examples of surrogate-based methods that use surrogates of the objective function only include the Efficient Global Optimization (EGO) method by Jones et al. [30] and the Radial Basis Function (RBF) methods by Gutmann [31]. There are also Adaptive Response Correction (ARC) methods (Koziel and Leifsson [32]) that exploit physics-based surrogates, which incorporate some knowledge about the system of interest. Moreover, some algorithms that use surrogates to approximate constraint functions include ConstrLMSRBF (Regis [14]), Kriging-based Constrained Global Optimization (KCGO) (Li et al. [33]), GOSAC (Müller and Woodbury [34]), and the Kriging-based methods by Boukouvala and Ierapetritou [35] and by Feliot et al. [36]. There is also a hybrid method that uses Kriging with the expected improvement (EI) criterion combined with the augmented Lagrangian framework (Gramacy et al. [37]). Another possibility is to use surrogates within a multistart framework to identify promising starting points for a nonlinear programming local solver (Krityakierne and Shoemaker [38]).

Surrogates are used to approximate the black-box objective and/or constraint functions in derivative-free trust region methods (e.g., Powell [19], Regis and Wild [39]) and also in direct search methods (e.g., Abramson et al. [40]). Moreover, surrogates have been used to assist various heuristics. Examples of surrogate models that have been used with evolutionary algorithms include multivariate quadratic polynomials ([41], [42]), multilayer perceptron neural networks [43], kriging and Gaussian process models ([44], [45]), radial basis functions ([46], [47], [45]), support vector machines (SVM) ([48], [49], [42]) and nearest neighbors regression [50]. Multiple surrogates may also be used to balance exploration and exploitation in an evolutionary algorithm (e.g., [51]). In addition, surrogates have been used for multi-fidelity models to assist an evolutionary algorithm (Liu et al. [52]. A survey on surrogate-assisted EAs is provided by Jin [53].

Penalty functions are also commonly used to handle constraints in surrogate-assisted EAs. For example, Shi and Rasheed [42] use a stochastic penalty function and an adaptive mechanism for switching from lower complexity polynomial models to higher complexity SVM models while Runarsson [54] uses a penalty-based Stochastic Ranking ES combined with a nearest neighbor regression model. However, incorporating a penalty function and using a surrogate to approximate either the penalty function or the objective function plus penalty might not be the most effective way to handle expensive black-box constraints since information about individual constraint violations is lost (Powell [19]). Instead, Powell [19] suggests treating the constraints separately by building individual surrogates, one for each constraint. In fact, Regis [14, 47] provided numerical evidence that using surrogates to approximate a penalized objective function is *not* as effective as using separate surrogates for the constraints, and hence, we adopt the latter approach in this paper.

3. Accelerated Random Search for Bound Constrained Optimization

Before presenting our proposed algorithms, we first describe the *Pure Random Search (PRS)* and *Accelerated Random Search (ARS)* (Appel et al. [6]) algorithms for bound constrained optimization.

Suppose we wish to minimize $f(x)$ over a search space $[\ell, u]$ defined by bound constraints. In the classic *Pure Random Search (PRS)* algorithm, we select iterates uniformly at random over the search space $[\ell, u]$. These iterates are treated as d-dimensional i.i.d. random vectors $\{Y_n\}_{n\geq 1}$ where the $Y_n \sim U[\ell, u]$ for all $n \geq 1$. Let $\{X_n\}_{n\geq 1}$ be the sequence of best solutions obtained by PRS. Then, it can be shown that $f(X_n) \to f^*$ a.s. (almost surely) (e.g., see Spall [5]). However, the number of iterations required to find the global minimum within some level of accuracy is exponential in the dimension of the problem.

The *Finite Descent Accelerated Random Search (ARS)* algorithm (Appel et al. [6]) is similar to PRS except that we select the iterates uniformly at random from shrinking neighborhoods around the current best solution whenever they do not result in an improvement in the objective function value. Moreover, we re-initialize the search neighborhood to the whole search space whenever the iterate results in an improvement, or whenever the search radius falls below some threshold. Because of these re-initializations, there is always a subsequence of iterations for which the algorithm samples uniformly at random over the entire search space. Hence, it can also be proved that Finite Descent ARS converges to the global minimum almost surely (e.g., Appel et al. [6] and also using Theorem 4 in Regis [55]). More importantly, under certain conditions, Appel et al. [6] proved that the sequence of best objective function values obtained by Finite Descent ARS converges, with probability one, to the global minimum of a continuous objective function faster than that of PRS. In addition, these results were verified numerically using a series of test problems in Appel et al. [6].

Because of the theoretical convergence guarantees as well as good numerical results in the bound constrained case, it seems natural and promising to consider extensions of Finite Descent ARS to the constrained case. To the best of our knowledge, this has not been carried out in the literature. To extend Finite Descent ARS to constrained black-box optimization, we use the concept of a *constraint violation function* defined in Regis [9].

Definition 3.1. *Let $[\ell, u] \subseteq \mathbb{R}^d$ be the search space and let $G(x)$ be the constraint function for problem (1). Moreover, let $\mathcal{D} = \left\{x \in \mathbb{R}^d : \ \ell \leq x \leq u, \ G(x) \leq 0\right\}$ be the feasible region of the problem. A constraint violation function for G over $[\ell, u]$ is a function $V : [\ell, u] \to \mathbb{R}^+$ satisfying the following conditions:*

(i) $V(x) = 0$ for all $x \in \mathcal{D}$;
(ii) $V(x) > 0$ for all $x \notin \mathcal{D}$; and
(iii) If $G(x) \leq G(y)$, then $V(x) \leq V(y)$.

A constraint violation function is intended to measure of the degree of constraint violation of a point in the search space $[\ell, u]$. Note that feasible solutions are points in $[\ell, u]$ whose constraint violation function value is 0 while infeasible solutions are points whose constraint violation function value is strictly positive. Moreover, if a point $x \in [\ell, u]$ has better constraint function values than $y \in [\ell, u]$, then x has a smaller constraint violation function value than y. When comparing solutions in the search space, we use the following rules: (i) feasible solutions are always better than infeasible ones; (ii) if two points are feasible, then the better solution is that with the smaller objective function value; and (iii) if two points are infeasible, then the better point is that with the smaller constraint violation function value.

It is easy to verify that $V(x) = \sum_{j=1}^{m} [\max\{g_j(x), 0\}]^q$, where $q > 0$, and $V(x) = \sum_{j=1}^{m} I([g_j(x) > 0])$, where $I(\cdot)$ is the indicator function, are constraint violation functions for G over $[\ell, u]$. Commonly used examples are $V(x) = \sum_{j=1}^{m} [\max\{g_j(x), 0\}]$ and $V(x) = \sum_{j=1}^{m} [\max\{g_j(x), 0\}]^2$.

4. Accelerated Random Search for Constrained Black-Box Optimization

This section presents the *Constrained Accelerated Random Search (CARS)* algorithm, which is an extension of Finite Descent ARS (Appel et al. [6]) to constrained black-box optimization. Because CARS is meant to handle constraints, it needs a constraint violation function to rank potential solutions within the search space. In this study, we use $V(x) = \sum_{j=1}^{m} [\max\{g_j(x), 0\}]^2$. As with Finite Descent ARS, we select the iterate in CARS uniformly at random within a search neighborhood of the current best solution. Here, the current best solution is determined by the objective function and our chosen constraint violation function. The search neighborhood is reduced whenever the iterate does *not* improve the current best solution. It is re-initialized to the entire search space whenever the iterate results in an improvement in the current best solution or whenever the radius of the search neighborhood falls below some threshold value.

A detailed description of CARS is given below. This algorithm is suitable for optimization problems that involve black-box objective and black-box constraint functions, i.e., an optimization problem of the form (1). At the end of this section, we provide a theorem that states that, under certain conditions, the sequence of best solutions obtained by CARS converges to the global minimum in a probabilistic sense.

In the notation below, $[0,1]^d$ is the search space, f is the objective function, $G : [\ell, u] \to R^m$ is the vector-valued function whose components are the individual constraint functions $g_1, \ldots, g_m$, $V : [\ell, u] \to \mathbb{R}^+$ is the constraint violation function, and n is the counter for the number of simulations, where one simulation yields the value of the objective function and the values of all the m constraint functions at one input vector. Note that the assumption that the search space is the unit hypercube $[0,1]^d$ is not restrictive since any constrained optimization problem with a search space of the form $[\ell, u] \subset R^d$ can be easily transformed to an equivalent problem whose search space is the unit hypercube $[0,1]^d$.

Also, $\| \cdot \|_\infty$ denotes the ∞-norm or sup-norm on $\mathcal{D}$, which is defined as follows: For each $x = (x^{(1)}, \ldots, x^{(d)}) \in \mathbb{R}^d$, $\|x\|_\infty = \max\{|x^{(1)}|, \ldots, |x^{(d)}|\}$. Moreover, $B(x,r) = \{y \in \mathcal{D} : \|y - x\|_\infty \leq r\}$ denotes the closed ∞-norm ball of radius r centered at x in $\mathcal{D}$. Because we are using the ∞-norm, $B(x,r)$ is simply a box centered at x with side length $2r$. It is possible to use other norms to define the ball $B(x,r)$, but we use ∞-norm as in the original implementation of ARS for bound constrained optimization. In addition, X_n denotes the current best solution after n simulations, r_n is the radius of the ∞-norm ball around X_n where the next sample point Y_n is chosen. The simulation is run at Y_n to yield the objective and constraint function values at that point.

Constrained Accelerated Random Search (CARS)

Inputs:

1. A $CBOP(f, G, [0,1]^d)$ problem of the form (1) satisfying the assumptions in Section 1. Assume that the values of f and G are obtained in one computer simulation and that the simulator will not crash for any input in $x \in [0,1]^d$.
2. A constraint violation function $V : [0,1]^d \to \mathbb{R}^+$ for G over $[0,1]^d$. In the implementation, $V(x) = \sum_{j=1}^m [\max\{g_j(x), 0\}]^2$.
3. A starting point, $X_1 \in [0,1]^d$. (This may or may not be feasible.)
4. Contraction factor, $c > 1$
5. Precision threshold, $\rho > 0$
6. The maximum number of simulations allowed, n_{max}

Output: The best point encountered by the algorithm.

Step 0 (Evaluate Initial Sample Point) Set $n = 1$, $r_1 = 1$ and run simulation to obtain $f(X_1)$ and $G(X_1) = (g_1(X_1), \ldots, g_m(X_1))$. Calculate $V(X_1)$.

Step 1 (Generate Sample Point) Given $r_n > 0$, generate Y_n from a uniform distribution on $B(X_n, r_n) \cap [0,1]^d$, where $B(X_n, r_n) = \{y \in \mathbb{R}^d \ : \ \|y - X_n\|_\infty \le r_n\}$.

Step 2 (Evaluate Sample Point) Run simulation to obtain the values of $f(Y_n)$ and $G(Y_n) = (g_1(Y_n), \ldots, g_m(Y_n))$. Calculate $V(Y_n)$.

Step 3 (Update Best Solution and Search Radius) If $V(Y_n) < V(X_n)$ or if $(V(Y_n) = V(X_n)$ and $f(Y_n) < f(X_n))$, then

let $X_{n+1} = Y_n$ and $r_{n+1} = 1$,

Else

let $X_{n+1} = X_n$ and $r_{n+1} = \dfrac{r_n}{c}$.

If $r_{n+1} \le \rho$, then let $r_{n+1} = 1$.

Step 4 (Increment Evaluation Counter and Check Termination) Increment $n := n + 1$. If the termination condition is satisfied (e.g., $n = n_{max}$), then stop and return X_n, $f(X_n)$ and $G(X_n)$; otherwise go back to Step 1.

In Step 0, CARS runs a simulation to obtain the objective and constraint function values, as well as the constraint violation value, at the initial point X_1. It also initializes the search radius and the counter for the number of simulations, denoted by n. Again one simulation at an input vector yields the values of the objective function and all constraint functions. In Step 1, a new sample point, denoted by Y_n is chosen from an ∞-norm ball centered at X_n with radius r_n. The simulation is then run at the new sample point in Step 2. Next, the center of the search box and the radius are updated in Step 3. In particular, if the new sample point is an improvement over the previous best solution, then it becomes the new center and the radius is re-initialized to 1. On the other hand, if there is no improvement in the current best solution, the current center remains and the radius is reduced by a *contraction factor* $c > 1$. Here, an improvement occurs if the constraint violation function value of the new sample point is better than that of the current best solution ($V(Y_n) < V(X_n)$) or, if this is not the case, the objective function value of the new sample point is better than that of the current best solution ($V(Y_n) = V(X_n)$ and $f(Y_n) < f(X_n)$). Finally, in Step 4, the simulation counter is incremented and the algorithm checks if the termination condition has been satisfied (e.g., the computational budget has been reached). If so, it returns the current center (the current best solution) and its objective and constraint function values. Otherwise, it goes back to Step 1. In black box global optimization, a widely used termination criterion is the use of a fixed number of function evaluations or simulations. However, other termination criteria are also used. For example, in cases where the global minimum value or a lower bound for it is known, the algorithm can be terminated after reaching a specified target value.

5. Accelerated Random Search Assisted by Surrogate Models

5.1. Algorithm Description

To improve the performance of the CARS algorithm on computationally expensive constrained black-box optimization problems, a natural approach is to build and maintain response surface or surrogate models that approximate the objective and inequality constraint functions. Then, the algorithm generates multiple trial points in each iteration and identifies the most promising according to the surrogate models for the objective and constraint functions (and also the resulting surrogate for the constraint violation function). This most promising trial point then becomes the new sample point where the simulation is run to obtain the objective and constraint function values. The resulting algorithmic framework will be referred to as *Constrained Accelerated Random Search assisted by Response Surface models (CARS-RS)* and is described below.

As before, the search space is assumed to be $[0,1]^d$ (after a suitable transformation of the variables), f is the objective function, $G : [0,1]^d \to \mathbb{R}^m$ is the vector-valued inequality constraint function where $G(x) = (g_1(x), \ldots, g_m(x))$, $V : [0,1]^d \to \mathbb{R}^+$ is the constraint violation function, and n is the counter for the number of simulations. Moreover, we use the same notation as in CARS with some additions. For example, Y_n still denotes the nth sample point where the simulation is run to yield the objective and constraint function values, and X_n denotes the current best solution after n simulations. However, now Y_n is selected from a set of trial points $\Omega_n = \{Y_{n,1}, \ldots, Y_{n,t}\}$ that are chosen uniformly at random over the region $B(X_n, r_n) \cap [0,1]^d$. In addition, $s_n^f(y)$ and $s_n^G(y) = (s_n^{g_1}(y), \ldots, s_n^{g_m}(y))$ denote the surrogate models for the objective and constraint functions built from all available sample points after n simulations.

For simplicity, we assume in this section that a feasible starting point is given as was done in Regis [14]. This assumption is not unreasonable since it is often the case in practical scenarios that a feasible design to the problem is known and one is simply looking for an improved solution. Future work will deal with the absence of feasible starting points.

Constrained Accelerated Random Search assisted by Response Surface models (CARS-RS)

Inputs:

1. A $CBOP(f, G, [0,1]^d)$ problem of the form (1) satisfying the assumptions in Section 1. Assume that the values of f and G are obtained in one computer simulation and that the simulator will not crash for any input in $x \in [0,1]^d$.
2. A constraint violation function $V : [0,1]^d \to \mathbb{R}^+$ for G over $[0,1]^d$. In the implementation, $V(x) = \sum_{j=1}^{m} [\max\{g_j(x), 0\}]^2$.
3. A set of initial sample points $\mathcal{I} = \{Y_1, Y_2, \ldots, Y_{n_0}\} \subseteq [0,1]^d$ that includes a feasible starting point Y_1 (the other points are not required to be feasible) and that contains a set of $d+1$ affinely independent points (and so, $n_0 \geq d+1$).

4. Contraction factor, $c > 1$
5. Precision threshold, $\rho > 0$
6. The maximum number of simulations allowed, n_{max}
7. A particular response surface model, e.g., a cubic RBF model with a linear polynomial tail (see Section 5.2)
8. The number of candidate points in each iteration, t

Output: The best point encountered by the algorithm.

Step 0 (Evaluate Initial Sample Points) For each $i = 1, \ldots, n_0$, evaluate $f(Y_i)$ and $G(Y_i) = (g_1(Y_i), \ldots, g_m(Y_i))$. Set $n = n_0$, $r_{n_0} = 1$, and set X_{n_0} to be the best feasible point among $Y_1, \ldots, Y_{n_0}$.

Step 1 (Generate Sample Point)

> **Step 1.1** *(Fit/Update Surrogate Models)* Fit or update the surrogate model $s_n^f(y)$ for the objective function and the vector-valued surrogate model $s_n^G(y) = (s_n^{g_1}(y), \ldots, s_n^{g_m}(y))$ for the m inequality constraint functions, using all available data points
> $\mathcal{P}_n = \{(Y_i, f(Y_i), G(Y_i)) : i = 1, \ldots, n\}$.
>
> **Step 1.2** *(Generate Multiple Random Trial Points)* Given $r_n > 0$, randomly generate *trial points* $\Omega_n = \{Y_{n,1}, \ldots, Y_{n,t}\}$ from a uniform distribution over $B(X_n, r_n) \cap [0,1]^d$, where $B(X_n, r_n) = \{y \in \mathbb{R}^d \ : \ \|y - X_n\|_\infty \leq r_n\}$.
>
> **Step 1.3** *(Collect Valid Trial Points)* Use the surrogate models for the inequality constraint functions (i.e., $s_n^G(y)$) to identify the trial points in Ω_n that are predicted to be feasible or that have the minimum number of predicted constraint violations. Let Ω_n^{valid} be the collection of such trial points. These points are called the *valid trial points*.
>
> **Step 1.4** *(Select New Sample Point)* Using the information from the surrogate models for the objective and constraint functions $s_n^f(y)$ and $s_n^G(y)$ and from the data points $\mathcal{P}_n$, select the sample point Y_n from the trial points in Ω_n^{valid}.

Step 2 (Evaluate Sample Point) Run simulator to obtain the values of $f(Y_n)$ and $G(Y_n) = (g_1(Y_n), \ldots, g_m(Y_n))$.

Step 3 (Update Best Solution and Search Radius) If $V(Y_n) = 0$ and $f(Y_n) < f(X_n)$, then

> let $X_{n+1} = Y_n$ and $r_{n+1} = 1$,

Else

let $X_{n+1} = X_n$ and $r_{n+1} = \frac{r_n}{c}$.

If $r_{n+1} \leq \rho$, then let $r_{n+1} = 1$.

Step 4 (Increment Evaluation Counter and Check Termination) Increment $n := n + 1$. If the termination condition is satisfied (e.g., $n = n_{max}$), then stop and return X_n; otherwise go back to Step 1.

In Step 0, every initial sample point is evaluated. The best point so far is identified, and the radius is set to the initial value of 1 as before. Note that CARS-RS requires a set $\mathcal{I}$ of sample points to fit the initial response surface model. In our implementation, we require that $\mathcal{I}$ contains a set of $d+1$ affinely independent points, so that $|\mathcal{I}| = n_0 \geq d+1$. This can be accomplished, for example, by randomly generating space-filling designs such as Latin hypercube designs of size n_0 until we obtain one where the P matrix in Equation (3) in Section 5.2 has rank $d + 1$.

Step 1 generates the next sample point by performing the following steps. Step 1.1 fits or updates the response surface model using the objective and constraint values of the sample points obtained so far. In Step 1.2, multiple trial points are generated from a uniform distribution over the intersection of the ∞-norm ball around the current best sample point and the search space $[0, 1]^d$. The number of trial points, denoted by t, is typically set to a large number to allow the algorithm to examine a diverse collection of candidate points. In Step 1.3, the response surface models for the inequality constraint functions are used to determine which of the trial points are predicted to be feasible and collects them. If there are no trial points predicted to be feasible, then Step 1.3 collects those with the minimum number of predicted constraint violations. The collected points are called the *valid trial points*. Step 1.4 selects the best valid trial point according to the response surface models for the objective and constraint functions. This selected point, which will be the new sample point Y_n, is the point with the lowest predicted objective function value if there are trial points predicted to be feasible or the point with the smallest predicted constraint violation value if none of the trial points are predicted to be feasible.

Once the new sample point Y_n has been selected, the algorithm proceeds as in the CARS algorithm. That is, Step 2 evaluates the objective and constraint functions at this sample point. Then, Step 3 updates the information about the best sample point and resizes the radius depending on how the selected sample point Y_n compares to the current best sample point. Finally, in Step 4, the counter for the simulations is incremented and the algorithm verifies if the termination condition has been met. For example, the algorithm checks if the maximum number of simulations allowed has been reached. If the termination condition has been satisfied, the best sample point X_n is returned. Otherwise, the algorithm goes back to Step 1.

5.2. Radial Basis Function Interpolation

The proposed surrogate-assisted CARS algorithm can be implemented using any type of surrogate model that can approximate the functions very well. Among the most widely used surrogate modeling techniques include radial basis functions (RBF) and kriging (or Gaussian process modeling). We use the RBF interpolation model from [56], which has been successfully used to in various surrogate-based methods (e.g., [31, 14, 57]).

Next, we describe this surrogate modeling technique. Assume that n distinct points $x_1, \ldots, x_n \in \mathbb{R}^d$ and their objective function values $f(x_1), \ldots, f(x_n)$ are given. This modeling technique uses an interpolating function of the form

$$s(x) = \sum_{i=1}^{n} \lambda_i \phi(\|x - x_i\|) + p(x), \ x \in \mathbb{R}^d, \tag{2}$$

where $\|\cdot\|$ is the Euclidean norm, $\lambda_i \in \mathbb{R}$ for $i = 1, \ldots, n$, $p(x)$ is a linear polynomial in d variables, and ϕ has the *cubic* form: $\phi(r) = r^3$. The function ϕ can take other forms such as the thin plate spline ($\phi(r) = r^2 \log r$), Gaussian ($\phi(r) = \exp(-(r/\gamma)^2)$, and multiquadric ($\phi(r) = \sqrt{r^2 + \gamma^2}$), where γ is a parameter. A cubic RBF model is used here because it has been successfully used in various surrogate-based and surrogate-assisted optimization algorithms (e.g., [14], [58]). An advantage of this cubic RBF model over the Gaussian RBF model is that it is simpler because it does not require an extra parameter. The parameter γ in the Gaussian RBF is typically found using leave-one-out cross-validation and this adds to the computational cost of fitting the model. Moreover, numerical experiments with the Gaussian RBF shows that, for some choices of the γ parameter, the Gaussian RBF model sometimes has many more local minima than the black-box function that it is trying to approximate (e.g., [47]). In contrast, this issue did not seem to come up for the cubic RBF model.

To fit the above RBF model, define the matrix $\Phi \in \mathbb{R}^{n \times n}$ by: $\Phi_{ij} := \phi(\|x_i - x_j\|), \quad i, j = 1, \ldots, n$. Also, define the matrix $P \in \mathbb{R}^{n \times (d+1)}$ so that its i^{th} row is $[1, x_i^T]$. Now, the RBF model that interpolates the points $(x_1, f(x_1)), \ldots, (x_n, f(x_n))$ is obtained by solving the system

$$\begin{pmatrix} \Phi & P \\ P^T & 0_{(d+1)\times(d+1)} \end{pmatrix} \begin{pmatrix} \lambda \\ c \end{pmatrix} = \begin{pmatrix} F \\ 0_{d+1} \end{pmatrix}, \tag{3}$$

where $0_{(d+1)\times(d+1)} \in \mathbb{R}^{(d+1)\times(d+1)}$ is a matrix of zeros, $F = (f(x_1), \ldots, f(x_n))^T$, $0_{d+1} \in \mathbb{R}^{d+1}$ is a vector of zeros, $\lambda = (\lambda_1, \ldots, \lambda_n)^T \in \mathbb{R}^n$ and $c = (c_1, \ldots, c_{d+1})^T \in \mathbb{R}^{d+1}$ consists of the coefficients for the linear polynomial $p(x)$. It is well-known that the coefficient matrix in (3) is invertible if and only if $\text{rank}(P) = d + 1$ (e.g., see [56]). This condition is equivalent to having a subset of $d + 1$ affinely independent points among the points $\{x_1, \ldots, x_n\}$. Once there are $d + 1$ affinely independent points among the sample points, the above P matrix will always have rank $d + 1$ throughout the entire run of the algorithm.

The above RBF model can also be used to construct a surrogate for each of the m inequality constraint functions. Hence, we will build/update $m + 1$ surrogates in every iteration. For a given set of data points where the objective and constraint function values are known, the same interpolation matrix is used so fitting multiple RBF models can be done relatively quickly even when m is large by means of standard matrix factorizations. The resulting surrogated-assisted CARS algorithms will be referred to as CARS-RBF.

6. Convergence

Recall that the search radius in CARS is reset to the initial value whenever an iteration improves the best feasible objective function value or whenever the search radius falls below the threshold ρ. Moreover, when the search radius is at the initial value, the search neighborhood covers the entire search space $[\ell, u]$. Consequently, there is always a subsequence of iterations for which CARS samples uniformly at random over the entire search space $[\ell, u]$. We now prove the convergence of CARS to the global minimum by proving the convergence of a general class of random search algorithms where a subsequence of iterations sample uniformly at random throughout the entire search space defined by the bound constraints. This class of algorithms will be referred to as *Scattered Uniform Random Search (SCAT-URS)* and its general framework is presented below.

Scattered Uniform Random Search (SCAT-URS) for Constrained Optimization
Inputs:

(i) CBOP$(f, G, [\ell, u])$;

(ii) CV function $V_G : [\ell, u] \longrightarrow \mathbb{R}_+$;

(iii) indices of iterations $\{n_k\}_{k\geq 1}$ when uniform random search is performed on entire search space;

(iv) probability distributions for generating sample points for other iterations.

Step 0. Set $n = 1$.
Step 1. Generate realization of random vector $Y_n : (\Omega, \mathcal{B}) \longrightarrow (\mathbb{R}^d, \mathcal{B}(\mathbb{R}^d))$: If $n = n_k$ for some $k \geq 1$, then $Y_n \sim U[\ell, u]$; otherwise, generate Y_n according to the specified probability distribution whose outcome is in $[\ell, u]$.
Step 2. Evaluate $f(Y_n)$ and $G(Y_n)$. (This is equivalent to one simulation.)
Step 3. Let X_n be the best point found so far with respect to $f(x)$ and $V_G(x)$ as defined in (4).
Step 4. Increment $n \leftarrow n + 1$ and go back to Step 1.

To prove the convergence of SCAT-URS algorithms for constrained optimization, we begin by recalling the definitions of convergence in probability and almost sure (a.s.) convergence (e.g., see [59]). Also, recall that convergence a.s. implies convergence in probability but the converse is false. In the definitions below, $\{X_n\}_{n\geq 1}$ is a sequence of random vectors and X is another random vector defined on the same probability space $(\Omega, \mathcal{B}, P)$, where Ω is the sample space, $\mathcal{B}$ is a σ-field of subsets of Ω, and P is the probability measure. Moreover, $\|\cdot\|$ denotes the 2-norm. In addition, given a collection of random elements $\mathcal{E}$ defined on a probability space, let $\sigma(\mathcal{E})$ be the σ-field generated by $\mathcal{E}$.

Definition 6.1. *Let $\{X_n\}_{n\geq 1}$ and X be random vectors defined on the same probability space $(\Omega, \mathcal{B}, P)$. The sequence $\{X_n\}_{n\geq 1}$ converges in probability to X iff for any $\epsilon > 0$,*

$\lim_{n\to\infty} P[\|X_n - X\| > \epsilon] = 0$. *Moreover,* $\{X_n\}_{n\geq 1}$ *converges almost surely (a.s.) to* X*, written* $X_n \longrightarrow X$ *a.s., iff there exists an event* $\mathcal{N} \in \mathcal{B}$ *such that* $P(\mathcal{N}) = 0$ *and* $\lim_{n\to\infty} X_n(\omega) = X(\omega)$ *for all* $\omega \in \Omega \setminus \mathcal{N}$*.*

Because a SCAT-URS algorithm is stochastic, its iterates are treated as d-dimensional random vectors $\{Y_n\}_{n\geq 1}$ defined on a probability space $(\Omega, \mathcal{B}, P)$ and whose realizations are on the search space $[\ell, u]$. Here, the random vector $Y_n : (\Omega, \mathcal{B}) \longrightarrow (\mathbb{R}^d, \mathcal{B}(\mathbb{R}^d))$ represents the nth trial iterate generated by the uniform distribution over an ∞-norm ball of radius r_n (hypercube of side length $2r_n$) centered at the current best solution. Moreover, the sequence $\{X_n\}_{n\geq 1}$ represent the best points visited by the algorithm with respect to the objective function $f(x)$ and a constraint violation function $V_G(x)$ as follows: set $X_1 := Y_1$, and for any $n > 1$, set

$$X_n := \begin{cases} Y_n & \text{if } Y_n \text{ and } X_{n-1} \text{ are feasible and } f(Y_n) < f(X_{n-1}) \\ Y_n & \text{if } X_{n-1} \text{ is infeasible and } V_G(Y_n) < V_G(X_{n-1}) \\ X_{n-1} & \text{otherwise} \end{cases} \tag{4}$$

In addition, in the proofs below, $\sigma(\mathcal{E}_n)$ is the σ-field generated by the random elements in $\mathcal{E}_n = \{Y_1, \ldots, Y_n\}$ for all $n \geq 1$.

SCAT-URS algorithms, including CARS, follow the GARSCO framework in Regis [9], and hence under certain mild conditions they can be proved to converge to the global minimum in a probabilistic sense. For completeness and to make this paper self-contained, we provide a convergence proof that is tailored for SCAT-URS algorithms.

Before we state and prove our convergence result, we first prove the following lemmas. For convenience, define the following subset of the feasible region $\mathcal{D}$ consisting of *ϵ-optimal solutions* of a CBOP$(f, G, [\ell, u])$ for any given $\epsilon > 0$:

$$\mathcal{S}_\epsilon := \{x \in \mathcal{D} : f(x) < f^* + \epsilon\}. \tag{5}$$

Moreover, let int$(\mathcal{D})$ denote the interior of $\mathcal{D}$ and let bd$(\mathcal{D})$ denote the boundary of $\mathcal{D}$.

Lemma 6.1. *Let* $[\ell, u] \subset \mathbb{R}^d$ *be a hypercube. For any* $\delta > 0$*,*

$$\psi_{[\ell,u]}(\delta) := \inf_{z\in[\ell,u]} \mu(B(z,\delta) \cap [\ell, u]) \geq \left(\frac{\min\{\delta, \frac{1}{2}\|u - \ell\|_\infty\}}{2}\right)^d \frac{\pi^{d/2}}{\Gamma(\frac{d}{2} + 1)} > 0.$$

Here, $B(z, \delta)$ *is the 2-norm ball centered at* z *of radius* δ*,* μ *is the Lebesgue measure on* $\mathbb{R}^d$*, and* Γ *is the gamma function defined by* $\Gamma(n) := \displaystyle\int_0^\infty x^{n-1} e^{-x} dx$*.*

Proof. This follows from Proposition 1 in Regis [55]. □

Lemma 6.2. *Consider a CBOP*$(f, G, [\ell, u])$ *where* f *is a continuous function and the feasible region is* $\mathcal{D} = \{x \in \mathbb{R}^d \mid \ell \leq x \leq u,\ G(x) \leq 0\}$ *compact. Moreover, assume that* $\mathcal{D}$ *has* *a nonempty interior and that every neighborhood of a boundary point of $\mathcal{D}$ intersects the interior of $\mathcal{D}$. Let $\{Y_n\}_{n\geq 1}$ be the sequence of iterates of a SCAT-URS algorithm applied to the CBOP$(f, G, [\ell, u])$ and let $\{X_n\}_{n\geq 1}$ be the sequence of best solutions. Furthermore, let $\{Y_{n_k}\}_{k\geq 1}$ be the subsequence of iterates where $Y_{n_k} \sim U[\ell, u]$, then for every $\epsilon > 0$, there exists $0 < L(\epsilon) < 1$ such that for all $k \geq 1$,*

$$P[Y_{n_k} \in \mathcal{S}_\epsilon \mid \sigma(\mathcal{E}_{(n_k)-1})] \geq L(\epsilon),$$

where $\mathcal{S}_\epsilon$ is defined in (5) and $\sigma(\mathcal{E}_{(n_k)-1})$ is the σ-field generated by the random elements in $\mathcal{E}_{(n_k)-1}$.

Proof. Let x^* be a global minimizer of the CBOP. Fix $\epsilon > 0$ and an integer $k \geq 1$. Since f is continuous at x^*, $\exists\ \delta(\epsilon) > 0$ such that $|f(x) - f(x^*)| < \epsilon$ whenever $\|x - x^*\| < \delta(\epsilon)$. Hence,

$$\begin{aligned} P[Y_{n_k} \in \mathcal{S}_\epsilon \mid \sigma(\mathcal{E}_{(n_k)-1})] &= P[Y_{n_k} \in \mathcal{D} \text{ and } f(Y_{n_k}) < f(x^*) + \epsilon \mid \sigma(\mathcal{E}_{(n_k)-1})] \\ &\geq P[Y_{n_k} \in \mathcal{D} \text{ and } \|Y_{n_k} - x^*\| < \delta(\epsilon) \mid \sigma(\mathcal{E}_{(n_k)-1})] \\ &= P[Y_{n_k} \in B(x^*, \delta(\epsilon)) \cap \mathcal{D} \mid \sigma(\mathcal{E}_{(n_k)-1})]. \end{aligned}$$

Next, since $\mathcal{D}$ is closed, we have $\mathcal{D} = \text{int}(\mathcal{D}) \cup \text{bd}(\mathcal{D})$. We consider two cases: $x^* \in \text{int}(\mathcal{D})$; and $x^* \in \text{bd}(\mathcal{D})$. First, suppose $x^* \in \text{int}(\mathcal{D})$. Then $\exists 0 < \eta_1 \leq \delta(\epsilon)$, where η_1 depends on $\delta(\epsilon)$ such that $B(x^*, \eta_1) \subseteq \mathcal{D}$. Now since uniform random probabilities are obtained using the Lebesgue measure, observe that

$$\begin{aligned} P[Y_{n_k} \in B(x^*, \delta(\epsilon)) \cap \mathcal{D} \mid \sigma(\mathcal{E}_{(n_k)-1})] &\geq P[Y_{n_k} \in B(x^*, \eta_1) \cap [\ell, u] \mid \sigma(\mathcal{E}_{(n_k)-1})] \\ &= \mu(Y_{n_k} \in B(x^*, \eta_1) \cap [\ell, u]) \geq \psi_{[\ell,u]}(\eta_1) =: L_1(\epsilon), \end{aligned}$$

where the last inequality follows from the definition within Lemma 6.1.

Next, suppose $x^* \in \text{bd}(\mathcal{D})$. By assumption, $\exists\ z \in \text{int}(\mathcal{D}) \cap B(x^*, \delta(\epsilon))$, where z depends on $\delta(\epsilon)$. Since $z \in \text{int}(\mathcal{D})$, $\exists 0 < \eta_2 \leq \delta(\epsilon)$, where η_2 also depends on z and $\delta(\epsilon)$, such that $B(z, \eta_2) \subseteq \mathcal{D}$. Again,

$$\begin{aligned} P[Y_{n_k} \in B(x^*, \delta(\epsilon)) \cap \mathcal{D} \mid \sigma(\mathcal{E}_{(n_k)-1})] &\geq P[Y_{n_k} \in B(z, \eta_2) \cap [\ell, u] \mid \sigma(\mathcal{E}_{(n_k)-1})] \\ &= \mu(Y_{n_k} \in B(z, \eta_2) \cap [\ell, u]) \geq \psi_{[\ell,u]}(\eta_2) =: L_2(\epsilon). \end{aligned}$$

Define

$$L(\epsilon) := \begin{cases} L_1(\epsilon) & \text{if } x^* \in \text{int}(\mathcal{D}) \\ L_2(\epsilon) & \text{if } x^* \in \text{bd}(\mathcal{D}) \end{cases}$$

Clearly, $0 < L(\epsilon) < 1$ and

$$P[Y_{n_k} \in \mathcal{S}_\epsilon \mid \sigma(\mathcal{E}_{(n_k)-1})] \geq P[Y_{n_k} \in B(x^*, \delta(\epsilon)) \cap \mathcal{D} \mid \sigma(\mathcal{E}_{(n_k)-1})] \geq L(\epsilon).$$

□

We are now ready to state and prove our convergence result for the class of Scattered Uniform Random Search (SCAT-URS) algorithms for constrained optimization.

Theorem 6.1. *Consider a CBOP$(f, G, [\ell, u])$ where f is a continuous function and the feasible region $\mathcal{D} = \{x \in \mathbb{R}^d \mid \ell \leq x \leq u,\ G(x) \leq 0\}$ is compact. Moreover, assume that $\mathcal{D}$ has a nonempty interior and that every neighborhood of a boundary point of $\mathcal{D}$ intersects the interior of $\mathcal{D}$. Let $\{X_n\}_{n\geq 1}$ be the sequence of best solutions produced by CARS when run indefinitely and let $f^* := \min_{x\in\mathcal{D}} f(x)$. Then $f(X_n) \longrightarrow f^*$ almost surely (a.s.).*

The requirement in the above theorem that every neighborhood of a boundary point of $\mathcal{D}$ intersects the interior of $\mathcal{D}$ is *not* restrictive. In particular, this requirement is satisfied if $\mathrm{cl}(\mathrm{int}(\mathcal{D})) = \mathrm{cl}(\mathcal{D})$, where $\mathrm{cl}(\cdot)$ denotes the closure of a set in $\mathbb{R}^d$ (see Proposition 2.4 in Regis [9]). For example, this condition is satisfied if $\mathcal{D}$ is a convex set with a nonempty interior. However, convexity of $\mathcal{D}$ is not necessary and it is easy to create examples that satisfy this condition where $\mathcal{D}$ is nonconvex.

Proof. Let $\{n_k\}_{k\geq 1}$ be the indices of the iterations where CARS samples the entire search space uniformly, i.e., $Y_{n_k} \sim U[\ell, u]$ for all $k \geq 1$. Fix $\epsilon > 0$. From Lemma 6.2, $\exists\, 0 < L(\epsilon) < 1$ such that

$$P[Y_{n_k} \in \mathcal{S}_\epsilon \mid \sigma(\mathcal{E}_{(n_k)-1})] \geq L(\epsilon), \text{ for any } k \geq 1. \tag{6}$$

Now for each $k \geq 1$, we have

$$\begin{aligned} P[Y_{n_1} \notin \mathcal{S}_\epsilon, Y_{n_2} \notin \mathcal{S}_\epsilon, \ldots, Y_{n_k} \notin \mathcal{S}_\epsilon] &= \textstyle\prod_{i=1}^k P[Y_{n_i} \notin \mathcal{S}_\epsilon | Y_{n_1} \notin \mathcal{S}_\epsilon, Y_{n_2} \notin \mathcal{S}_\epsilon, \ldots, Y_{n_{(i-1)}} \notin \mathcal{S}_\epsilon] \\ &= \textstyle\prod_{i=1}^k \left(1 - P[Y_{n_i} \in \mathcal{S}_\epsilon | Y_{n_1} \notin \mathcal{S}_\epsilon, Y_{n_2} \notin \mathcal{S}_\epsilon, \ldots, Y_{n_{(i-1)}} \notin \mathcal{S}_\epsilon]\right). \end{aligned}$$

From (6), we obtain

$$P[Y_{n_i} \in \mathcal{S}_\epsilon | Y_{n_1} \notin \mathcal{S}_\epsilon, Y_{n_2} \notin \mathcal{S}_\epsilon, \ldots, Y_{n_{(i-1)}} \notin \mathcal{S}_\epsilon] \geq L(\epsilon).$$

Thus,

$$P[Y_{n_1} \notin \mathcal{S}_\epsilon, Y_{n_2} \notin \mathcal{S}_\epsilon, \ldots, Y_{n_k} \notin \mathcal{S}_\epsilon] \leq (1 - L(\epsilon))^k.$$

Next, observe that if $X_{n_k} \notin \mathcal{S}_\epsilon$, then $Y_{n_1} \notin \mathcal{S}_\epsilon, Y_{n_2} \notin \mathcal{S}_\epsilon, \ldots, Y_{n_k} \notin \mathcal{S}_\epsilon$. Hence, for each $k \geq 1$,

$$P[f(X_{n_k}) - f^* \geq \epsilon] \leq P[X_{n_k} \notin \mathcal{S}_\epsilon] \leq P[Y_{n_1} \notin \mathcal{S}_\epsilon, Y_{n_2} \notin \mathcal{S}_\epsilon, \ldots, Y_{n_k} \notin \mathcal{S}_\epsilon] \leq (1 - L(\epsilon))^k.$$

This implies that $\lim_{k\to\infty} P[f(X_{n_k}) - f^* \geq \epsilon] = 0$, and so, $f(X_{n_k}) \longrightarrow f^*$ in probability. Moreover, $f(X_{n_{k(i)}}) \longrightarrow f^*$ a.s. as $i \to \infty$ for some subsequence $\{n_{k(i)}\}_{i\geq 1}$ (e.g., see [59], Theorem 6.3.1(b)). Hence, $\exists \mathcal{H} \subseteq \Omega$ such that $P(\mathcal{H}) = 0$ and $\lim_{i\to\infty} f(X_{n_{k(i)}}(\omega)) = f^*$ for all $\omega \in \Omega \setminus \mathcal{H}$.

Next, we define

$$\mathcal{I} := \{\omega \in \Omega \ :\ X_{n_k}(\omega) \notin \mathcal{D} \text{ for all } k\} = \bigcap_{k=1}^{\infty} [X_{n_k} \notin \mathcal{D}],$$

and show that $P(\mathcal{I}) = 0$. Note that for all $k \geq 1$,

$$P(\mathcal{I}) = P(\bigcap_{j=1}^{\infty}[X_{n_j} \notin \mathcal{D}]) \leq P(X_{n_k} \notin \mathcal{D}) \leq P[X_{n_k} \notin \mathcal{S}_\epsilon] \leq (1 - L(\epsilon))^k,$$

where the last inequality follows from the same argument as above. Hence,

$$P(\mathcal{I}) \leq \lim_{k \to \infty} (1 - L(\epsilon))^k = 0,$$

and so, $P(\mathcal{I}) = 0$.

Now, since $P(\mathcal{I}) = P(\mathcal{H}) = 0$, we have $P(\mathcal{I} \cup \mathcal{H}) = 0$. Next, fix $\omega \in \Omega \setminus (\mathcal{I} \cup \mathcal{H})$. Since $\omega \in \Omega \setminus \mathcal{I}$, $\exists k(\omega)$ such that $X_{n_{k(\omega)}}(\omega) \in \mathcal{D}$. This implies that $X_n(\omega) \in \mathcal{D}$ for all $n \geq n_{k(\omega)}$. Since $\{X_n(\omega)\}_{n \geq n_{k(\omega)}}$ is the sequence of best feasible solutions for $n \geq n_{k(\omega)}$, the sequence $\{f(X_n(\omega))\}_{n \geq n_{k(\omega)}}$ is monotonically non-increasing. Moreover, $f(X_n(\omega)) \geq f^*$ for all $n \geq n_{k(\omega)}$. Hence, $\lim_{n \to \infty} f(X_n(\omega))$ exists. Also, since $\omega \in \Omega \setminus \mathcal{H}$, we have that $\lim_{i \to \infty} f(X_{n_{k(i)}}(\omega)) = f^*$. This shows that $\lim_{n \to \infty} f(X_n(\omega)) = f^*$. Thus, $f(X_n) \longrightarrow f^*$ a.s. $\square$

Theorem 6.1 proves the almost sure (a.s.) convergence to the global minimum of the CARS algorithm without surrogates. It may be possible to prove the convergence of CARS assisted by surrogates, but this will be the subject of future work. In general, the use of surrogates is expected to improve the convergence rate of CARS and this is confirmed by the numerical results on most of the test problems in Section 8. However, the numerical results on a few test problems also show that surrogates do *not* always result in faster convergence. This could be because in the early iterations, the surrogates might not approximate the objective or constraint functions well in the region where the sample points are selected, especially on problems that are pathological. In any case, we still expect that CARS with surrogates can be proved to converge though its convergence rate might not always be faster.

7. Numerical Experiments

7.1. Test Problems

We compare the performance of the proposed CARS and CARS-RBF algorithms with alternative optimization methods on 31 widely used benchmark test problems. Many of these test problems are from Michalewicz and Schoenauer [60] and the CEC 2010 Competition on Constrained Real-Parameter Optimization (Mallipeddi and Suganthan [61]). These test problems were used in Hedar and Fukushima [62], Mezura-Montes and Cetina-Dominguez [63], Regis ([14, 57]), and other studies. For some of the problems, we modified the objective function or constraint functions by applying a strictly increasing transformation to avoid extremely large values as was done in Regis ([14, 57]). Note that applying such a transformation does not change the location of the local and global optima of the problem. Table 1 summarizes the characteristics of these test problems. The convexity or nonconvexity of the objective functions and the feasible regions were determined analytically or numerically. In a few cases, it was not straightforward to determine whether the feasible region is convex or nonconvex and we simply indicated that it is compact. Note that majority of the problems have nonconvex objective functions and nonconvex feasible regions.

Four of the test problems are engineering design problems and these are the WB4 (*Welded Beam Design Problem*) (Deb [64], Coello Coello and Mezura-Montes [65]), PVD4 (*Pressure Vessel Design Problem*) (Coello Coello and Mezura-Montes [65]), GTCD4 (*Gas Transmission Compressor Design Problem*) (Beightler and Phillips [66]), and SR7 (*Speed Reducer Design* for small aircraft engine) (Floudas and Pardalos [67]).

Ten of the test problems are taken from a collection of well-known constrained optimization test problems by Michalewicz and Schoenauer [60]. These are labeled G1, G2, G3MOD, G4, G5MOD, G6, G7, G8, G9, and G10. The five test problems G13MOD, G16, G18, G19 and G24 are taken from Mezura-Montes and Cetina-Dominguez [63]. The G3MOD, G5MOD and G13MOD problems were obtained from the original G3, G5 and G13 problems by replacing all equality constraints with $\leq$ inequality constraints. In addition, another test problem was due to Hesse [68].

Finally, eleven of the test problems come from the CEC 2010 Competition on Constrained Real-Parameter Optimization (Mallipeddi and Suganthan [61]) (`http://www3.ntu.edu.sg/home/epnsugan/index_files/CEC10-Const/TR-April-2010.pdf`). These are C01, C02MOD, C05MOD, C06MOD, C07, C08, C09MOD, C10MOD, C14, C15, and C17MOD. As before, the problems labeled MOD are obtained from the corresponding original version by replacing all equality constraints with $\leq$ inequality constraints. These benchmark problems are scalable, and we used the 10-dimensional instances of these problems.

The objective and constraint functions for the above optimization problems are not really expensive to evaluate. However, as was done in Regis [14, 57], we can still conduct meaningful comparisons of performance of the different algorithms for constrained black-box optimization by pretending that the objective and constraint functions are computationally expensive. This can be done by keeping track of the best feasible objective function values obtained by the different algorithms as the number of simulations (objective and constraint function evaluations) increase. In the truly computationally expensive setting where each simulation could possibly take hours, the overhead run times of surrogate-based algorithms (e.g., total run time minus the total time spent on all the simulations in one optimization run) would only be a very small fraction of the total run times of the algorithms (see Regis ([14, 57] to get an idea of overhead run times of surrogate-based algorithms). Hence, the relative performance of algorithms on these test problems are expected to be similar to the relative performance of these algorithms on truly expensive problems whose objective and constraint functions have the same general form or shape as the objective and constraint functions of our test problems.

Table 1: Constrained optimization test problems (d is the number of decision variables, m is the number of inequality constraints).

Test Problem	d	m	Search Space	Objective Function Characteristics	Feasible Region Characteristics
G6	2	2	$[13, 100] \times [0, 100]$	cubic polynomial	nonconvex
G8	2	2	$[0, 10]^2$	nonconvex	convex
WB4 (Welded Beam Design)	4	6	$[0.125, 10] \times [0.1, 10]^3$	nonconvex polynomial	nonconvex
GTCD4 (Gas Transmission Compressor Design)	4	1	$[20, 50] \times [1, 10] \times [20, 50] \times [0.1, 60]$	nonconvex	nonconvex
PVD4 (Pressure Vessel Design)	4	3	$[0, 1]^2 \times [0, 50] \times [0, 240]$	nonconvex polynomial	compact
G5MOD	4	5	$[0, 1200]^2 \times [-0.55, 0.55]^2$	convex polynomial	compact
G4	5	6	$[78, 102] \times [33, 45] \times [27, 45]^3$	nonconvex polynomial	nonconvex
Hesse	6	6	$[0, 5] \times [0, 4] \times [1, 5]$ $\times [0, 6] \times [1, 5] \times [0, 10]$	concave quadratic	nonconvex
SR7 (Speed Reducer Design)	7	11	$[2.6, 3.6] \times [0.7, 0.8] \times [17, 28]$ $\times [7.3, 8.3]^2 \times [2.9, 3.9] \times [5.0, 5.5]$	nonconvex polynomial	compact
G9	7	4	$[-10, 10]^7$	nonconvex polynomial	compact
G10	8	6	$[10^2, 10^4] \times [10^3, 10^4]^2 \times [10, 10^3]^5$	linear	nonconvex
G2	10	2	$[0, 10]^{10}$	nonconvex	convex
G7	10	8	$[-10, 10]^{10}$	convex quadratic	compact
G3MOD	20	1	$[0, 1]^{20}$	nonconvex polynomial	closed convex ball
G1	13	9	$[0, 1]^9 \times [0, 100]^3 \times [0, 1]$	concave quadratic	convex polyhedron
G13MOD	5	3	$[-2.3, 2.3]^2 \times [-3.2, 3.2]^3$	nonconvex	nonconvex
G16	5	38	$[704.4148, 906.3855] \times [68.6, 288.88]$ $\times [0, 134.75] \times [193, 287.0966]$ $\times [25, 84.1988]$	nonconvex	nonconvex
G18	9	13	$[-10, 10]^8 \times [0, 20]$	nonconvex polynomial	nonconvex
G19	15	5	$[0, 10]^{15}$	convex polynomial	convex polyhedron
G24	2	2	$[0, 3] \times [0, 4]$	linear	disconnected
C01	10	2	$[0, 10]^{10}$	nonconvex	convex
C02MOD	10	3	$[-5.12, 5.12]^{10}$	convex	nonconvex
C05MOD	10	2	$[-600, 600]^{10}$	convex	nonconvex
C06MOD	10	2	$[-600, 600]^{10}$	convex	nonconvex
C07	10	1	$[-140, 140]^{10}$	nonconvex polynomial	nonconvex
C08	10	1	$[-140, 140]^{10}$	nonconvex polynomial	nonconvex
C09MOD	10	1	$[-500, 500]^{10}$	nonconvex polynomial	nonconvex
C10MOD	10	1	$[-500, 500]^{10}$	nonconvex polynomial	nonconvex
C14	10	3	$[-1000, 1000]^{10}$	nonconvex polynomial	nonconvex
C15	10	3	$[-1000, 1000]^{10}$	nonconvex polynomial	nonconvex
C17MOD	10	3	$[-10, 10]^{10}$	convex polynomial	nonconvex

7.2. Alternative Methods and Experimental Setup

As mentioned earlier CARS will be compared with Constrained Pure Random Search (CPRS), which is an extension of Pure Random Search to inequality constrained problems, and with the Accelerated Particle Swarm Optimization (APSO) algorithm by Yang [1]. Moreover, CARS-RBF will be compared with CARS to assess the effect of using RBF surrogates on the latter. It will also be compared with ConstrLMSRBF (Regis [14]) and with an RBF-assisted extension of CPRS called CPRS-RBF. Both ConstrLMSRBF and CPRS-RBF are methods that use RBF surrogates to approximate the objective and constraint functions. Finally, CARS-RBF will also be compared with SDPENm, which is a Matlab implementation of a sequential penalty derivative-free algorithm by Liuzzi et al. [13].

Each algorithm is run on each test problem for 30 trials. Each trial has a computational budget of $100(d+1)$ simulations, where d is the dimension of the test problem. The number of points in the initial space-filling design is $2(d+1)$, and the number of random space-filling designs to generate in each trial is 100. All numerical computations are performed in Matlab 9.2.0 using an Intel Core i5-3210M CPU 2.50 GHz MacBook Pro.

8. Results and Discussion

8.1. Comparisons with Alternative Methods in Terms of Best Feasible Values Found

Table 2 shows the mean best feasible value found by each algorithm on each test problem after 30 trials with a computational budget of $100(d+1)$ simulations per trial, where d is the dimension of the problem. The number in the parenthesis next to the mean best value is the standard error of the sample mean. All results shown in the table are rounded to four decimal places.

Since ARS is shown to outperform PRS both theoretically and experimentally on bound constrained problems, it is expected that CARS and CPRS, as extensions that solve constrained problems, will show similar relative performances. Table 2 confirms this belief; CARS does better than CPRS on 29 out of 31 test problems. Table 2 also shows that CARS outperforms a state-of-the-art algorithm such as APSO ([1]). CARS does better than APSO on 19 out of 31 test problems. Given the property of guaranteed convergence to the global minimum in probability as well as good experimental results, CARS is a promising method for constrained black-box global optimization.

The addition of surrogate models is expected to improve these results. It is expected that CARS-RBF and CPRS-RBF will perform better than CARS and CPRS, respectively. Again, Table 2 confirms this expectation. CARS-RBF outperforms CARS on 29 out of 31 test problems. Note also that CPRS-RBF does better than CPRS on 30 of the 31 test problems. Thus, Table 2 confirms that the addition of RBF surrogate models improves performance and that CARS-RBF substantially outperforms CARS.

Test Problem	CARS	CPRS	APSO	CARS-RBF	CPRS-RBF	ConstrLMSRBF	SDPENm
G6	-4.2491e+03 (2.3984e+02)	-3.3347e+03 (2.3885e+02)	-3.3765e+03 (2.3826e+02)	-6.8478e+03 (1.0763e+01)	-6.1040e+03 (7.8529e+01)	-6.8843e+03 (1.0650e+01)	-3.3136e+03 (2.4380e+02)
G8	-8.5874e-02 (4.3859e-03)	-1.7458e-02 (3.9055e-03)	-5.3426e-02 (4.6505e-03)	-6.2234e-02 (6.2338e-03)	-8.6539e-02 (3.9757e-03)	-4.5896e-02 (5.5923e-03)	-5.6894e-02 (6.3295e-03)
G24	-5.4736e+00 (5.5785e-03)	-5.1136e+00 (3.1563e-02)	-5.3040e+00 (1.6342e-02)	-5.5061e+00 (2.6636e-04)	-5.4979e+00 (9.8807e-04)	-5.3249e+00 (8.6699e-02)	-5.0213e+00 (7.8673e-02)
G5MOD	5.3059e+03 (2.3254e+01)	7.0125e+03 (1.1974e+02)	5.8545e+03 (3.5952e+01)	5.1307e+03 (6.8687e-01)	5.2860e+03 (1.1029e+01)	5.1303e+03 (5.3444e-01)	5.6887e+03 (5.1396e+01)
WB4	3.7551e+00 (2.0304e-01)	5.9186e+00 (2.2342e-01)	4.7761e+00 (1.9328e-01)	2.4539e+00 (2.9738e-02)	3.3966e+00 (8.9805e-02)	2.3002e+00 (1.0454e-02)	3.8683e+00 (2.4455e-01)
PVD4	6.6177e+03 (9.6494e+01)	7.7194e+03 (1.5705e+02)	6.5064e+03 (4.0839e+01)	5.8699e+03 (7.6198e+00)	6.0477e+03 (1.1440e+01)	5.8958e+03 (1.1520e+01)	5.8938e+03 (1.3780e+01)
GTCD4	3.1515e+06 (2.5598e+04)	4.9665e+06 (1.3930e+05)	3.4497e+06 (1.8993e+04)	3.0026e+06 (7.4556e+03)	3.0284e+06 (5.2810e+03)	3.0033e+06 (9.3224e+03)	3.6774e+06 (8.1200e+04)
G4	-3.0056e+04 (5.4570e+01)	-2.9851e+04 (4.7797e+01)	-3.0366e+04 (3.1169e+01)	-3.0664e+04 (1.0583e-01)	-3.0512e+04 (6.5526e+00)	-3.0664e+04 (1.0408e-01)	-3.0587e+04 (1.7593e+01)
G13MOD	1.1706e-02 (3.3846e-03)	1.2294e-01 (2.1910e-02)	1.7294e-02 (4.3537e-03)	3.5761e-03 (1.3628e-05)	5.2518e-03 (1.2874e-04)	1.3888e-02 (3.4056e-03)	2.5392e-01 (2.9946e-02)
G16	-1.4964e+00 (1.5548e-02)	-1.2372e+00 (2.9459e-02)	-1.4207e+00 (2.6359e-02)	-1.9032e+00 (1.4206e-04)	-1.7177e+00 (6.3602e-03)	-1.9032e+00 (1.8738e-04)	-1.5246e+00 (2.5875e-02)
Hesse	-2.4941e+02 (2.2547e+00)	-2.1829e+02 (3.5010e+00)	-2.4881e+02 (6.6183e+00)	-2.9030e+02 (1.4442e+00)	-2.8265e+02 (8.3145e-01)	-2.7642e+02 (8.2340e+00)	-2.7295e+02 (8.7140e+00)
G9	7.1678e+02 (7.2765e+00)	1.1388e+05 (7.3226e+04)	7.5151e+02 (4.4613e+00)	7.7888e+02 (1.8201e+01)	9.7861e+02 (2.3571e+01)	9.9629e+02 (4.6514e+01)	8.5724e+02 (1.0174e+02)
SR7	3.0928e+03 (8.1756e+00)	4.2808e+03 (1.3476e+02)	3.0285e+03 (2.1960e+00)	3.0056e+03 (7.1960e-01)	3.0755e+03 (3.1320e+00)	3.0039e+03 (7.2614e-01)	2.9944e+03 (2.4786e-06)
G10	1.3821e+04 (5.3130e+02)	1.9938e+04 (6.6102e+02)	1.6694e+04 (5.8020e+02)	7.6393e+03 (9.1632e+01)	1.3513e+04 (2.9508e+02)	7.2959e+03 (2.9906e+01)	1.0566e+04 (3.1847e+02)
G18	-5.9215e-01 (1.7677e-02)	-2.3243e-01 (1.7915e-02)	-2.5322e-01 (1.7599e-02)	-8.1354e-01 (1.4906e-02)	-2.3243e-01 (1.7915e-02)	-8.2248e-01 (1.3001e-02)	-6.2217e-01 (2.7918e-02)
G2	-3.1334e-01 (1.0130e-02)	-2.5736e-01 (5.4936e-03)	-3.5652e-01 (1.8363e-02)	-5.4267e-01 (1.5301e-02)	-4.6623e-01 (5.1095e-03)	-3.3069e-01 (1.7471e-02)	-4.3263e-01 (7.7502e-03)
G7	9.8160e+01 (1.5856e+01)	1.4923e+03 (1.5521e+02)	3.2783e+02 (7.6895e+01)	2.4647e+01 (1.9148e-02)	5.0061e+02 (4.7844e+01)	2.4560e+01 (1.2624e-02)	3.1379e+01 (1.0978e+00)
C01	-3.2124e-01 (1.1112e-02)	-2.5358e-01 (5.3333e-03)	-4.1732e-01 (1.8342e-02)	-4.5715e-01 (1.3609e-02)	-4.1669e-01 (8.7688e-03)	-2.9030e-01 (1.0710e-02)	-3.6128e-01 (1.6921e-02)
C02MOD	-1.3469e+00 (1.2549e-01)	9.1605e-01 (9.9798e-02)	-1.8036e+00 (2.7753e-02)	-2.2114e+00 (3.1975e-02)	-1.0889e+00 (3.3414e-02)	-2.0195e+00 (6.6629e-02)	-3.7407e-01 (9.4392e-02)
C05MOD	-3.0722e+02 (1.5864e+01)	4.9781e+01 (1.4947e+01)	-3.7210e+02 (4.7749e+00)	-4.7165e+02 (4.0030e+00)	-2.9240e+02 (6.5636e+00)	-4.0531e+02 (2.1516e+01)	-5.4104e+01 (5.2330e+01)
C06MOD	-1.8557e+02 (1.0771e+01)	5.4491e+01 (1.0833e+01)	-2.7518e+02 (1.7099e+01)	-3.6983e+02 (1.7466e+01)	-1.5177e+02 (3.7059e+00)	-3.2976e+02 (2.2255e+01)	4.4243e+01 (3.6501e+01)
C07	5.4764e+04 (2.0013e+04)	4.9675e+09 (3.7937e+08)	1.0884e+06 (1.1809e+05)	3.6011e+05 (7.4414e+04)	8.1804e+08 (1.3970e+08)	1.0590e+06 (8.9464e+04)	1.6520e+03 (6.0242e+02)
C08	2.0875e+06 (1.4001e+06)	5.2327e+09 (4.8470e+08)	3.0506e+06 (5.2235e+05)	5.7753e+05 (1.5980e+05)	8.8208e+08 (1.4292e+08)	2.0133e+06 (3.0348e+05)	3.1226e+03 (8.8623e+02)
C09MOD	8.7872e+06 (4.0528e+06)	7.3026e+11 (6.2102e+10)	1.3755e+08 (1.6906e+07)	3.6364e+07 (7.0368e+06)	6.2069e+10 (9.4480e+09)	1.7092e+08 (1.1831e+07)	5.5028e+03 (1.7349e+03)
C10MOD	1.7769e+07 (1.1099e+07)	8.3362e+11 (7.3201e+10)	2.7822e+08 (3.7106e+07)	3.5231e+07 (5.3131e+06)	5.1332e+10 (8.1646e+09)	1.6627e+08 (1.3735e+07)	7.0532e+03 (2.1123e+03)
C14	9.9039e+13 (9.8802e+12)	9.1359e+13 (7.7087e+12)	1.2658e+14 (1.1474e+13)	8.8915e+10 (7.2741e+10)	4.1754e+13 (6.1619e+12)	3.1018e+09 (2.2457e+08)	1.6872e+05 (1.4420e+05)
C15	1.6636e+14 (1.4941e+13)	1.6587e+14 (1.4900e+13)	1.6780e+14 (1.4952e+13)	1.6534e+14 (1.4784e+13)	1.5982e+14 (1.4021e+13)	1.6565e+14 (1.4788e+13)	4.0739e+07 (1.1635e+07)
C17MOD	4.5259e+00 (7.9747e-01)	1.2081e+02 (8.1880e+00)	5.3102e+00 (3.8269e-01)	4.8892e+00 (5.6902e-01)	3.6618e+01 (2.4068e+00)	5.7031e+00 (7.7777e-01)	9.7229e+00 (4.2845e+00)
G1	-4.8460e+00 (2.5393e-01)	-2.6520e+00 (3.0268e-01)	-5.0664e+00 (3.2589e-01)	-8.1982e+00 (3.5685e-01)	-5.7125e+00 (1.4856e-01)	-1.2358e+01 (3.1299e-01)	-1.3386e+01 (3.2610e-01)
G19	6.3727e+02 (3.9925e+01)	1.2717e+04 (1.2120e+03)	8.0113e+02 (7.1961e+01)	7.7821e+01 (2.3259e+00)	7.9816e+02 (2.3129e+01)	6.4197e+01 (2.1127e+00)	1.2328e+04 (1.2407e+03)
G3MOD	-5.8040e-01 (6.7039e-03)	-8.5689e-06 (7.8293e-06)	-6.0100e-01 (6.1620e-03)	-6.9302e-01 (3.7562e-06)	-2.1476e-05 (1.1090e-05)	-6.4612e-01 (3.2067e-02)	-1.0379e-03 (5.1217e-04)

Table 2: Mean of the best feasible objective function values found for 30 trials, $100 * (d+1)$ simulations per trial. Numbers in parenthesis are the standard errors for the corresponding sample means. The best value found for the test problem out of the seven algorithms is colored in blue and outlined by a solid box. The 2nd best is colored in magenta and outlined by a box of dashes.

Though CARS-RBF is a substantial improvement to CARS, it also outperforms or competes with the other optimization methods, including established alternative optimization methods. Table 2 shows that CARS-RBF performs better than CPRS-RBF on 30 of the 31 test problems, SDPENm on 22 out of 31 test problems, and ConstrLMSRBF for 18 of the 31 test problems. SDPENm does not use a surrogate model, and ConstrLMSRBF uses the same type of RBF model that is used by CPRS-RBF and CARS-RBF. It follows that Table 2 also validates the claim that CARS-RBF is competitive with established optimization methods.

8.2. Data Profiles

We also compare the CARS and CARS-RBF with alternative methods using data profiles [69]. For the purpose of creating these profiles, we define a problem p to be a pairing of a test problem (from Table 1) and a feasible starting point. Since we have 31 test problems and 30 feasible starting points, we consider $31 \times 30 = 930$ problems in total. Given a set $\mathcal{P}$ of problems, we run a set $\mathcal{S}$ of solvers. In the comparisons, there are 7 solvers: CARS, CARS-RBF, ConstrLMSRBF, CPRS, CPRS-RBF, SDPENm, and APSO.

Given a solver $s \in \mathcal{S}$, the *data profile* of a solver s on problem set $\mathcal{P}$ is the function

$$d_s(\alpha) = \frac{1}{|\mathcal{P}|} \left| \left\{ p \in \mathcal{P} \ : \ \frac{t_{p,s}}{n_p + 1} \leq \alpha \right\} \right|, \qquad \text{for } \alpha > 0,$$

where $t_{p,s}$ is the number of simulations required by solver s to satisfy the convergence test on problem p described below, and n_p is the number of variables in problem p. For a given solver s and any $\alpha > 0$, $d_s(\alpha)$ is the fraction of problems "solved" (i.e., problems where the solver generated a point satisfying the convergence test) by s within $\alpha \cdot (n_p + 1)$ simulations (equivalent to α simplex gradient estimates) [69].

In constrained expensive black-box optimization, algorithms are compared given a fixed and relatively limited number of simulations. As in Regis and Wild (2016), our convergence test uses tolerances $\epsilon, \tau > 0$, and it checks whether a point x obtained by a solver satisfies

$$\max_{i=1,\ldots,q} g_i(x) \leq \epsilon, \qquad \text{and} \qquad f(x^{(0)}) - f(x) \geq (1 - \tau) \left(f(x^{(0)}) - f_L^{\epsilon,\mathcal{S},\mu_f} \right). \tag{7}$$

Here, $f_L^{\epsilon,\mathcal{S},\mu_f}$ is the minimum objective function value of ϵ-feasible points obtained by *any* of the solvers within a given budget μ_f of simulations. That is, if x^{ϵ,s,μ_f} is the best ϵ-feasible point obtained by solver $s \in \mathcal{S}$ within μ_f simulations, then $f_L^{\epsilon,\mathcal{S},\mu_f} = \min_{s \in \mathcal{S}} f(x^{\epsilon,s,\mu_f})$.

Since the starting point $x^{(0)}$ is feasible in all our tests, it follows that at least one solver $s \in \mathcal{S}$ will satisfy the convergence test (7) for any given $\epsilon, \tau, \mu_f > 0$. In cases where there are multiple points from solver s satisfying (7) on problem p, the performance measure $t_{p,s}$ is taken to be the minimum number of evaluations needed to satisfy (7).

8.3. Comparisons with Alternative Methods Using Data Profiles

Figure 1 shows the data profiles of the 7 solvers on all 31 test problems in this study. The tolerances for this convergence test are $\epsilon = 10^{-8}$ and $\tau = 0.01$. The maximum number of

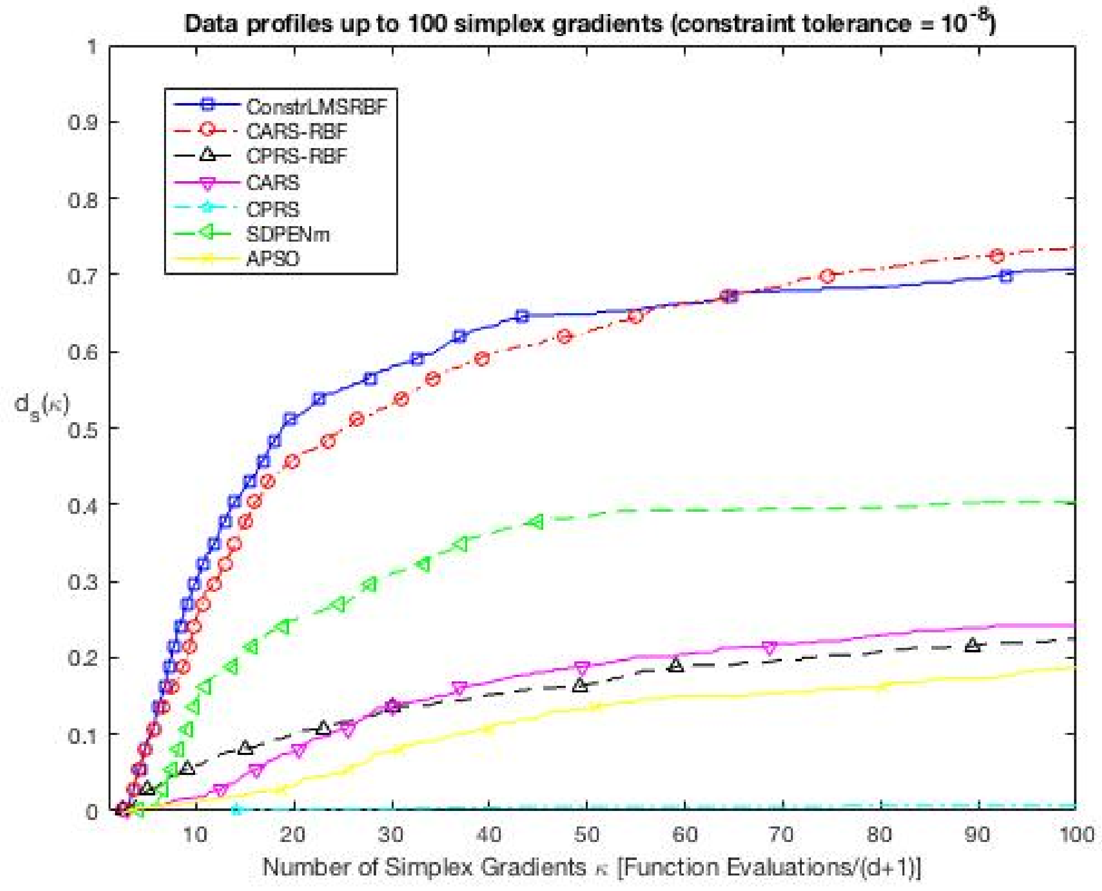

Figure 1: Data profiles of the 7 algorithms on the 31 test problems with 30 different starting feasible points, $\epsilon = 10^{-8}$, and $\tau = 0.01$. The 31 test problems have 2-20 decision variables and 1-38 inequality constraints.

simulations for these data profiles is equivalent to 100 simplex gradient estimates (1 simplex gradient estimate is equivalent to $d + 1$ simulations). One simulation yields the values of $f(x)$ and $G(x)$ for any given x.

Figure 1 reveals the effectiveness and competitiveness of the CARS-RBF algorithm. The performance improvement between the data profiles of CARS-RBF and CARS reveals that the addition of RBF surrogate models yields substantial performance improvements for the CARS algorithm. There is a change of over 50% between the performances of CARS and CARS-RBF. In addition to CARS-RBF being a substantial improvement over CARS, this extended algorithm also competes against established alternative optimization methods. In Figure 1, CARS-RBF performs better than ConstrLMSRBF after about 60 simplex gradient estimates (i.e., $60(d+1)$ simulations) and better than SDPENm the entire time. Observe that CARS-RBF satisfies the convergence test in almost 75% of the problems within 100 simplex gradient estimates (i.e., $100(d + 1)$ simulations) while SDPENm satisfies the convergence test in about 40% and ConstrLMSRBF in about 70% of the problems within the same computational budget. The positive slope of CARS-RBF at 100 simplex gradients suggests that CARS-RBF will not plateau and continue finding better feasible solutions if given a larger computational budget. In contrast, it seems ConstrLMSRBF is either at a plateau or is finding better solutions at a slower rate. Either way, the data profiles confirm the results from the data table and reaffirm that CARS-RBF is a competitive contender for constrained black-box global optimization.

Although the addition of surrogate models is a substantial improvement, there is also promise to CARS without surrogates. As mentioned earlier, ARS was proven both theoretically and experimentally to perform better than PRS on bound constrained problems, so it is expected the constrained versions will perform similarly. In relation to all 7 algorithms, CARS satisfies the convergence test in almost 25% of the problems within 100 simplex gradient estimates. In contrast, CPRS satisfies the convergence test in about 1% of the problems within the same computational budget. It was also discovered that CARS outperformed other state-of-the-art optimization methods such as the Accelerated Particle Swarm Optimization (APSO) algorithm for constrained problems developed by Yang ([1]). APSO satisfies the convergence test in almost 20% of the problems within 100 simplex gradient estimates. However, APSO does not possess any theoretical convergence guarantee to the global optimal solution. Having the property of convergence, CARS is a promising method for constrained black-box global optimization.

Next, we focus on the 11 test problems from the CEC 2010 benchmark (Mallipeddi and Suganthan [61]). The data profiles in Figure 2 again confirm the claims that CARS-RBF is competitive against established alternative optimization algorithms. In this subset of test problems, CARS-RBF performs as well as ConstrLMSRBF and better than SDPENm within a computational budget equivalent to 10 simplex gradient estimates (i.e., $10(d + 1)$ simulations). CARS-RBF performs worse than SDPENm between 10 and 40 simplex gradient simulations and worse than ConstrLMSRBF between 10 and 60 simplex gradient simulations. However, CARS-RBF continues to find better feasible solutions and performs better than ConstrLMSRBF and SDPENm after 60 and 40 simplex gradient simulations, respectively.

CARS, on the other hand, outperforms APSO the entirety of the computational budget on the CEC 2010 benchmark problems until the very end when they perform equally as well. Meanwhile, CARS performs better than CPRS on these problems the entire time, as expected. It also performs better than the surrogate-based CPRS-RBF algorithm after 20 simplex gradient estimates. Note that CPRS and CPRS-RBF satisfy the convergence test in only about 1% and 25% of the problems, respectively, within 100 simplex gradient estimates, while CARS and APSO satisfies the convergence test in over 40% of the problems within the same computational budget. These results confirm that CARS generally outperforms CPRS and its RBF-assisted extension and competes with other optimization methods on the CEC 2010 benchmark problems.

Interestingly, both SDPENm and APSO performed better in the CEC 2010 problems than in the complete set of test problems. Comparing CARS-RBF with SDPENm and CARS with APSO, the gaps in the data profiles are smaller in this subset of test problems than in the complete set. These results reveal that the algorithmic framework of CARS

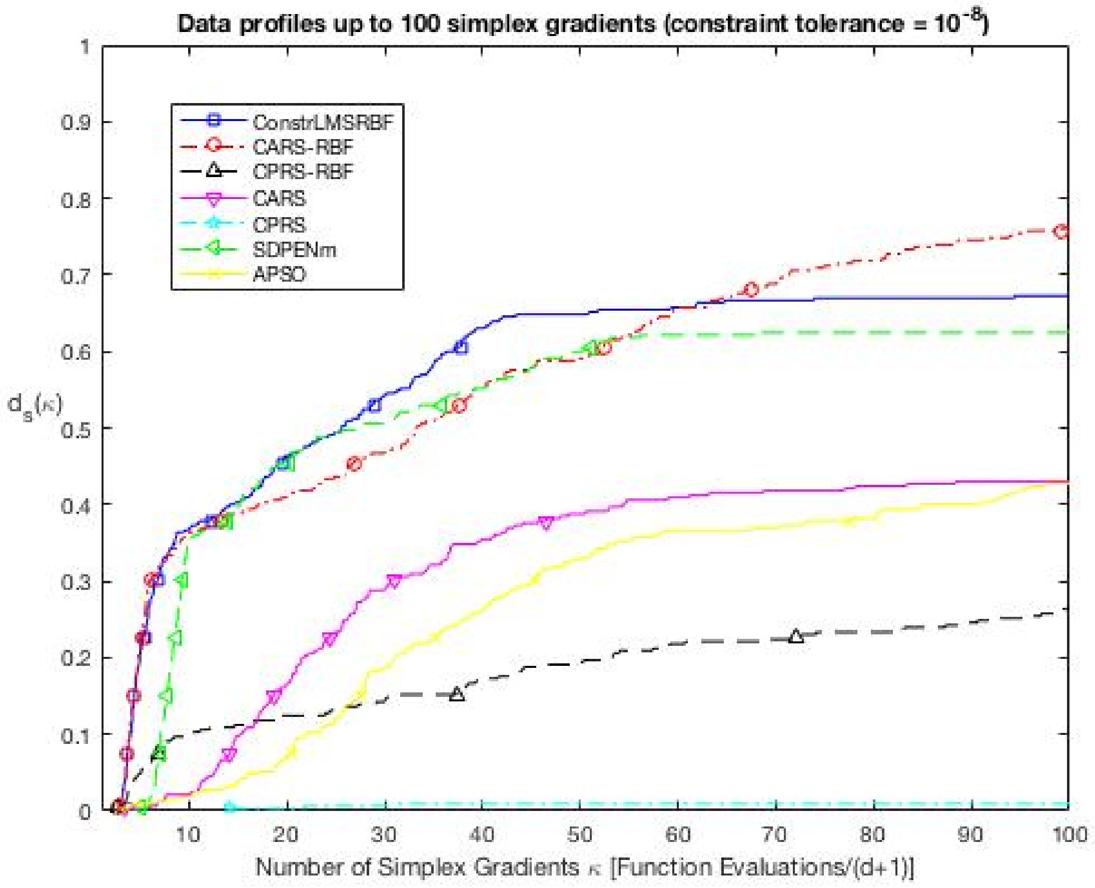

Figure 2: Data profiles of the 7 algorithms on 11 test problems from the CEC 2010 benchmark with 30 different starting feasible points, $\epsilon = 10^{-8}$, and $\tau = 0.01$. The 11 test problems have 10 decision variables and 1–3 inequality constraints.

still performs well despite focusing on the more challenging CEC 2010 benchmark problems. Also, the smaller gaps suggest that adding surrogate models to other optimization methods, such as SDPENm and APSO, might result in new methods that compete with CARS-RBF.

8.4. Comparison of CARS-RBF Performance for Different Types of RBF Models

To understand the sensitivity of the CARS-RBF algorithm to the type of RBF model used, six different RBF models were compared. These models differ in the form of ϕ used in (2). The six forms of ϕ are:

- cubic: $\phi(r) = r^3$,
- Gaussian: $\phi(r) = exp(-(r/\gamma)^2)$,
- multiquadric: $\phi(r) = \sqrt{r^2 + \gamma^2}$,

- quintic: $\phi(r) = r^5$,
- thin plate spline: $\phi(r) = r^2 log(r)$, and
- polyharmonic quartic: $\phi(r) = r^4 log(r)$.

For the Gaussian and multiquadric RBF models, the parameter γ was found using leave-one-out cross-validation after the objective function values were obtained from the initial sample points.

Table 3 shows the mean of the best feasible value found by each version of CARS-RBF on each test problem after 30 trials with a computational budget of $100(d + 1)$ simulations per trial, where d is the dimension of the problem. As before, the number in the parenthesis next to the mean best value is the standard error of the sample mean and all results are rounded to four decimal places.

From the results of Table 3, below are the number of test problems where each RBF model is either the best or the second best in terms of the mean of the best feasible values among the six types of RBF models:

- cubic: 13 out of 31 test problems
- quintic: 12 out of 31 test problems
- polyharmonic quartic: 11 out of 31 test problems
- multiquadric: 10 out of 31 test problems
- thin plate spline: 9 out of 31 test problems
- Gaussian: 7 out of 31 test problems

Note that the cubic RBF model has the most number of test problems where it is either the best or the second best among different types of RBF models. Moreover, the performance of the cubic RBF model is better than or comparable to the performance of the Gaussian, multiquadric, quintic, thin plate spline and polyharmonic quartic RBF models on 25, 18, 16, 20 and 17 of the test problems, respectively. This confirms that the cubic RBF model is a good choice among the different types of RBF models.

It is also worth noting that the Gaussian and the multiquadric RBF models did not always yield the best performance except on the two-dimensional problems (G6, G8 and G24). This suggests that having a parameter for the RBF model does not necessarily result in better performance, especially for higher dimensional problems. This might be because determining a good value for the parameter is difficult in high dimensions when the number of available sample points is relatively small.

Figure 3 shows the data profiles of the six versions of CARS-RBF on all 31 test problems in this study. As before, the tolerances for the convergence test are $\epsilon = 10^{-8}$ and $\tau = 0.01$. Moreover, the maximum number of simulations shown in these data profiles is equivalent to 100 simplex gradient estimates.

Test Problem	Cubic	Gaussian	Multiquadric	Quintic	Thin Plate Spline	Polyharmonic Quartic
G6	-6.8478e+03 (1.0763e+01)	-6.8961e+03 (6.7292e+00)	-6.8901e+03 (7.2540e+00)	-6.8864e+03 (7.3661e+00)	-6.7720e+03 (1.9618e+01)	-6.8561e+03 (1.1217e+01)
G8	-6.2234e-02 (6.2338e-03)	-9.5788e-02 (4.8965e-06)	-9.5801e-02 (2.8593e-06)	-9.1347e-02 (3.0876e-03)	-6.8582e-02 (6.1885e-03)	-7.3509e-02 (5.8548e-03)
G24	-5.5061e+00 (2.6636e-04)	-5.5068e+00 (1.6721e-04)	-5.5067e+00 (1.2729e-04)	-5.5063e+00 (1.9446e-04)	-5.5056e+00 (2.4421e-04)	-5.5065e+00 (1.5345e-04)
G5MOD	5.1307e+03 (6.8687e-01)	5.1312e+03 (7.3560e-01)	5.1316e+03 (7.3736e-01)	5.1314e+03 (7.8907e-01)	5.1295e+03 (2.0932e-01)	5.1301e+03 (3.5448e-01)
WB4	2.4539e+00 (2.9738e-02)	2.3867e+00 (2.7163e-02)	2.3833e+00 (2.3078e-02)	2.3211e+00 (1.1089e-02)	2.7402e+00 (6.1620e-02)	2.4680e+00 (3.4344e-02)
PVD4	5.8699e+03 (7.6198e+00)	7.1931e+03 (1.0436e+02)	7.1709e+03 (1.1780e+02)	6.6247e+03 (9.7750e+01)	5.8616e+03 (9.9553e+00)	6.5009e+03 (9.2920e+01)
GTCD4	3.0026e+06 (7.4556e+03)	3.1233e+06 (2.3948e+04)	3.0977e+06 (2.3308e+04)	3.0304e+06 (1.3315e+04)	3.0510e+06 (1.3395e+04)	3.0254e+06 (9.8858e+03)
G4	-3.0664e+04 (1.0583e-01)	-3.0664e+04 (9.6787e-02)	-3.0664e+04 (9.7273e-02)	-3.0664e+04 (6.8759e-02)	-3.0664e+04 (1.0024e-01)	-3.0664e+04 (1.2138e-01)
G13MOD	3.5761e-03 (1.3628e-05)	6.8130e-03 (2.0012e-03)	3.6561e-03 (8.3935e-05)	3.5482e-03 (1.2782e-05)	3.6862e-03 (6.6550e-05)	3.5612e-03 (1.3344e-05)
G16	-1.9032e+00 (1.4206e-04)	-1.9029e+00 (1.7942e-04)	-1.9028e+00 (2.1105e-04)	-1.9030e+00 (1.6660e-04)	-1.9030e+00 (1.9912e-04)	-1.9029e+00 (2.0306e-04)
Hesse	-2.9030e+02 (1.4442e+00)	-2.9397e+02 (1.6857e+00)	-2.9199e+02 (1.4655e+00)	-2.9382e+02 (1.6959e+00)	-2.9229e+02 (1.2836e+00)	-2.9082e+02 (1.7242e+00)
G9	7.7888e+02 (1.8201e+01)	7.9049e+02 (5.8693e+01)	7.5698e+02 (2.8928e+01)	7.1026e+02 (9.6833e+00)	8.7038e+02 (2.9708e+01)	7.2160e+02 (7.9088e+00)
SR7	3.0056e+03 (7.1960e-01)	3.0056e+03 (7.5791e-01)	3.0055e+03 (6.6929e-01)	3.0048e+03 (5.6092e-01)	3.0053e+03 (7.5629e-01)	3.0042e+03 (6.3760e-01)
G10	7.6393e+03 (9.1632e+01)	9.5957e+03 (4.1426e+02)	8.8774e+03 (2.7959e+02)	8.7109e+03 (2.8774e+02)	7.5681e+03 (6.6557e+01)	8.4605e+03 (2.3778e+02)
G18	-8.1354e-01 (1.4906e-02)	-8.1340e-01 (1.4904e-02)	-8.2549e-01 (1.3143e-02)	-8.2603e-01 (1.3164e-02)	-8.0719e-01 (1.5438e-02)	-8.1596e-01 (1.3618e-02)
G2	-5.4267e-01 (1.5301e-02)	-4.7729e-01 (2.5259e-02)	-5.6077e-01 (1.8794e-02)	-3.4988e-01 (1.4737e-02)	-5.9255e-01 (1.4251e-02)	-4.9838e-01 (1.9640e-02)
G7	2.4647e+01 (1.9148e-02)	2.5305e+01 (4.3526e-01)	2.4640e+01 (2.2822e-02)	2.4658e+01 (2.5159e-02)	2.4691e+01 (2.9527e-02)	2.4869e+01 (4.9013e-02)
C01	-4.5715e-01 (1.3609e-02)	-3.0225e-01 (1.3841e-02)	-2.7882e-01 (7.5012e-03)	-3.8328e-01 (1.5212e-02)	-3.0758e-01 (1.4360e-02)	-4.4406e-01 (1.4352e-02)
C02MOD	-2.2114e+00 (3.1975e-02)	-1.4803e+00 (2.2804e-01)	-2.1131e+00 (4.6339e-02)	-2.1345e+00 (4.1732e-02)	-2.2433e+00 (1.1885e-02)	-2.2030e+00 (3.3206e-02)
C05MOD	-4.7165e+02 (4.0030e+00)	-3.5709e+02 (2.3276e+01)	-4.4214e+02 (9.8065e+00)	-3.8068e+02 (1.7398e+01)	-4.6260e+02 (6.2386e+00)	-4.4369e+02 (8.9606e+00)
C06MOD	-3.6983e+02 (1.7466e+01)	-3.0881e+02 (3.7729e+01)	-3.1590e+02 (1.9449e+01)	-3.2090e+02 (2.1112e+01)	-3.7112e+02 (1.7414e+01)	-3.4707e+02 (2.1013e+01)
C07	3.6011e+05 (7.4414e+04)	3.6071e+08 (3.6051e+08)	6.4858e+04 (5.0669e+04)	7.4973e+04 (3.6936e+04)	1.4009e+06 (2.1615e+05)	8.5168e+04 (2.4333e+04)
C08	5.7753e+05 (1.5980e+05)	1.8912e+08 (1.4327e+08)	7.4825e+05 (4.0364e+05)	8.6770e+04 (2.4521e+04)	7.7793e+06 (1.6282e+06)	1.5706e+05 (7.9693e+04)
C09MOD	3.6364e+07 (7.0368e+06)	4.4338e+08 (4.3672e+08)	4.8804e+07 (2.7200e+07)	2.1495e+06 (5.8552e+05)	2.2579e+08 (3.1512e+07)	4.1310e+06 (8.7558e+05)
C10MOD	3.5231e+07 (5.3131e+06)	5.9766e+09 (5.9715e+09)	5.7175e+06 (2.5554e+06)	4.1869e+06 (1.0249e+06)	1.6129e+08 (2.7813e+07)	4.4159e+06 (8.8065e+05)
C14	8.8915e+10 (7.2741e+10)	6.1363e+12 (9.6427e+11)	5.0637e+12 (1.1808e+12)	1.3534e+12 (1.3314e+12)	5.3005e+10 (2.1085e+10)	1.3228e+13 (4.9253e+12)
C15	1.6534e+14 (1.4784e+13)	1.5772e+14 (1.4630e+13)	1.5831e+14 (1.5195e+13)	1.6664e+14 (1.4947e+13)	1.6198e+14 (1.5023e+13)	1.6639e+14 (1.4941e+13)
C17MOD	4.8892e+00 (5.6902e-01)	6.7175e+00 (5.7244e+00)	2.7308e+00 (1.1998e+00)	1.8225e+00 (3.1648e-01)	9.7802e+00 (1.7986e+00)	1.9722e+00 (3.8438e-01)
G1	-8.1982e+00 (3.5685e-01)	-8.1237e+00 (3.6195e-01)	-8.0055e+00 (3.7309e-01)	-8.2439e+00 (3.8116e-01)	-7.8291e+00 (3.2728e-01)	-8.1993e+00 (3.5297e-01)
G19	7.7821e+01 (2.3259e+00)	8.0114e+01 (3.1134e+00)	7.8781e+01 (2.3855e+00)	8.5405e+01 (3.4538e+00)	8.2525e+01 (1.8023e+00)	8.5213e+01 (2.5075e+00)
G3MOD	-6.9302e-01 (3.7562e-06)	-6.9302e-01 (3.4128e-06)	-6.9302e-01 (3.4710e-06)	-6.9302e-01 (3.2091e-06)	-6.9301e-01 (3.5724e-06)	-6.9302e-01 (4.2164e-06)

Table 3: Mean of the best feasible objective function values found for 30 trials, $100 * (d+1)$ simulations per trial. Numbers in parenthesis are the standard errors for the corresponding sample means. The best value found for the test problem out of all algorithms is colored in blue and outlined by a solid box. The 2^{nd} best is colored in magenta and outlined by a box of dashes.

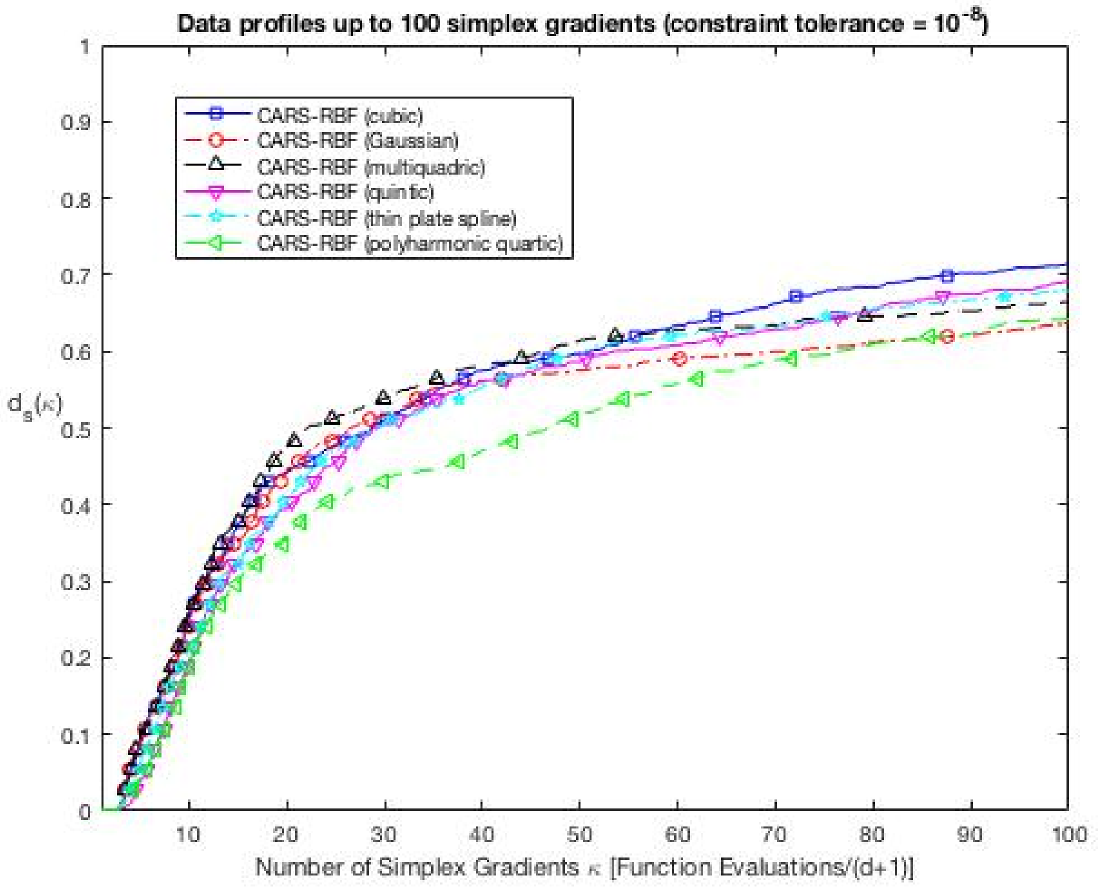

Figure 3: Data profiles of CARS-RBF implemented with 6 different RBF models on the 31 test problems with 30 different starting feasible points, $\epsilon = 10^{-8}$, and $\tau = 0.01$. The 31 test problems have 2-20 decision variables and 1-38 inequality constraints.

From Figure 3, each version of CARS-RBF performs similarly well, though the polyharmonic quartic RBF model takes longer to perform comparably to the other models. CARS-RBF with the cubic RBF model performs the best within the total computational budget. It satisfies the convergence test in about 72% of the problems while the other versions of CARS-RBF such as the one with the Gaussian or the polyharmonic quartic model satisfies the convergence test in less than 65% of the problems. However, others do show better performance at certain numbers of simulations. For example, CARS-RBF with the multiquadric RBF model has the best performance roughly between 10 and 35 simplex gradient estimates.

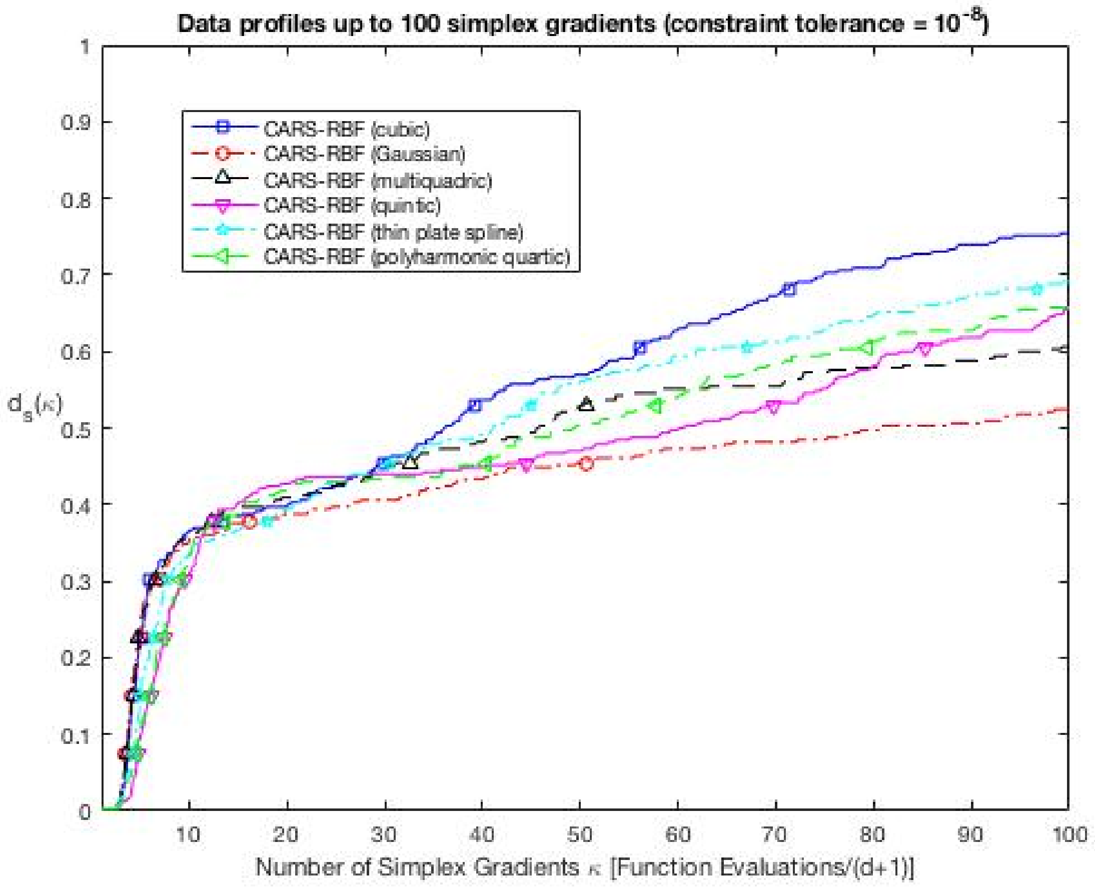

Figure 4: Data profiles of CARS-RBF implemented with 6 different RBF models on 11 test problems from the CEC 2010 benchmark with 30 different starting feasible points, $\epsilon = 10^{-8}$, and $\tau = 0.01$. The 11 test problems have 10 decision variables and 1–3 inequality constraints.

Next, we again focus on the 11 test problems from the CEC 2010 benchmark (Mallipeddi and Suganthan [61]). The data profiles in Figure 4 again confirm that the cubic RBF model outperformed the other versions. In this subset, however, the cubic RBF model consistently and significantly performed the best among the different versions. Interestingly, the Gaussian and multiquadric RBF models performed worse on the CEC 2010 benchmark than in the entire set of test problems.

9. Summary and Conclusions

This paper developed Constrained Accelerated Random Search (CARS), which is an extension of the Accelerated Random Search (ARS) algorithm to optimization problems with black-box inequality constraints. In CARS, the new sample point is chosen uniformly at random from a search neighborhood around the current best solution (in terms of the objective function and a measure of constraint violation). The radius of the search neighborhood is reduced whenever the new sample point is not an improvement over the current best solution, and it is reset to the initial value whenever the new sample point is better than the current best solution. Moreover, the radius is also reset if it falls below a pre-determined threshold value. This paper also presented the class of Scattered Uniform Random Search (SCAT-URS) algorithms, which includes CARS as a special case. Under certain reasonable assumptions, we proved the convergence to the global minimum in a probabilistic sense of any SCAT-URS algorithm, including CARS. In the numerical experiments, we compared CARS with an extension of Pure Random Search to constrained optimization (CPRS) and with the Accelerated Particle Swarm Optimization (APSO) (Yang [1]) on 31 widely used test problems whose dimensions range from 2 to 20 and with up to 38 inequality constraints. Included among the test problems are some CEC 2010 benchmark problems.

The results showed that CARS obtained better feasible objective function values on average than CPRS on 29 of the test problems and APSO on 19 of the test problems when the computational budget is set at $100(d+1)$ simulations, where one simulation yields all the objective and constraint function values at a single input. Moreover, data profiles up to a computational budget equivalent to 100 simplex gradient estimates (i.e., $100(d+1)$ simulations) confirm that CARS performed better than CPRS and APSO. Thus, this paper numerically confirmed that the effectiveness of the Accelerated Random Search strategy extends to constrained black-box optimization and that the resulting algorithm competes with a state-of-the-art algorithm such as APSO.

To improve the performance of CARS on computationally expensive problems, we also developed a surrogate-assisted version of CARS. The surrogate model used is a Radial Basis Function (RBF) interpolation model, and the resulting method is called CARS-RBF. Results show that CARS-RBF is much better than CARS on 29 out of 31 test problems, confirming the idea that RBF surrogates can enhance the performance of CARS. Moreover, CARS-RBF outperformed an RBF-assisted extension of CPRS called CPRS-RBF on 30 of the 31 test problems. It also outperformed SDPENm [13] on 22 out of 31 test problems, and ConstrLMSRBF [14] for 18 of the 31 test problems.

Data profiles of the algorithms up to a computational budget of 100 simplex gradient estimates confirm that CARS-RBF is a substantial improvement over CARS. The profiles also show that CARS-RBF outperformed SDPENm and CPRS-RBF and that it is competitive with ConstrLMSRBF. Moreover, on the more challenging CEC 2010 benchmark problems, CARS-RBF outperformed ConstrLMSRBF, which is an established surrogate-based method for constrained black-box optimization.

This study also examined the sensitivity of the CARS-RBF algorithm to the type of RBF model used. CARS-RBF was run on the same test problems using six types of RBF models: cubic, Gaussian, multiquadric, quintic, thin plate spline, and polyharmonic quartic. The results show that the cubic RBF model is a good choice among the various alternatives in terms of the mean of the best feasible objective value found within $100(d+1)$ simulations. The data profiles also show that the cubic RBF model performed generally the best and the other versions yielded comparable results, though the polyharmonic quartic RBF model took longer to achieve the same level of performance as the others. From the data profiles of the six versions of CARS-RBF on the more challenging CEC 2010 benchmark problems, the cubic RBF model significantly outperformed the other five RBF models while the Gaussian RBF model performed significantly worse than the other models.

These results demonstrate the effectiveness of using RBF surrogates to improve the performance of CARS. It also shows that CARS-RBF is competitive with some established alternative methods for constrained black-box optimization. Overall, CARS-RBF is a promising method for constrained expensive black-box optimization.

Acknowledgements

We thank the Saint Joseph's University (SJU) 2015 Summer Scholars Program and NSF grant DUE-1154161 for providing funding to Luigi Nuñez and Kayla Varela during the early stages of this project.

References

[1] X.-S. Yang, Nature-inspired metaheuristic algorithms, 2nd edition, Luniver Press, 2010.

[2] Z. B. Zabinsky, Stochastic Adaptive Search in Global Optimization, Kluwer Academic Publishers, 2003.

[3] A. Zhigljavsky, A. Žilinskas, Stochastic Global Optimization, Springer US, 2008.

[4] F. J. Solis, R. J. B. Wets, Minimization by random search techniques, Mathematics of Operations Research 6 (1) (1981) 19–30.

[5] J. C. Spall, Introduction to Stochastic Search and Optimization, John Wiley & Sons, Inc., New Jersey, 2003.

[6] M. J. Appel, R. LaBarre, D. Radulović, On accelerated random search, SIAM Journal on Optimization 14 (3) (2004) 708–731.

[7] N. Baba, Convergence of a random optimization method for constrained optimization problems, Journal of Optimization Theory and Applications 33 (4) (1981) 451–461.

[8] J. D. Pinter, Convergence properties of stochastic optimization procedures, Optimization: A Journal of Mathematical Programming and Operations Research 15 (3) (1984) 405–427.

[9] R. G. Regis, On the convergence of adaptive stochastic search methods for constrained and multi-objective black-box optimization, Journal of Optimization Theory and Applications 170 (3) (2016) 932–959.

[10] A. Husain, K.-D. Lee, K.-Y. Kim, Enhanced multi-objective optimization of a dimpled channel through evolutionary algorithms and multiple surrogate methods, International Journal for Numerical Methods in Fluids 66 (6) (2011) 742–759.

[11] E. A. Kontoleontos, V. G. Asouti, K. C. Giannakoglou, An asynchronous metamodel-assisted memetic algorithm for CFD-based shape optimization, Engineering Optimization 44 (3) (2012) 157–173.

[12] V. K. Ky, C. D'Ambrosio, Y. Hamadi, L. Liberti, Surrogate-based methods for black-box optimization, International Transactions in Operational Research 24 (3) (2017) 393–424.

[13] G. Liuzzi, S. Lucidi, M. Sciandrone, Sequential penalty derivative-free methods for nonlinear constrained optimization, SIAM Journal on Optimization 20 (5) (2010) 2614–2635.

[14] R. G. Regis, Stochastic radial basis function algorithms for large-scale optimization involving expensive black-box objective and constraint functions, Computers & Operations Research 38 (5) (2011) 837–853.

[15] C. Audet, J. E. Dennis, Jr., Mesh adaptive direct search algorithms for constrained optimization, SIAM Journal on Optimization 17 (2) (2006) 188–217.

[16] C. Audet, J. E. Dennis, Jr., A progressive barrier for derivative-free nonlinear programming, SIAM Journal on Optimization 20 (1) (2009) 445–472.

[17] S. Le Digabel, Algorithm 909: NOMAD: Nonlinear optimization with the mads algorithm, ACM Transactions on Mathematical Software 37 (4) (2011) 44:1–44:15.

[18] A. R. Conn, K. Scheinberg, L. N. Vicente, Introduction to Derivative-Free Optimization, MPS/SIAM Series on Optimization, Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 2009.

[19] M. J. D. Powell, A direct search optimization methods that models the objective and constraint functions by linear interpolation, in: S. Gomez, J. P. Hennart (Eds.), Advances in Optimization and Numerical Analysis, Kluwer, Dordrecht, 1994, pp. 51–67.

[20] E. Mezura-Montes, C. A. Coello Coello, R. Landa-Becerra, Engineering optimization using simple evolutionary algorithm, in: Proceedings of the 15th IEEE International Conference on Tools with Artificial Intelligence, 2003, pp. 149–156.

[21] T. P. Runarsson, X. Yao, Stochastic ranking for constrained evolutionary optimization, IEEE Transactions on Evolutionary Computation 4 (3) (2000) 284–294.

[22] B. Tessema, G. G. Yen, A self adaptive penalty function based algorithm for constrained optimization, in: IEEE Congress on Evolutionary Computation, 2006 (CEC 2006), 2006, pp. 246–253.

[23] Y. Wang, Z. Cai, Combining multiobjective optimization with differential evolution to solve constrained optimization problems, IEEE Transactions on Evolutionary Computation 16 (1) (2012) 117–134.

[24] R. Datta, K. Deb, Individual penalty based constraint handling using a hybrid bi-objective and penalty function approach, in: 2013 IEEE Congress on Evolutionary Computation (CEC 2013), IEEE Press, Cancún, México, 2013, pp. 2720–2727.

[25] K. Deb, R. Datta, A bi-objective constrained optimization algorithm using a hybrid evolutionary and penalty function approach, Engineering Optimization 45 (5) (2013) 503–527.

[26] T. Takahama, S. Sakai, Efficient constrained optimization by the epsilon constrained rank-based differential evolution, in: Proceedings of 2012 IEEE Congress on Evolutionary Computation (CEC2012), Brisbane, Australia, 2012, pp. 62–69.

[27] C. A. Coello Coello, R. Landa-Becerra, Efficient evolutionary optimization through the use of a cultural algorithm, Engineering Optimization 36 (2) (2004) 219–236.

[28] E. Mezura-Montes, C. A. Coello Coello, A simple multimembered evolution strategy to solve constrained optimization problems, IEEE Transactions on Evolutionary Computation 9 (1) (2005) 1–17.

[29] A. I. J. Forrester, A. Sobester, A. J. Keane, Engineering Design via Surrogate Modelling: A practical guide, John Wiley & Sons, 2008.

[30] D. Jones, M. Schonlau, W. Welch, Efficient global optimization of expensive black-box functions, Journal of Global Optimization 13 (4) (1998) 455–492.

[31] H. M. Gutmann, A radial basis function method for global optimization, Journal of Global Optimization 19 (3) (2001) 201–227.

[32] S. Koziel, L. Leifsson, Efficient knowledge-based optimization of expensive computational models using adaptive response correction, Journal of Computational Science 11 (2015) 1–11.

[33] Y. Li, Y. Wu, J. Zhao, L. Chen, A kriging-based constrained global optimization algorithm for expensive black-box functions with infeasible initial points, Journal of Global Optimization 67 (1) (2017) 343–366.

[34] J. Müller, J. D. Woodbury, GOSAC: global optimization with surrogate approximation of constraints, Journal of Global Optimization 69 (1) (2017) 117–136.

[35] F. Boukouvala, M. G. Ierapetritou, Derivative-free optimization for expensive constrained problems using a novel expected improvement objective function, AIChE Journal 60 (7) (2014) 2462–2474.

[36] P. Feliot, J. Bect, E. Vazquez, A bayesian approach to constrained single- and multi-objective optimization, Journal of Global Optimization 67 (1) (2017) 97–133.

[37] R. B. Gramacy, G. A. Gray, S. Le Digabel, H. K. H. Lee, P. Ranjan, G. Wells, S. M. Wild, Modeling an augmented Lagrangian for blackbox constrained optimization, Technometrics 58 (1) (2016) 1–11. doi:10.1080/00401706.2015.1014065.

[38] T. Krityakierne, C. A. Shoemaker, SOMS: surrogate multistart algorithm for use with nonlinear programming for global optimization, International Transactions in Operational Research 24 (5) (2017) 1139–1172.

[39] R. G. Regis, S. M. Wild, CONORBIT: constrained optimization by radial basis function interpolation in trust regions, Optimization Methods and Software 32 (3) (2017) 552–580.

[40] M. A. Abramson, T. J. Asaki, J. E. Dennis, R. Magallanez, M. J. Sottile, An efficient class of direct search surrogate methods for solving expensive optimization problems with cpu-time-related functions, Structural and Multidisciplinary Optimization 45 (1) (2012) 53–64.

[41] M. C. Araujo, E. F. Wanner, F. G. Guimarães, R. H. C. Takahashi, Constrained optimization based on quadratic approximations in genetic algorithms, in: E. Mezura-Montes (Ed.), Constraint-Handling in Evolutionary Computation, Vol. 198 of Studies in Computational Intelligence, Springer, Berlin, 2009, Ch. 9, pp. 193–217.

[42] L. Shi, K. Rasheed, ASAGA: an adaptive surrogate-assisted genetic algorithm, in: Proceedings of the Genetic and Evolutionary Computation Conference (GECCO 2008), 2008, pp. 1049–1056.

[43] Y. Jin, M. Olhofer, B. Sendhoff, A framework for evolutionary optimization with approximate fitness functions, IEEE Transactions on Evolutionary Computation 6 (5) (2002) 481–494.

[44] M. Emmerich, A. Giotis, M. M. Özdemir, T. Bäck, K. Giannakoglou, Metamodel-assisted evolution strategies, in: Parallel Problem Solving from Nature VII, 2002, pp. 362–370.

[45] Z. Zhou, Y. S. Ong, P. B. Nair, A. J. Keane, K. Y. Lum, Combining global and local surrogate models to accelerate evolutionary optimization, IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews 37 (1) (2007) 66–76.

[46] A. Isaacs, T. Ray, W. Smith, Multiobjective design optimization using multiple adaptive spatially distributed surrogates, International Journal of Product Development 9 (1–3) (2009) 188–217.

[47] R. G. Regis, Evolutionary programming for high-dimensional constrained expensive black-box optimization using radial basis functions, IEEE Transactions on Evolutionary Computation 18 (3) (2014) 326–347.

[48] F. Gieseke, O. Kramer, Towards non-linear constraint estimation for expensive optimization, in: Esparcia-Alczar, A. Isabel (Eds.), EvoApplications, Vol. 7835 of Lecture Notes in Computer Science, Springer, 2013, pp. 459–468.

[49] I. Loshchilov, M. Schoenauer, M. Sebag, Self-adaptive surrogate-assisted covariance matrix adaptation evolution strategy, in: Proceedings of the Genetic and Evolutionary Computation Conference (GECCO 2012), 2012, pp. 321–328.

[50] T. P. Runarsson, Constrained evolutionary optimization by approximate ranking and surrogate models, in: Parallel Problem Solving from Nature VII (PPSN-2004), Vol. 3242 of Lecture Notes in Computer Science, Springer Verlag, 2004, pp. 401–410.

[51] A. A. Montaño, C. A. Coello Coello, E. Mezura-Montes, Multi-objective airfoil shape optimization using a multiple-surrogate approach, in: Proceedings of the IEEE Congress on Evolutionary Computation 2012, IEEE Press, 2012, pp. 1188–1195.

[52] B. Liu, S. Koziel, Q. Zhang, A multi-fidelity surrogate-model-assisted evolutionary algorithm for computationally expensive optimization problems, Journal of Computational Science 12 (2016) 28–37.

[53] Y. Jin, Surrogate-assisted evolutionary computation: recent advances and future challenges, Swarm and Evolutionary Computation 1 (2) (2011) 61–70.

[54] T. P. Runarsson, Approximate evolution strategy using stochastic ranking, in: IEEE Congress on Evolutionary Computation, 2006, pp. 745–752.

[55] R. G. Regis, Convergence guarantees for generalized adaptive stochastic search methods for continuous global optimization, European Journal of Operational Research 207 (3) (2010) 1187–1202.

[56] M. J. D. Powell, The theory of radial basis function approximation in 1990, in: W. Light (Ed.), Advances in Numerical Analysis, Volume 2: Wavelets, Subdivision Algorithms and Radial Basis Functions, Oxford University Press, Oxford, U.K., 1992, pp. 105–210.

[57] R. G. Regis, Constrained optimization by radial basis function interpolation for high-dimensional expensive black-box problems with infeasible initial points, Engineering Optimization 46 (2) (2014) 218–243.

[58] S. M. Wild, R. G. Regis, C. A. Shoemaker, ORBIT: optimization by radial basis function interpolation in trust-regions, SIAM Journal on Scientific Computing 30 (6) (2008) 3197–3219.

[59] S. I. Resnick, A Probability Path, Birkhäuser, Boston, 1999.

[60] Z. Michalewicz, M. Schoenauer, Evolutionary algorithms for constrained parameter optimization problems, Evolutionary Computation 4 (1) (1996) 1–32.

[61] R. Mallipeddi, P. N. Suganthan, Problem definitions and evaluation criteria for the cec 2010 competition on constrained real-parameter optimization, Tech. rep., Nanyang Technological University, Singapore (2010).

[62] A. Hedar, M. Fukushima, Derivative-free filter simulated annealing method for constrained continuous global optimization, Journal of Global Optimization 35 (4) (2006) 521–549.

[63] E. Mezura-Montes, O. Cetina-Dominguez, Empirical analysis of a modified artificial bee colony for constrained numerical optimization, Applied Mathematics and Computation 218 (22) (2012) 10943–10973.

[64] K. Deb, An efficient constraint handling method for genetic algorithms, Computer Methods in Applied Mechanics and Engineering 186 (2–4) (2000) 311–338.

[65] C. A. Coello Coello, E. Mezura-Montes, Constraint-handling in genetic algorithms through the use of dominance-based tournament selection, Advanced Engineering Informatics 16 (2002) 193–203.

[66] C. S. Beightler, D. T. Phillips, Applied Geometric Programming, Wiley, New York, 1976.

[67] C. A. Floudas, P. M. Pardalos, A Collection of Test Problems for Constrained Global Optimization Algorithms, Springer-Verlag, Berlin, 1990.

[68] R. Hesse, A heuristic search procedure for estimating a global solution of nonconvex programming problems, Operations Research 21 (1973) 1267–1280.

[69] J. J. Moré, S. M. Wild, Benchmarking derivative-free optimization algorithms, SIAM Journal on Optimization 20 (1) (2009) 172–191.