J Glob Optim (2007) 37:113–135
DOI 10.1007/s10898-006-9040-1

ORIGINAL ARTICLE

Improved strategies for radial basis function methods for global optimization

Rommel G. Regis · Christine A. Shoemaker

Received: 26 September 2005 / Accepted: 28 April 2006 /
Published online: 21 July 2006

Abstract We propose some strategies that can be shown to improve the performance of the radial basis function (RBF) method by Gutmann [J. Global optim. **19**(3), 201–227 (2001a)] (Gutmann-RBF) and the RBF method by Regis and Shoemaker [J. Global optim. **31**, 153–171 (2005)] (CORS–RBF) on some test problems when they are initialized by symmetric Latin hypercube designs (SLHDs). Both methods are designed for the global optimization of computationally expensive functions with multiple local optima. We demonstrate how the original implementation of Gutmann-RBF can sometimes converge slowly to the global minimum on some test problems because of its failure to do local search. We then propose *Controlled Gutmann-RBF* (*CG-RBF*), which is a modification of Gutmann-RBF where the function evaluation point in each iteration is restricted to a subregion of the domain centered around a global minimizer of the current RBF model. By varying the size of this subregion in different iterations, we ensure a better balance between local and global search. Moreover, we propose a complete restart strategy for CG-RBF and CORS-RBF whenever the algorithm fails to make any substantial progress after some threshold number of consecutive iterations. Computational experiments on the seven Dixon and Szegö [Towards Global optimization, pp. 1–13. North-Holland, Amsterdam (1978)] test problems and on nine Schoen [J. Global optim. **3**, 133–137 (1993)] test problems indicate that the proposed strategies yield significantly better performance on some problems. The results also indicate that, for some fixed setting of the restart parameters, the two modified RBF algorithms, namely *CG-RBF-Restart* and

R. G. Regis (✉)
Cornell Theory Center and School of Operations Research & Industrial Engineering,
Cornell University, Ithaca, NY 14853, USA
e-mail: rgr6@cornell.edu

C. A. Shoemaker
School of Civil & Environmental Engineering and School of
Operations Research & Industrial Engineering,
Cornell University, Ithaca, NY 14853, USA
e-mail: cas12@cornell.edu

CORS-RBF-Restart, are comparable on the test problems considered. Finally, we examine the sensitivity of CG-RBF-Restart and CORS-RBF-Restart to the restart parameters.

Keywords Global optimization · Expensive function · Function approximation · Response surface · Surrogate model · Radial basis function

1 Introduction

In this paper, we propose strategies for improving the performance of two radial basis function (RBF) algorithms for finding the global optimum of computationally expensive functions. Given a compact subset $\mathcal{D}$ of $\mathbb{R}^d$ and a deterministic continuous function $f: \mathcal{D} \to \mathbb{R}$, the *global optimization problem* (GOP) is to find $x^* \in \mathcal{D}$ such that $f(x^*) = \inf_{x \in \mathcal{D}} f(x)$. A solution x^* is called a *global minimum point of f over $\mathcal{D}$*. Note that the compactness of $\mathcal{D}$ and the continuity of f over $\mathcal{D}$ guarantees the existence of a global minimum point. An extensive treatment of GOP can be found in Horst et al. (2000) and Torn and Zilinskas (1989).

Our focus is on methods for solving GOP when f is computationally expensive to evaluate and its derivatives are not available. These GOPs are important because expensive black box functions can be found in many real-world scientific and engineering applications. We are particularly interested in global optimization methods that utilize *function approximation models* (also called *response surface models* or *surrogate models*) for the expensive function. Examples of these are the RBF method by Gutmann (2001a) (Gutmann-RBF), the Constrained Optimization using Response Surfaces (CORS) method by Regis and Shoemaker (2005), and the kriging-based Efficient Global Optimization (EGO) method by Jones et al. (1998).

In the literature, there are also local optimization methods that utilize function approximation models. These include the optimization framework developed by Serafini (1998) and by Booker et al. (1999) that combines pattern search algorithms and surrogate models. Other examples are the derivative-free trust region methods by Conn et al. (1997), Powell (2000, 2002) and Marazzi and Nocedal (2002). Each of these methods can be easily converted to a global optimization method by combining it with a global search strategy such as multistart.

This paper has four objectives. First, we estimate the average case performance of Gutmann-RBF (Gutmann 2001a) and CORS-RBF (Regis and Shoemaker 2005) on some test problems when they are initialized by symmetric Latin hypercube designs (SLHDs) (Ye et al. 2000). Previous work on these RBF methods (Gutmann 2001a; Regis and Shoemaker 2005) reported the performance of these algorithms when they are initialized by the corners of the hypercube domains of the test problems. The reason for using an SLHD instead of corner points is that it has excellent space-filling properties (Ye et al. 2000), and more importantly, its size can be specified by the user independent of the problem dimension. In contrast, the number of corner points grows exponentially with the problem dimension, making it infeasible for higher dimensional problems. The results indicate that for some test problems, there is a significant amount of variability in the number of function evaluations required to get within 1% of the optimal value when using randomly generated SLHDs. For either RBF method, there are many combinations of test problem and SLHD for which the algorithm did not even get close to the global optimum value after a large number of function evaluations.

This brings us to the second main objective, which is to understand why slow convergence occurs and to propose strategies for dealing with these problems in RBF methods. We demonstrate how the original implementation of Gutmann-RBF can sometimes fail to achieve a balance between local and global search. We then propose *Controlled Gutmann-RBF* (*CG-RBF*), which is a modification of Gutmann-RBF where the global optimization subproblem that generates the iterates for Gutmann-RBF is restricted to a subregion of the domain centered around a global minimizer of the RBF model. By varying the size of this subregion in different iterations, we ensure a better balance between local and global search. In addition, we propose a complete restart strategy (i.e. start completely from scratch using a new SLHD) for either CG-RBF or CORS-RBF whenever the algorithm is not making any substantial progress after some threshold number of consecutive function evaluations. We shall demonstrate that the proposed strategies will result in better performance for these RBF methods on some test problems.

Third, we perform a statistical comparison of the two modified RBF algorithms, namely *CG-RBF-Restart* and *CORS-RBF-Restart*, on the seven Dixon and Szegö (1978) test problems and on nine Schoen (1993) test problems. The results indicate that, for some fixed setting of the restart parameters, these two modified RBF algorithms are comparable on the test problems considered. Finally, we examine the sensitivity of CG-RBF-Restart and CORS-RBF-Restart to the restart parameters.

2 Radial basis function interpolation model

The following RBF interpolation model was used as the basis of the RBF methods by Gutmann (2001a) and Regis and Shoemaker (2005). This RBF model was extensively studied by Powell (1992, 1999) and Buhmann (2003).

Given n distinct points $x_1, \ldots, x_n \in \mathbb{R}^d$ where the function values $f(x_1), \ldots, f(x_n)$ are known, we use an interpolant of the form

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

where $\|\cdot\|$ is the Euclidean norm, $\lambda_i \in \mathbb{R}$ for $i = 1, \ldots, n$, $p \in \Pi_m^d$ (the linear space of polynomials in d variables of degree less than or equal to m), and ϕ is a real-valued function that can take many forms. In this investigation, we used $\phi(r) = r^2 \log r$, $r > 0$ and $\phi(0) = 0$ (thin plate spline). However, other forms of ϕ include: $\phi(r) = r^3$ (cubic), $\phi(r) = \sqrt{r^2 + \gamma^2}$ (multiquadric), and $\phi(r) = e^{-\gamma r^2}$ (Gaussian), where $r \geq 0$ and γ is a positive constant.

Select a particular ϕ. Define the matrix $\Phi \in \mathbb{R}^{n \times n}$ by: $\Phi_{ij} := \phi(\|x_i - x_j\|)$, $i, j = 1, \ldots, n$. Moreover, define m_ϕ to be -1 if ϕ is Gaussian, 0 if ϕ is a multiquadric, and 1 if ϕ is cubic or the thin plate spline. Let $m \geq m_\phi$ and let $\widehat{m}$ be the dimension of the linear space Π_m^d. (Note that $\widehat{m} = \binom{m+d}{d}$.) Also, let $p_1, \ldots, p_{\widehat{m}}$ be a basis of Π_m^d, and define the matrix $P \in \mathbb{R}^{n \times \widehat{m}}$ as follows: $P_{ij} := p_j(x_i)$, $i = 1, \ldots, n; j = 1, \ldots, \widehat{m}$. Here, the RBF that interpolates the points $(x_1, f(x_1)), \ldots, (x_n, f(x_n))$ is obtained by solving the system

$$\begin{pmatrix} \Phi & P \\ P^T & 0 \end{pmatrix} \begin{pmatrix} \lambda \\ c \end{pmatrix} = \begin{pmatrix} F \\ 0_{\widehat{m}} \end{pmatrix}, \tag{2}$$

where $F = (f(x_1), \ldots, f(x_n))^T$, $\lambda = (\lambda_1, \ldots, \lambda_n)^T \in \mathbb{R}^n$ and $c = (c_1, \ldots, c_{\widehat{m}})^T \in \mathbb{R}^{\widehat{m}}$.

It can be shown that the coefficient matrix in (2) is invertible if and only if the matrix P defined above has full column rank (Powell 1992). Hence, a necessary condition for the invertibility of the coefficient matrix is that $n \geq \widehat{m}$. Moreover, if the coefficient matrix in (2) is invertible, then it will remain invertible with the addition of new data points that are distinct from the previous ones.

3 Radial basis function methods

This paper focuses on improving the performance of the RBF method by Gutmann (2001a) and the RBF method by Regis and Shoemaker (2005). This section describes these two methods in enough detail to explain how and why the suggested improvements affect the performance of each method.

3.1 RBF method by Gutmann

The RBF method by Gutmann (2001a) was based on the following general iterative technique by Jones (1996). Let $x_1, \ldots, x_n$ be the previously evaluated (or sampled) points in $\mathcal{D}$ and assume that, we have an estimate f_n^* of the global minimum value of the expensive function f in the current iteration. We refer to f_n^* as a *target value*. Jones (1996) proposed the idea of sampling where it is "most reasonable" to imagine that the function has a global minimum point assuming that its global minimum value is, in fact, f_n^*. More precisely, for each $y \notin \{x_1, \ldots, x_n\}$, assume that there is a unique function s_n^y, belonging to some linear space of functions, that interpolates the data points $(x_1, f(x_1)), \ldots, (x_n, f(x_n))$ and the additional data point (y, f_n^*). Jones (1996) proposed that the next evaluation point x_{n+1} be chosen to be the point $y \in \mathcal{D}$ such that s_n^y is "most reasonable". Figure 1 shows four possible locations of the global minimum point (denoted by a diamond symbol). In each case, we constructed a thin plate spline RBF model that interpolates all previously evaluated data points and the assumed global minimum point. The idea is that the most likely location of the global minimum point is the one that yields the "most reasonable" or "least complicated" RBF interpolant, which in this case would be the bottom right subfigure.

Gutmann (2001a) interpreted "reasonable" to mean "less bumpy" and discovered that there is a natural measure of *bumpiness* of the RBF interpolant in (1) given by the semi-norm

$$\sigma(s_n) = \langle s_n, s_n \rangle := (-1)^{m_\phi + 1} \sum_{i=1}^{n} \lambda_i s_n(x_i).$$

This resulted in the development of an RBF method for global optimization, which we shall refer to as Gutmann-RBF.

Next, we discuss the global minimization of the bumpiness function $B_n(y) := \sigma(s_n^y)$ in the case of RBFs. We recall the notation in Sect. 2. Let s_n^* denote the global minimum value of the current RBF model $s_n(x)$ over $\mathcal{D}$, i.e. $s_n^* := \inf_{x \in \mathcal{D}} s_n(x)$. Moreover, for each $y \in \mathcal{D}$, define $u_n(y) := (\phi(\|y - x_1\|), \ldots, \phi(\|y - x_n\|))^T$ and $\pi(y) := (p_1(y), \ldots, p_{\widehat{m}}(y))^T$. Gutmann (2001b) showed that, if the target value $f_n^* < s_n^*$, then the global minimization of $B_n(y)$ over all $y \in \mathcal{D}$ is equivalent to the global maximization of

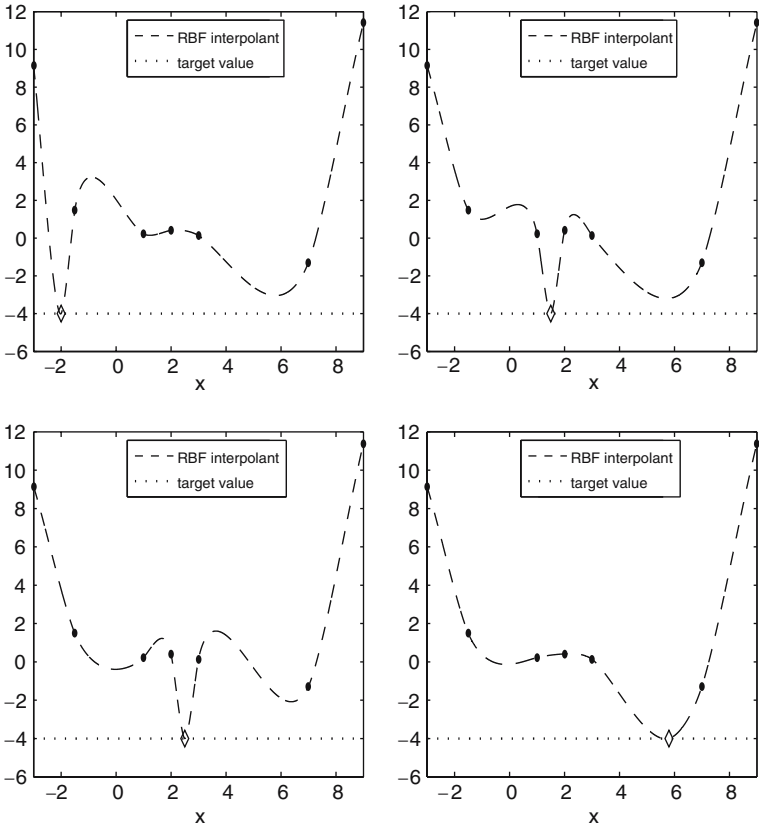

Fig. 1 The four subfigures represent possible scenarios for the location of the global minimum point, which is represented by the diamond symbol. The *solid dots* are previously evaluated points and the *dotted line* represents the guess of the global minimum value (i.e. target value f_n^*) for the current iteration. The *dashed curve* is the thin plate spline RBF model that interpolates the previously evaluated points and the assumed global minimum point. Jones (1996) proposed the idea of sampling where it is "most reasonable" to imagine that the function has a global minimum, i.e. select the candidate global minimum point that corresponds to the "least complicated" RBF model

$$h_n(y) := \frac{(-1)^{m_\phi+1}}{[s_n(y)-f_n^*]^2} \times \left[\phi(0) - (u_n(y)^T \quad \pi(y)^T)\begin{pmatrix}\Phi & P\\ P^T & 0\end{pmatrix}^{-1}\begin{pmatrix}u_n(y)\\ \pi(y)\end{pmatrix}\right], \quad y \in \mathcal{D}, \qquad (3)$$

where m_ϕ and $s_n(y)$ are defined as in Sect. 2. However, if $f_n^* = s_n^*$, then the global minimum of $B_n(y)$ over $\mathcal{D}$ occurs at any global minimizer of $s_n(x)$ over $\mathcal{D}$ (Gutmann 2001a). Hence, we cannot set $f_n^* = s_n^*$ if our estimated global minimizer of $s_n(x)$ over $\mathcal{D}$ coincides with or is too close to a previously evaluated point.

The target values f_n^* are selected in the interval $[-\infty, s_n^*]$ and are set by performing cycles of $N+1$ iterations, where each cycle starts with a low target value and ends with a high target value equal or close to s_n^*. We refer to N as the *cycle length*. Gutmann (2001a) noted that the low target values correspond to global search while the high

target values correspond to local search. In the linear, cubic and thin plate spline cases, Gutmann (2001a) showed that, by choosing target values in a particular manner, convergence to the global minimum is guaranteed for any continuous function defined over a compact set.

In this investigation, we adopt the procedure used by Gutmann (2001a) and by Björkman and Holmström (2000) in setting the target values. First, we set the cycle length to $N = 5$. Let α be a permutation of $\{1, \ldots, n\}$ such that $f(x_{\alpha(1)}) \leq \cdots \leq f(x_{\alpha(n)})$. Also, let n_0 be the number of initial evaluation points (i.e. number of points in the space-filling experimental design). For $n \geq n_0$, set

$$f_n^* = s_n^* - W_n \cdot \left(f(x_{\alpha(k_n)}) - s_n^*\right), \tag{4}$$

where

$$W_n = \left[\frac{\mathrm{mod}(N - (n - n_0),\ N + 1)}{N}\right]^2$$

$$\text{and} \quad k_n = \begin{cases} n, & \text{if } \mathrm{mod}(n - n_0, N + 1) = 0, \\ k_{n-1} - \lfloor (n - n_0)/N \rfloor, & \text{otherwise.} \end{cases}$$

When $W_n = 0$, we have $f_n^* = s_n^*$. However, as noted earlier, this choice is only valid if there is a global minimizer of $s_n(x)$ over $\mathcal{D}$ that is not too close to a previously evaluated point. Otherwise, we need to reset f_n^* to a value slightly below s_n^*.

When $N = 5$, note that W_n cycles through the values $1, (4/5)^2, (3/5)^2, (2/5)^2, (1/5)^2, 0$. The purpose of decreasing the values of W_n within a cycle is to produce target values that progressively get closer to s_n^*. Moreover, the above fractions are squared to put more emphasis on local search. Multiplying W_n by $(f(x_{\alpha(k_n)}) - s_n^*)$ makes the desired improvement proportional to the scale of the function. Furthermore, when n is at the beginning of a cycle (i.e. when $\mathrm{mod}(n - n_0, N + 1) = 0$), we have $f(x_{\alpha(k_n)}) = \max_{1 \leq i \leq k_n} f(x_{\alpha(i)}) = \max_{1 \leq i \leq n} f(x_i)$ in Eq. 4. However, as n increases in the cycle, more and more points with high function values are excluded in the calculation of $f(x_{\alpha(k_n)}) = \max_{1 \leq i \leq k_n} f(x_{\alpha(i)})$. The purpose of excluding high function values is to control the effect of unusually large function values in the calculation of f_n^*. This procedure for setting the target values is somewhat heuristic but it seems to work well in practice.

Gutmann (2001b) noted that $s_n, h_n \in C^1(\mathbb{R}^d)$ in the thin plate spline case, $s_n, h_n \in C^2(\mathbb{R}^d)$ in the cubic case, and $s_n, h_n \in C^\infty(\mathbb{R}^d)$ in the multiquadric and Gaussian cases. These properties allow us to use multistart gradient-based local optimization methods to find the global minimum of s_n over $\mathcal{D}$ (needed in setting the target value f_n^*) and the global maximum of h_n over $\mathcal{D}$. However, Björkman and Holmström (2000) found that maximizing $h_n(y)$ could lead to numerical difficulties. Hence, we adopted their procedure of minimizing $-\log(h_n(y))$ instead when selecting the next evaluation point x_{n+1}. Moreover, we used the matrix factorizations described in Powell (1996) and Björkman and Holmström (2000) to obtain an efficient implementation of Gutmann-RBF.

3.2 RBF method by Regis and Shoemaker

As before, let $x_1, \ldots, x_n \in \mathcal{D}$ be the previously evaluated points. Define $\Delta_n := \max_{x \in \mathcal{D}} \min_{1 \leq j \leq n} \|x - x_j\|$ to be the *maximin distance* in $\mathcal{D}$ relative to $x_1, \ldots, x_n$. Clearly, the distance between any $x \in \mathcal{D}$ and any previously evaluated point is at most Δ_n. In the CORS-RBF method (Regis and Shoemaker 2005), the next evaluation point

x_{n+1} is chosen to be a point that solves the following global optimization subproblem:

$$\text{Minimize } \{ s_n(x) : x \in \mathcal{D},\ \|x - x_j\| \geq \beta_n \Delta_n,\ j = 1, \ldots, n\}, \tag{5}$$

where Δ_n is given above and $0 \leq \beta_n \leq 1$ is a parameter to be set by the user. That is, the next evaluation point is chosen to be the point $y \in \mathcal{D}$ that minimizes the RBF model subject to the constraints that y be of distance at least $\beta_n \Delta_n$ from each previously evaluated point. We refer to β_n as a *distance factor* and we shall refer to the constrained optimization problem (5) as the CORS–RBF *auxiliary optimization subproblem*. If $\limsup_{n\to\infty} \beta_n > 0$, then CORS-RBF converges to the global minimum of an arbitrary continuous function defined over the compact set $\mathcal{D}$ (Regis and Shoemaker 2005).

To balance global and local search in CORS-RBF, we perform cycles of $N + 1$ iterations, where each cycle employs a range of values for the distance factor β_n, starting with a high value close to 1 (global search) and ending with a value of 0 (local search). More precisely, if n_0 is the number of initial evaluation points, we have $\beta_n = \beta_{n+N+1}$ for all $n \geq n_0$ and $1 \geq \beta_{n_0} \geq \beta_{n_0+1} \geq \cdots \geq \beta_{n_0+N} = 0$. We refer to N as the *cycle length* and we refer to the sequence $\langle \beta_{n_0}, \beta_{n_0+1}, \ldots, \beta_{n_0+N} = 0 \rangle$ as the *search pattern*. When $\beta_n = 0$ and our estimated global minimizer of $s_n(x)$ over $\mathcal{D}$ is too close to a previously evaluated point, we simply reset β_n to some small positive value.

4 Improved local search for the RBF method by Gutmann

4.1 Local search in Gutmann-RBF

In the context, of response surface methods for global optimization, it would be convenient to define *local* and *global search* loosely as follows: a response surface method is said to perform *local search* in the current iteration if the selected evaluation point is close to a global minimizer of the current response surface model. On the other hand, the algorithm is said to perform *global search* if the selected evaluation point is far from the previously evaluated points.

Recall from Sect. 3.1 that when $W_n = 0$ (or equivalently, when $f_n^* = s_n^* := \inf_{x\in\mathcal{D}} s_n(x)$), then the next iterate x_{n+1} will be a global minimizer of $s_n(x)$ that is not a previously evaluated point. Clearly, the algorithm performs local search when $W_n = 0$ and there is a global minimizer of $s_n(x)$ that is not in $\{x_1, \ldots, x_n\}$. Moreover, Gutmann (2001a) noted that when $W_n > 0$ is small (or equivalently, when f_n^* is close to s_n^* but strictly less than s_n^*), then the algorithm is also expected to perform local search. A more precise statement concerning the connection between local search and high target values close to s_n^* is given by the following theorem.

Theorem 1 *Suppose $\mathcal{D}$ is a compact set in $\mathbb{R}^d$. Let $\Omega(s_n, \mathcal{D})$ be the set of global minimizers of s_n in $\mathcal{D}$ and let $\Gamma(h_n, \mathcal{D})$ be the set of global maximizers of h_n in $\mathcal{D}$. Let $\widetilde{\Omega}_n := \Omega(s_n, \mathcal{D}) \setminus \{x_1, \ldots, x_n\}$ and assume that $\widetilde{\Omega}_n \neq \emptyset$. Then $\forall \epsilon > 0$, $\exists \delta_n > 0$ with the following property*:

$$0 < s_n^* - f_n^* < \delta_n \Longrightarrow \Gamma(h_n, \mathcal{D}) \subseteq \left(\bigcup_{x^* \in \widetilde{\Omega}_n} B(x^*, \epsilon) \right) \bigcap \mathcal{D}. \tag{6}$$

Proof Fix $\epsilon > 0$. If $\bigcup_{x^* \in \widetilde{\Omega}_n} B(x^*, \epsilon) \supseteq \mathcal{D}$, then any choice of $\delta_n > 0$ will work. Suppose $\mathcal{D} \setminus (\bigcup_{x^* \in \widetilde{\Omega}_n} B(x^*, \epsilon)) \neq \emptyset$. We recall the notation in Sect. 3.1. Moreover, let

$$V_n(y) := (-1)^{m_\phi + 1} \left[\phi(0) - (u_n(y)^T \quad \pi(y)^T) \begin{pmatrix} \Phi & P \\ P^T & 0 \end{pmatrix}^{-1} \begin{pmatrix} u_n(y) \\ \pi(y) \end{pmatrix} \right].$$

If $f_n^* < s_n^*$, then $h_n(y) = V_n(y)/[s_n(y) - f_n^*]^2$ for all $y \in \mathcal{D}$. Gutmann (2001b) showed that $V_n(y)$ is continuous on $\mathcal{D}$, $V_n(x_i) = 0$ for all $i = 1, \ldots, n$, and $V_n(y) > 0$ for all $y \in \mathcal{D} \setminus \{x_1, \ldots, x_n\}$. Since $\mathcal{D}$ is compact, $\exists K_n > 0$ such that $0 \leq V_n(y) \leq K_n \, \forall y \in \mathcal{D}$.

Next, consider the function $\xi_n(y) := s_n(y) - s_n^*$, $y \in \mathcal{D}$. By definition, $\xi_n(y) \geq 0$, $\forall y \in \mathcal{D}$. Moreover, $\xi_n(y)$ is continuous and strictly positive over the compact set $\mathcal{D} \setminus (\bigcup_{x^* \in \widetilde{\Omega}_n} B(x^*, \epsilon))$. Hence, $\exists H_n > 0$ such that $\xi_n(y) \geq H_n > 0$, $\forall y \in \mathcal{D} \setminus (\bigcup_{x^* \in \widetilde{\Omega}_n} B(x^*, \epsilon))$. Fix $x^* \in \widetilde{\Omega}_n$ and let $\delta_n = H_n \sqrt{V_n(x^*)}/(\sqrt{\eta K_n} - \sqrt{V_n(x^*)})$, where $\eta > 1$ is some constant. Since $x^* \notin \{x_1, \ldots, x_n\}$, it follows that $V_n(x^*) > 0$. Moreover, $V_n(x^*) < \eta K_n$. Hence, $\delta_n > 0$. We wish to show that Property (6) holds for this choice of δ_n.

We argue by contradiction. Suppose that for some $f_n^* \in (s_n^* - \delta_n, s_n^*)$, $\exists y^* \in \Gamma(h_n, \mathcal{D}) \setminus (\bigcup_{x^* \in \widetilde{\Omega}_n} B(x^*, \epsilon))$. Then

$$\frac{V_n(y^*)}{[s_n(y^*) - f_n^*]^2} = h_n(y^*) \geq h_n(x^*) = \frac{V_n(x^*)}{[s_n(x^*) - f_n^*]^2}. \tag{7}$$

Let $s_n^* - f_n^* = \theta_n$. Note that $s_n(y^*) - f_n^* = s_n(y^*) - (s_n^* - \theta_n) = \theta_n + \xi_n(y^*)$. Hence, (7) becomes

$$\frac{V_n(y^*)}{[\theta_n + \xi_n(y^*)]^2} \geq \frac{V_n(x^*)}{\theta_n^2}. \tag{8}$$

Note that $y^* \notin \bigcup_{x^* \in \widetilde{\Omega}_n} B(x^*, \epsilon)$, and so, $\xi_n(y^*) \geq H_n$. Moreover, $0 < \theta_n < \delta_n$. Hence, (8) implies that

$$\begin{aligned} V_n(y^*) &\geq V_n(x^*) \left[1 + \frac{\xi_n(y^*)}{\theta_n} \right]^2 \geq V_n(x^*) \left[1 + \frac{H_n}{\delta_n} \right]^2 \\ &= V_n(x^*) \left[1 + \frac{\sqrt{\eta K_n} - \sqrt{V_n(x^*)}}{\sqrt{V_n(x^*)}} \right]^2 = \eta K_n, \end{aligned}$$

which is a contradiction since $V_n(y^*) \leq K_n$. Thus, Property (6) holds for the chosen δ_n. □

Theorem 1 assumes that there is a global minimizer of $s_n(x)$ in $\mathcal{D}$ that is not a previously evaluated point. Given an $\epsilon > 0$, Theorem 1 guarantees the existence of a $\delta_n > 0$ such that the next iterate x_{n+1} is within ϵ-radius of one such minimizer of $s_n(x)$ whenever f_n^* is in the open interval $(s_n^* - \delta_n, s_n^*)$. However, a particular value of δ_n that guarantees this condition is hard to calculate in practice. As will be seen in an example below, a small value of W_n (which gives a value of f_n^* that is relatively close to s_n^*) could result in an iterate x_{n+1} that is as far away as possible from the global minimizer of $s_n(x)$.

4.2 A convergence problem in Gutmann-RBF

In our computational experiments, we observed a convergence problem in Gutmann-RBF on the Shekel problems from the Dixon and Szegö (1978) testbed. In particular, when Gutmann-RBF was applied to these problems, there were many instances where the best solution found is already in the basin of the global minimizer of the test problem, and yet, the algorithm takes a long time to converge to the global minimum point. Our investigation suggests that the slow convergence was due to the failure of local search. We now provide some examples that demonstrate how local search could fail in Gutmann-RBF. For simplicity in the discussion below, assume that $s_n(x)$ has a unique *global* minimizer x_n^* in $\mathcal{D}$ in every iteration. Note that $s_n(x)$ could still have multiple local minima on $\mathcal{D}$.

In Fig. 2, we applied Gutmann-RBF to the function

$$f(x) = \frac{-1}{(x-0.15)^2 + 0.001}, \quad x \in [0,1].$$

Here, $n = 6$ and the dots represent the previously evaluated data points $(0.05, -90.91)$, $(0.09, -217.39)$, $(0.11, -384.62)$, $(0.19, -384.62)$, $(0.22, -169.49)$, $(0.5, -8.10)$. Fig. 2a shows the function (solid curve) and the RBF model $s_n(x)$ that interpolates the previously evaluated points (dashed curve). Observe that x_n^*, the global minimizer of $s_n(x)$ on $[0,1]$, is not one of the previously evaluated points.

We consider one iteration of Gutmann-RBF with $W_n = 0.04$, which is one of the values taken by W_n when $N = 5$ (see Sect. 3.1). This corresponds to a high target value of $f_n^* = -498.49$ (represented by the dotted line in Fig. 2a), which is relatively close to $s_n^* = -479.63$. By doing an exhaustive line search in the interval $[0,1]$, we will see that the point $y \in [0,1]$ that yields the minimum bumpiness for the RBF model

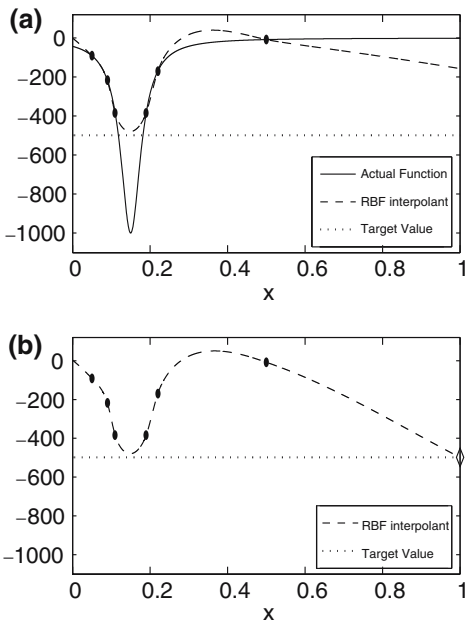

Fig. 2 In (**a**), the *solid curve* is the actual function. The *dots* are previously evaluated points and the dashed curve is the thin plate spline RBF model that interpolates the previously evaluated points. The *dotted line* represents the target value f_n^* for the current iteration of Gutmann-RBF and it is close to $s_n^* = s_n(x_n^*)$. In (**b**), the point $y \in [0,1]$ that yields the minimum bumpiness for the RBF model that interpolates the previously evaluated data points and the additional data point (y, f_n^*) is $y = 1$, which is far from x_n^*

that interpolates the previously evaluated data points and the additional data point (y, f_n^*) is $y = 1$. Figure 2b shows the RBF model associated with the point selected for function evaluation. Note that the candidate point for function evaluation (i.e. $y = 1$) is far from the previously evaluated points in $[0, 1]$ and is also far from x_n^*, which is the global minimizer of $s_n(x)$ (shown in Fig. 2a) on $[0, 1]$.

Another situation that can cause the failure of local search in Gutmann-RBF is when x_n^* (i.e. the global minimizer of $s_n(x)$ over $\mathcal{D}$) coincides with one of the previously evaluated points $x_1, \ldots, x_n$. This could happen if a function has steep valleys. That is, if the function value of the current iterate is much lower than those around it (i.e. if we stumble on a steep valley), then it will often happen that the global minimum of the updated RBF surface occurs exactly at this especially low point. If x_n^* coincides with a previously evaluated point, then as we have noted in Sect. 3.1, we cannot set $f_n^* = s_n^*$. Moreover, when $f_n^* < s_n^*$, Gutmann (2001b) showed that $h_n(x_i) = 0$ for $i = 1, \ldots, n$ and $h_n(y) > 0$ for all $y \in \mathcal{D} \backslash \{x_1, \ldots, x_n\}$. Hence, the new iterate x_{n+1} will be away from x_n^* (since this is one of the x_i's) even if we set f_n^* close to s_n^*. The distance of x_{n+1} from x_n^* will depend on the function f. However, an example below shows that x_{n+1} could be as far away as possible from x_n^* even if W_n is fairly small (i.e. f_n^* is fairly close to s_n^*).

In Fig. 3, we have a similar setup as in Fig. 2 except that the global minimizer $x_n^* = 0.135$ of $s_n(x)$ is now one of the previously evaluated points. Here, the data points are $(0.05, -90.91)$, $(0.135, -816.33)$, $(0.18, -526.32)$, $(0.22, -169.49)$, $(0.5, -8.10)$.

As before, we consider one iteration of Gutmann-RBF with $W_n = 0.04$. This corresponds to a high target value of $f_n^* = -848.66$ (represented by the dotted line in Fig. 3a), which is relatively close to $s_n^* = -816.33$. By doing an exhaustive line search in the interval $[0, 1]$, we will again see that the global minimum value of the bumpiness function $B_n(y)$ (see Sect. 3.1) occurs at $y = 1$ (see Fig. 3b), which is far from the previously evaluated points (one of which is x_n^*).

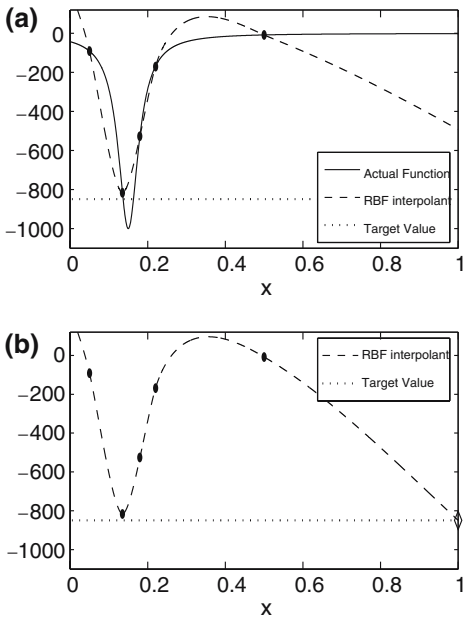

Fig. 3 The setup is similar to Fig. 2 except that the global minimizer x_n^* of the RBF model is now one of the previously evaluated points. Note that the point $y \in [0, 1]$ that yields the minimum bumpiness for the RBF model that interpolates the previously evaluated data points and the additional data point (y, f_n^*) is again $y = 1$, which is far from x_n^*

Finally, if x_n^* is too close to a previously evaluated point but not exactly identical to it, then local search would most likely be ineffective. In this case, Theorem 1 still holds. However, for every $\epsilon > 0$, the value of δ_n that guarantees that x_{n+1} is within ϵ-radius of x_n^* would be very tiny because $V_n(x_n^*)$ (from the proof of Theorem 1) would be a very small positive number. Now among the values of f_n^* in a cycle, N are strictly less than s_n^* and one is exactly equal to s_n^*. Since δ_n is so tiny, none of the N target values $f_n^* < s_n^*$ that we compute are likely to be within distance δ_n from s_n^*. Hence, local search will only occur when $f_n^* = s_n^*$. In this case, our iterate will be x_n^*, which by assumption is very close to a previously evaluated point, and so, our local search will be highly ineffective.

To summarize, we have shown that there is no guarantee that Gutmann-RBF performs local search when $W_n > 0$ is small. This is true whether or not the global minimizer of the RBF model is a previously evaluated point. This results in slow convergence for some functions similar to the one in Figs. 2 and 3. This also provides one reason for the slow convergence of Gutmann-RBF on the Shekel problems, where the global minimum is somewhat "steep" just like the function in the example above. In the next section, we propose a strategy for dealing with this problem.

4.3 Restricted global minimization of the bumpiness function

Effective global optimization requires a balance between local and global search. Large values of W_n (see Eq. 4 in Sect. 3.1) are intended for global search while small values of W_n are intended for local search. However, we have noted in Sect. 4.2 that the small values of W_n in Gutmann-RBF do not guarantee local search. To deal with this problem, we propose a modification to Gutmann-RBF, called *Controlled Gutmann-RBF* (*CG-RBF*), where the global minimization of the bumpiness function $B_n(y)$ is restricted to a small hyperrectangle centered at a global minimizer of $s_n(x)$ in $\mathcal{D}$ whenever W_n is small.

More precisely, we will find the point $y \in \mathcal{D} = [a, b] \subseteq \mathbb{R}^d$ that minimizes $B_n(y)$ subject to the constraint that $y \in [x_n^* - \rho_n(b-a),\ x_n^* + \rho_n(b-a)] \cap \mathcal{D}$, where x_n^* is a global minimizer of $s_n(x)$ and ρ_n is a function of the parameter W_n with the following properties: $0 \le \rho_n \le 1$, $\rho_n = 0$ if $W_n = 0$, and $\rho_n(W_n) \le \rho_n(W_n')$ whenever $W_n \le W_n'$. Moreover, ρ_n should be small whenever W_n is small and $\rho_n = 1$ whenever W_n is large. Note that, when $\rho_n = 0$, then the iterate will be x_n^*. However, when $\rho_n = 1$, we have $[x_n^* - \rho_n(b-a),\ x_n^* + \rho_n(b-a)] \cap \mathcal{D} = \mathcal{D}$ so there is essentially no restriction on the new iterate.

In this investigation, we will use the following for ρ_n:

$$\rho_n(W_n) = \begin{cases} \nu\sqrt{W_n}, & \text{if } 0 \le W_n \le U, \\ 1, & \text{otherwise,} \end{cases}$$

where $0 < \nu < 1$ and $U > 0$ are parameters to be specified. In the numerical experiments, we will set $\nu = 1/2$ and $U = 1/4$.

Note that the above strategy for CG-RBF promotes a better balance between local and global search since it ensures that local search is always performed whenever W_n is small. In the computational experiments below, we will demonstrate that this strategy is helpful in alleviating some of the convergence problems of Gutmann-RBF when it is applied to functions whose characteristics are similar to those of the Shekel functions and the function shown in Figs. 2 and 3.

A similar strategy could be implemented in CORS-RBF. However, CORS-RBF does not exhibit the same convergence problem as Gutmann-RBF so there is hardly any benefit in doing this.

5 Restart strategy for RBF methods

In our computational experience, there is a significant amount of variability in the performance (number of function evaluations to get within a specified level of accuracy) of RBF methods on some test problems when these RBF methods are initialized by SLHDs. Depending on the SLHD used, some runs are relatively short while other runs take a long time (even after the strategy in Sect. 4.3 was implemented on Gutmann-RBF). In particular, for all the Shekel test problems, many trials of CG-RBF and CORS-RBF take a long time but a substantial number of trials are relatively short. These long trials might be due to the algorithm being unlucky with global search and spending considerable local search effort on the wrong local minima. That is, the global searches of these RBF algorithms are unable to locate the basin of the global minimum and so the local searches do not really help.

To deal with this problem, we propose a *complete restart strategy* for CG-RBF and CORS-RBF whenever the algorithm is not making any substantial progress. Here, a *complete restart* means that we start the algorithm from scratch using a new SLHD. However, when we count the number of function evaluations to get a specified level of accuracy we also count the total number of function evaluations in the previous runs that we stopped. If a substantial fraction of trials are relatively short and we happen get an SLHD that will yield a long trial, then restarts should eventually result in an SLHD that will yield a short run. Restart strategies have proven to be effective in backtrack procedures for the solution of some difficult satisfiability and constraint satisfaction problems (Gomes et al. 2000). Hence, it is reasonable to consider such strategies when applying RBF methods on difficult global optimization problems. We shall refer to CG-RBF with restart and CORS-RBF with restart as *CG-RBF-Restart* and *CORS-RBF-Restart*, respectively.

To determine the moment when the algorithm will be restarted, we will count the number of consecutive iterations that did not result in a substantial improvement in function value. By *substantial improvement*, we mean that the improvement in function value should be some percentage $I_{\min}$ of the difference between the lower quartile (25th percentile) and the minimum of all available function values. Here, $I_{\min}$ is a parameter that, we will set to 0.5%. If the number of non-improving iterations exceeds some threshold value, then we will restart the algorithm from scratch using a new SLHD. We will set this threshold value equal to $C_{\max}(N+1)$, where $C_{\max}$ is a parameter and N is the cycle length. Hence, the threshold number of non-improving iterations before performing a restart is equivalent to $C_{\max}$ cycles of the standard implementation of CG-RBF and CORS-RBF. In this investigation, we will set $C_{\max} = 5$. Since $N = 5$ in the standard implementation of CG-RBF and CORS-RBF, this threshold is equal to 30 iterations.

6 Numerical experiments

6.1 Measuring the performance of RBF methods

In order to assess the significance of our proposed improvements, we need an accurate measure of performance for the RBF methods. The performance of an RBF method depends heavily on the number of initial evaluation points and also on how these points are located in the domain $\mathcal{D}$. Some choices of initial evaluation points will lead to the global minimum more quickly than others. Hence, an accurate characterization of performance of an RBF method should be based on some average case performance over all possible initial evaluation points (i.e. all possible finite nonempty subsets of the domain such that the resulting P matrix from Sect. 2 has full column rank). Since it is impossible to compute an average performance over all possible finite subsets of $\mathcal{D}$, we simply estimate an average performance over some reasonable collection of initial evaluation points.

In the papers by Gutmann (2001a), Björkman and Holmström (2000), and Regis and Shoemaker (2005), the RBF methods were all initialized by evaluating the function at the corners of the hypercube domains of the Dixon–Szegö test functions. For higher dimensional problems, using the corners is not feasible since the number of corners grows exponentially as the dimension increases. An alternative is to use space-filling experimental designs such as Latin hypercubes (McKay et al. 1979) and orthogonal arrays. Latin hypercubes are particularly convenient since the user can specify the number of initial evaluation points. Moreover, they also have the desirable property that the projection of the Latin hypercube design onto any single dimension is a uniform grid. In this investigation, we will use SLHDs (Ye et al. 2000) as the initial evaluation points since the symmetry condition improves the space-filling properties of a Latin hypercube.

Now for a given number of initial evaluation points, say ℓ, there are many symmetric Latin hypercube designs in $\mathcal{D}$ with ℓ points. Hence, we could measure the performance of an RBF method initialized by an SLHD of size ℓ on a particular function by considering an average performance of the RBF method on the given function over all possible SLHDs of size ℓ. We estimate this average performance by randomly generating several SLHDs of size ℓ in $\mathcal{D}$ and computing the average number of function evaluations it takes an RBF algorithm to get to a specified level of accuracy.

The results obtained by Gutmann (2001a), Björkman and Holmström (2000) and Regis and Shoemaker (2005) when using the corners of the hypercube domain as the initial evaluation points were excellent. However, in the computational experiments below, the performance of these RBF methods on the Dixon-Szegö functions when using SLHDs will exhibit significant variability and that the estimated average case performance when using SLHDs are generally worse compared to the performance when using only the hypercube corners on the Dixon and Szegö (1978) test functions. We used SLHDs because they are practical for both low dimensional and high-dimensional problems. Moreover, for low dimensional problems, one can always augment an SLHD with the corner points.

6.2 Test problems

Since the RBF methods by Gutmann (2001a) and Regis and Shoemaker (2005) were both tested on the Dixon and Szegö (1978) functions, we will also test our proposed

Table 1 The Dixon-Szegö test functions (Dixon and Szegö 1978)

Test function	Dim	Domain	No. of local min	Global min value
Branin	2	$[-5,10]\times[0,15]$	3	0.398
Goldstein–Price	2	$[-2,2]^2$	4	3
Hartman3	3	$[0,1]^3$	4	−3.86
Shekel5	4	$[0,10]^4$	5	−10.1532
Shekel7	4	$[0,10]^4$	7	−10.4029
Shekel10	4	$[0,10]^4$	10	−10.5364
Hartman6	6	$[0,1]^6$	4	−3.32

strategies on these problems. The properties of the Dixon and Szegö problems are summarized in Table 1. In the Branin test problem, all local minima are also global minima. For the other six problems, there is only one global minimum point.

In addition, we also created Schoen (1993) test problems of the form:

$$f(x)=\frac{\sum_{i=1}^{k} f_i \prod_{j\neq i}\|x-z_j\|^2}{\sum_{i=1}^{k}\prod_{j\neq i}\|x-z_j\|^2},\quad x\in[0,1]^d,$$

where $k\geq 1$, $z_j\in[0,1]^d\ \forall j=1,\ldots,k$, and $f_i\in\mathbb{R}\ \forall i=1,\ldots,k$. These functions have the following properties (Schoen 1993): (1) $f(z_i)=f_i\quad\forall i=1,\ldots,k$; (2) $\min_{1\leq i\leq k} f_i\leq f(x)\leq\max_{1\leq i\leq k} f_i\quad\forall x\in[0,1]^d$; and (3) $\lim_{x\to z_i}\nabla f(x)=0$. For all the Schoen functions that will be created below, the points $z_1,\ldots,z_k$ will always be generated uniformly at random throughout $\mathcal{D}=[0,1]^d$.

First, we generated Schoen test functions of the above form with $k=10$ for dimensions $d=3,4,5$ where the function values $f_1,\ldots,f_k$ were generated uniformly at random in $[0,100]$. We shall refer to these test functions as Schoen3, Schoen4 and Schoen5.

To demonstrate the importance of the proposed strategies, we constructed Schoen (1993) functions with characteristics that are similar to those of the Shekel problems and the function shown in Figs. 2 and 3.

More precisely, to highlight the importance of the CG-RBF strategy, we created Schoen functions with $k=100$ for dimensions $d=3,4,5$ where 1 of the function values $f_1,\ldots,f_k$ was generated uniformly at random in $[-1000,-900]$ while the other 99 function values were generated uniformly at random in $[900,1000]$. Hence, each of these test problems has exactly one "steep" local minimum point, which is also the global minimum point. We shall refer to these test functions as Schoen3X, Schoen4X, and Schoen5X, respectively.

Finally, to show the importance of the restart strategy, we also created Schoen test functions with $k=100$ for dimensions $d=3,4,5$ where 3 of the function values $f_1,\ldots,f_k$ were generated uniformly at random in $[-1000,-500]$ while the other 97 function values were generated uniformly at random in $[900,1000]$. Hence, each of these test problems has exactly three "steep" local minima, one of which is the global minimum. We shall refer to these test functions as Schoen3Y, Schoen4Y, and Schoen5Y, respectively.

6.3 Experimental setup

To determine whether the proposed strategies are helpful, we will run Gutmann-RBF, CG-RBF and CORS-RBF with or without restart. Our CORS-RBF algorithms will

use a cycle of length 5 with a search pattern of $\langle 0.9, 0.75, 0.25, 0.05, 0.03, 0\rangle$. This choice of parameters was used in one of the implementations of CORS-RBF in Regis and Shoemaker (2005).

Each algorithm will be run 30 times on each test problem, where each trial uses a different SLHD of size $(d+1)(d+2)/2$ (d is the dimension). However, for comparison purposes, the same SLHD is used by the different RBF algorithms in a given trial. We chose SLHDs of size $(d+1)(d+2)/2$ since this is the minimum number of data points needed to fit a quadratic model, which is among the simplest nonlinear surfaces in $\mathbb{R}^d$.

For each test problem, we will record the average number of function evaluations an algorithm takes to get a solution with relative error $<1\%$. If f^* is the global minimum value of a function f and f_{best} is the best value obtained by an algorithm, then the *relative error* of the algorithm is given by $|f_{\text{best}} - f^*|/|f^*|$ provided that $f^* \neq 0$.

All RBF algorithms in this investigation adopted the strategy used by Gutmann (2001a), Björkman and Holmström (2000), and Regis and Shoemaker (2005) of replacing large function values by the median of all available function values whenever these values exceed the median. This is helpful in preventing oscillations in the RBF interpolant that are due to large differences in function values.

All numerical computations were performed in Matlab 7.0.4 on a 3.2 GHz Pentium 4 desktop. The solution of the global optimization subproblems in all RBF algorithms were carried out using a multistart implementation of the gradient-based solver FMINCON from the Matlab Optimization Toolbox (The Mathworks 2004), where derivatives are supplied for all the objective (i.e. $-\log(h_n(y))$ and $s_n(x)$) and constraint functions of the subproblems. In particular, for the global minimization of $s_n(x)$ and $-\log(h_n(y))$ over $\mathcal{D}$, we used the multistart approach known as multi level single linkage (Rinnooy Kan and Timmer 1987). For the global minimization of $s_n(x)$ subject to the distance constraints, we used a simple multistart approach that involves generating a sample from the feasible region, selecting some fraction of the sample that contains the best values for $s_n(x)$, and performing clustering to determine starting points for local minimization. Our implementation of Gutmann-RBF differs from the original implementation by Gutmann (2001a), which used a tunneling method to find the global minimum of the bumpiness function. In addition, our implementation of CORS-RBF also differs from the original implementation by Regis and Shoemaker (2005), which utilized the DIRECT algorithm (Jones et al. 1993) as implemented in Tomlab (Holmström 1999) to solve the CORS-RBF auxiliary problem (see Sect. 3.2). Note that the differences in how the global optimization subproblems are solved should not be important as long as each method solves each subproblem thoroughly.

7 Results and discussion

Table 2 shows the results of applying the different RBF algorithms on the 16 test problems. It records the average number of function evaluations required for an RBF algorithm to get a relative error of $<1\%$ on a test problem. All trials for CG-RBF-Restart and CORS-RBF-Restart were run until they got a relative error of $<1\%$. All trials for the other algorithms were only allowed to run for a maximum of 500 function evaluations. The reason for requiring CG-RBF-Restart and CORS-RBF-Restart to run to near optimality in each trial is that, we want to perform a paired t-test to compare the two methods later. In addition, we also recorded the standard errors of the mean for the CG-RBF-Restart and CORS-RBF-Restart algorithms in order to

provide some measure of variability in the results. The standard error of the mean is simply the standard deviation divided by the square root of the number of trials. The symbol > indicates that the average performance was computed with some runs being stopped before getting a relative error of <1%. The number after > is the average number of function evaluations to either get a relative error of <1% in less than 500 function evaluations or to terminate at exactly 500 function evaluations.

For convenience in the discussion below, we shall say that a trial is *successful* if it resulted in a relative error of <1% within 500 function evaluations; otherwise, we shall say that the trial is *unsuccessful*. The number inside the parenthesis in Table 2 represents the number of unsuccessful trials. In cases, where all trials are successful, we also performed paired t-tests at the 0.05 significance level to compare the performance of the different algorithms. The paired *t*-tests are essential for clarifying whether differences in performance actually exist by taking into account the variability in the results.

7.1 Performance of standard RBF algorithms initialized by SLHDs

Columns 2 and 6 of Table 2 show the results of the standard implementations of the two RBF methods (Gutmann-RBF and CORS-RBF) when initialized by SLHDs of size $(d + 1)(d + 2)/2$, where d is the dimension of the problem. For either Gutmann-RBF or CORS-RBF, there are many unsuccessful trials on the three Shekel problems, on Hartman6 and on the Schoen functions with three steep local minima.

Table 3 shows the original results for RBF methods obtained by Gutmann (2001a), Björkman and Holmström (RBFsolve) (2000), and Regis and Shoemaker (CORS-RBF) (2005) when using the corner points of the hypercube domain as the initial evaluation points. Gutmann (2001a) and Regis and Shoemaker (2005) used a thin plate spline RBF model while Björkman and Holmström (2000) used a cubic RBF model. Also, the results shown for Regis and Shoemaker (2005) used the same search pattern as the one used in this investigation. These results are better than the average results we obtained for Gutmann-RBF and CORS-RBF when using SLHDs. However, as noted in Sect. 6.1, the use of corner points is only feasible for low dimensional problems. Hence, it is important to determine how well RBF algorithms perform when using other types of space-filling experimental designs such as SLHDs.

Although the results shown in Table 3 are excellent, we should mention that Björkman and Holmström (2000) and Gutmann (2001b) also obtained unsatisfactory results when they experimented with different types of RBFs and various strategies for setting the target values. In particular, for the Shekel test problems, there were many combinations of RBF types and strategies where their RBF algorithm implementations did not get a relative error of < 1% after more than 150 function evaluations. Björkman and Holmström (2000) used all 2^d corner points of the hypercube domain for the initial evaluation points while Gutmann (2001b) used only $d+1$ corner points, specifically the point corresponding to the lower bounds on the variables and the d adjacent corners.

In the next two sections, we shall demonstrate that our proposed strategies will substantially reduce the number of unsuccessful trials for Gutmann-RBF and CORS-RBF on our test problems when they are initialized by SLHDs.

Table 2 Average number of function evaluations to get a relative error of < 1% for various RBF algorithms on 16 Test Problems

(1) Test function	(2) Gutmann-RBF	(3) CG-RBF	(4) Gutmann-RBF-Restart	(5) CG-RBF-Restart		(6) CORS-RBF	(7) CORS-RBF-Restart	
				Mean	SE		Mean	SE
Branin	40.53	45.47	40.53	46.60	4.29	43.90	43.90	2.46
Goldstein-Price	58.6	57.6	61.47	61.60	5.07	56.97	59.27	3.63
Hartman3	61.4	62.57	61.8	63.17	7.51	54.03	54.03	8.17
Shekel5	> 450.93 (26)	> 314.10 (16)	> 451.03 (24)	259.77 (2)	33.97	> 370.13 (20)	216.97 (2)	28.06
Shekel7	> 424.23 (23)	> 234.87 (9)	> 386.07 (17)	156.23	16.90	> 243.83 (10)	150.77	15.12
Shekel10	> 444.43 (23)	> 213.00 (8)	> 407.33 (18)	169.33	18.97	> 206.47 (9)	121.30	16.65
Hartman6	> 212.70 (9)	> 224.50 (10)	> 167.10 (1)	214.47 (3)	39.82	> 223.70 (9)	199.67 (2)	35.90
Schoen3	45.73	42.57	45.73	42.57	1.99	66.03	65.20	4.39
Schoen3X	109.8	52.47	129.23	52.47	1.59	46.60	46.60	1.64
Schoen3Y	> 416.70 (22)	> 396.67 (23)	> 380.43 (14)	290.70 (4)	52.45	> 378.93 (21)	296.50 (7)	48.22
Schoen4	62.6	72.33	62.6	77.60	9.41	> 106.03 (1)	92.53	11.86
Schoen4X	> 272.60 (5)	81.83	> 326.77 (10)	81.83	2.06	76.93	76.93	2.52
Schoen4Y	> 307.60 (12)	> 246.70 (12)	> 354.13 (13)	149.17	19.98	> 218.37 (10)	158.07 (2)	32.63
Schoen5	61.73	60.6	61.73	60.60	0.87	91.93	98.73	5.37
Schoen5X	> 307.37 (7)	84.3	> 442.17 (22)	86.20	3.05	72.00	72.00	1.60
Schoen5Y	> 326.33 (12)	> 273.37 (13)	> 398.17 (19)	177.70	19.82	> 260.20 (12)	196.97 (1)	25.68

All trials for CG-RBF-Restart and CORS-RBF-Restart were run until they got a relative error of <1%. All trials for the other algorithms were only allowed to run for a maximum of 500 function evaluations. A ">" symbol means that the average performance was computed with some runs being stopped before getting a relative error of <1%. The number inside the parenthesis represents the number of trials, out of 30, that did not get a relative error of <1% after 500 function evaluations

Table 3 Number of function evaluations to get a relative error of <1% for RBF algorithms on the Dixon–Szegö test problems

Test function	Gutmann-RBF	RBFsolve	CORS-RBF
Branin	44	26	40
Goldstein-Price	63	27	64
Hartman3	43	22	61
Shekel5	76	96	52
Shekel7	76	72	64
Shekel10	51	76	64
Hartman6	112	87	104

7.2 Effect of the strategies on Gutmann-RBF

Columns 2–5 of Table 2 show that CG-RBF-Restart (Column 5) is the best modification of Gutmann-RBF on the test problems considered. The improvements of CG-RBF-Restart (Column 5) over standard Gutmann-RBF (Column 2) are particularly remarkable on ten of the test problems, namely, the three Shekel functions, the six Schoen functions with steep local minima (Schoen3X, Schoen3Y, Schoen4X, Schoen4Y, Schoen5X, Schoen5Y), and Hartman6. For each of the six remaining test problems, a paired t-test found that CG-RBF-Restart is at least as good or significantly better than Gutmann-RBF.

We can examine the usefulness of just the restricted global minimization of the bumpiness function by comparing CG-RBF to Gutmann-RBF in terms of average performance or the number of unsuccessful trials in Columns 2 and 3 of Table 2. We can see that CG-RBF is better than Gutmann-RBF on seven test problems: the three Shekel functions, Schoen3, and the three Schoen functions with exactly one steep local minimum (Schoen3X, Schoen4X, and Schoen5X). From Columns 2, 3, and 5, it is worth noting that the improvements of CG-RBF-Restart over Gutmann-RBF on these Schoen functions are only due to the CG-RBF strategy. These results indicate that the restricted global minimization of the bumpiness function $B_n(y)$ is helpful for some problems with a steep global minimum. These results also confirm the failure of local search in Gutmann-RBF for the Shekel functions and the Schoen functions with one steep local minimum since better results are obtained when we force the algorithm to perform local search when W_n is small. The performances of CG-RBF and Gutmann-RBF are comparable on the nine remaining test problems. Hence, the restricted global minimization of $B_n(y)$ does not appear to hurt the performance of Gutmann-RBF on most of the other test problems. For problems with a few steep local minima, employing the restart strategy on top of the CG-RBF strategy yields much better results.

Columns 2 and 4 of Table 2 indicate that the restart strategy alone does not appear to be very effective in improving Gutmann-RBF on the test problems considered. The restart strategy improves the performance of Gutmann-RBF on Shekel7, Shekel10, Hartman6, and Schoen3Y but it also worsens the performance on Schoen5Y and the three Schoen functions with one steep local minimum. The performances of Gutmann-RBF-Restart and Gutmann-RBF are the same or comparable on the remaining test problems. However, from Columns 4 and 5 of Table 2, better results are obtained when the CG-RBF strategy is combined with the restart strategy on the three Shekel problems and the six Schoen problems with steep local minima.

7.3 Effect of the restart strategy on CORS-RBF

Columns 6 and 7 of Table 2 indicate that CORS-RBF-Restart is better than CORS-RBF on the three Shekel problems, Hartman6, Schoen4, and the Schoen test problems with three steep local minima (Schoen3Y, Schoen4Y, Schoen5Y). For each of the eight remaining test problems, a paired t-test shows that the difference in performance between CORS-RBF and CORS-RBF-Restart is not significant. Hence, these results suggest that the restart strategy is helpful for CORS-RBF on problems with a few steep local minima, one of which is the global minimum, and it does not appear to hurt performance on the other test problems.

7.4 Comparison between CG-RBF-Restart and CORS-RBF-Restart

We also performed paired t-tests at the 0.05 significance level to compare the average number of function evaluations to get within 1% of the optimal value for CG-RBF-Restart and CORS-RBF-Restart. The results show that the two RBF algorithms are comparable. In particular, although CORS-RBF-Restart is significantly better than CG-RBF-Restart on Shekel10, Schoen3X, and Schoen5X, the opposite is true for Schoen3 and Schoen5. Moreover, the difference in performance between the two methods is not significant on the remaining test problems.

7.5 Sensitivity of the Performance of RBF Methods to the Restart Parameters

Before we proceed, we note that there are reasonable settings for the values of the restart parameters. Given a small value for $C_{\max}$, a relatively large value of $I_{\min}$ would be generally ineffective since very few improvements would be considered *substantial*, resulting in frequent premature restarts. A large value of $C_{\max}$ would also be ineffective for some problems, since this would be almost the same as in the standard implementations of the RBF methods.

The original settings of $I_{\min} = 0.5\%$ and $C_{\max} = 5$ produced good results for CG-RBF-Restart and CORS-RBF-Restart on the test problems as can be seen from Table 2. Now, we run CG-RBF-Restart and CORS-RBF-Restart using other values of the restart parameters: (i) $I_{\min} = 0.5\%$ and $C_{\max} = 3$; (ii) $I_{\min} = 0.1\%$ and $C_{\max} = 5$; and (iii) $I_{\min} = 0.1\%$ and $C_{\max} = 3$. The results for CG-RBF-Restart and CORS-RBF-Restart are shown in Tables 4 and 5, respectively. As before, we included the standard errors of the mean to show some measure of variability in the results. Column 2 of Table 4 is the same as Column 5 of Table 2 while Column 2 of Table 5 is the same as Column 7 of Table 2. We included these results from Table 2 to facilitate the comparisons.

The results for CG-RBF-Restart in Table 4 are all much better than standard Gutmann-RBF on ten of the test problems: the three Shekel problems, the six Schoen functions with steep local minima, and Hartman6. Moreover, for the remaining problems except Schoen3, paired t-tests indicate that the differences in performance between CG-RBF-Restart and Gutmann-RBF are not statistically significant. Hence, the CG-RBF-Restart algorithms with the new values of the restart parameters are also generally better than standard Gutmann-RBF on the test problems.

Now we determine the effect of changing the restart parameters on the CG-RBF-Restart algorithm. We begin with the original settings $I_{\min} = 0.5\%$ and $C_{\max} = 5$ and consider the effect of decreasing $C_{\max}$ from 5 to 3 while holding $I_{\min} = 0.5\%$

Table 4 Average number of function evaluations to get a relative error of <1% for CG-RBF-Restart with different sets of parameter values on 16 test problems

(1) Test function	(2) CG-RBF-Restart $I_{\min} = 0.5\%$ $C_{\max} = 5$		(3) CG-RBF-Restart $I_{\min} = 0.5\%$ $C_{\max} = 3$		(4) CG-RBF-Restart $I_{\min} = 0.1\%$ $C_{\max} = 5$		(5) CG-RBF-Restart $I_{\min} = 0.1\%$ $C_{\max} = 3$	
	Mean	SE	Mean	SE	Mean	SE	Mean	SE
Branin	46.60	4.29	47.33	4.37	45.47	3.83	46.60	4.13
Goldstein-Price	61.60	5.07	74.30	7.80	60.33	4.77	66.47	5.22
Hartman3	63.17	7.51	66.30	7.92	63.17	7.51	63.43	7.67
Shekel5	259.77 (2)	33.97	306.27 (5)	41.11	253.30 (2)	32.35	257.13 (2)	31.44
Shekel7	156.23	16.90	192.30 (1)	27.98	169.80 (1)	19.46	164.27 (1)	17.63
Shekel10	169.33	18.97	181.20	17.16	167.10 (1)	21.65	175.57 (1)	20.43
Hartman6	214.47 (3)	39.82	162.27	21.69	287.90 (6)	62.68	211.37 (3)	36.22
Schoen3	42.57	1.99	43.50	2.56	42.57	1.99	42.57	1.99
Schoen3X	52.47	1.59	52.47	1.59	52.47	1.59	52.47	1.59
Schoen3Y	290.70 (4)	52.45	262.20 (4)	47.91	321.77 (4)	59.01	286.17 (4)	51.23
Schoen4	77.60	9.41	74.80	8.33	80.90	11.01	77.97	9.75
Schoen4X	81.83	2.06	89.33	5.55	81.83	2.06	81.83	2.06
Schoen4Y	149.17	19.98	153.03 (1)	21.46	157.60 (1)	22.91	148.40 (1)	20.29
Schoen5	60.60	0.87	60.60	0.87	60.60	0.87	60.60	0.87
Schoen5X	86.20	3.05	108.63	10.01	84.30	2.13	84.30	2.13
Schoen5Y	177.70	19.82	221.37 (2)	40.77	189.43 (1)	22.25	210.93 (1)	31.77

The number inside the parenthesis represents the number of trials, out of 30, that did not get a relative error of <1% after 500 function evaluations

Table 5 Average number of function evaluations to get a relative error of <1% for CORS-RBF-Restart with different sets of parameter values on 16 test problems

(1) Test function	(2) CORS-RBF-Restart $I_{\min} = 0.5\%$ $C_{\max} = 5$		(3) CORS-RBF-Restart $I_{\min} = 0.5\%$ $C_{\max} = 3$		(4) CORS-RBF-Restart $I_{\min} = 0.1\%$ $C_{\max} = 5$		(5) CORS-RBF-Restart $I_{\min} = 0.1\%$ $C_{\max} = 3$	
	Mean	SE	Mean	SE	Mean	SE	Mean	SE
Branin	43.90	2.46	46.00	3.02	43.90	2.46	44.87	2.67
Goldstein-Price	59.27	3.63	73.73	6.28	56.97	2.54	67.20	5.23
Hartman3	54.03	8.17	54.03	8.17	54.03	8.17	54.03	8.17
Shekel5	216.97 (2)	28.06	212.43 (2)	26.75	183.20	18.62	262.93 (2)	49.22
Shekel7	150.77	15.12	160.83 (1)	20.48	145.77	14.65	169.57 (1)	22.73
Shekel10	121.30	16.65	116.43	16.11	122.57	17.11	119.87	15.56
Hartman6	199.67 (2)	35.90	181.20 (2)	26.32	365.17 (5)	95.60	184.90 (2)	28.48
Schoen3	65.20	4.39	62.97	2.71	65.20	4.39	63.77	3.24
Schoen3X	46.60	1.64	46.60	1.64	46.60	1.64	46.60	1.64
Schoen3Y	296.50 (7)	48.22	196.43	22.26	436.03 (11)	68.05	269.63 (5)	46.92
Schoen4	92.53	11.86	111.47	13.06	90.23	12.31	93.30	10.62
Schoen4X	76.93	2.52	78.60	3.21	76.93	2.52	76.93	2.52
Schoen4Y	158.07 (2)	32.63	121.10	16.25	136.23 (1)	23.06	124.67	17.79
Schoen5	98.73	5.37	156.97 (1)	18.06	91.93	2.72	97.43	5.54
Schoen5X	72.00	1.60	72.00	1.60	72.00	1.60	72.00	1.60
Schoen5Y	196.97 (1)	25.68	174.10 (2)	24.64	228.40 (3)	35.71	190.80 (3)	27.36

The number inside the parenthesis represents the number of trials, out of 30, that did not get a relative error of <1% after 500 function evaluations

fixed. Although the results in Columns 2 and 3 of Table 4 seem to indicate considerable changes in performance on several test problems, paired t-tests indicate that statistically significant differences in performance are only present in Hartman6, Schoen3Y and Schoen4, where the performance became significantly better, and also in Schoen5X, where the performance became significantly worse. Next, when we decrease $I_{\min}$ from 0.5% to 0.1% while holding $C_{\max} = 5$ fixed (see Columns 2 and 4 of Table 4), there are statistically significant deteriorations in performance on Shekel7, Hartman6, and the Schoen problems with three steep local minima. Finally, when we simultaneously decrease $I_{\min}$ from 0.5% to 0.1% and decrease $C_{\max}$ from 5 to 3 (see Columns 2 and 5 of Table 4), there are no statistically significant changes in performance in any of the test problems. Overall, CG-RBF-Restart is somewhat sensitive to the restart parameters on some of the test problems.

The results for CORS-RBF-Restart in Table 5 are all much better than standard CORS-RBF on seven of the test problems: the three Shekel problems, the three Schoen functions with three steep local minima, and Hartman6. Moreover, for the remaining problems, paired t-tests indicate that the differences in performance between CORS-RBF-Restart and CORS-RBF are not statistically significant except in a few cases: Goldstein-Price when $C_{\max} = 3$ and $I_{\min} = 0.5\%$ or 0.1%; and Schoen5 when $C_{\max} = 3$ and $I_{\min} = 0.5\%$. Hence, the CORS-RBF-Restart algorithms with the new values of the restart parameters are also generally better than standard CORS-RBF on the test problems.

For CORS-RBF-Restart, changing the restart parameters did not result in statistically significant changes in performance except in a few cases. For example, decreasing $C_{\max}$ from 5 to 3 while holding $I_{\min} = 0.5\%$ fixed resulted in a statistically significant deterioration of performance on Goldstein-Price, Schoen4, and Schoen5 and it also resulted in a statistically significant improvement on Schoen3Y. Also, decreasing $I_{\min}$ from 0.5% to 0.1% while holding $C_{\max} = 5$ fixed resulted in a statistically significant deterioration on Schoen3Y.

8 Summary and conclusions

We evaluated the performance of Gutmann-RBF and CORS-RBF when they are initialized by SLHDs of size $(d+1)(d+2)/2$, where d is the dimension. We discovered that for some difficult test problems a sizable number of runs (where each run corresponds to a different SLHD) do not get within 1% of the global minimum value after a large number of function evaluations. In addition, for the Shekel problems from the Dixon and Szegö (1978) testbed, we observed slow convergence for Gutmann-RBF even when its best solution already lies in the basin of the global minimum point. Our computational experiments indicated that this was due to the failure of local search. Moreover, using a simple one-dimensional function, we showed how local search can fail in the standard implementation of Gutmann-RBF.

We proposed two strategies for dealing with the above problems. One strategy is to restrict the global minimization of the bumpiness function $B_n(y)$ in Gutmann-RBF to a relatively small subregion centered at a global minimizer of the RBF model whenever the weight parameter W_n is small, giving rise to the CG-RBF algorithm. This promotes a better balance between local and global search in Gutmann-RBF. The other strategy is to restart an RBF algorithm (either CG-RBF or CORS-RBF) completely from scratch whenever there is no substantial progress after some threshold number of consecutive iterations.

The computational results on the seven Dixon and Szegö (1978) test problems and on nine Schoen (1993) test problems indicate that the proposed strategies resulted in improved performance for Gutmann-RBF and CORS-RBF on many test problems while maintaining the performance on the other problems. In particular, the complete restart strategy was helpful for CORS-RBF while the combination of the CG-RBF strategy and the complete restart strategy was helpful for Gutmann-RBF. The paired t-tests also show that the modified RBF algorithms, namely CG-RBF-Restart and CORS-RBF-Restart, are comparable on the test problems considered when the restart parameters are set at $I_{\min} = 0.5\%$ and $C_{\max} = 5$. Finally, CG-RBF-Restart and CORS-RBF-Restart are also generally better than Gutmann-RBF and CORS-RBF, respectively, when using other reasonable values of the restart parameters. However, we also observed that these modified algorithms are somewhat sensitive to the two restart parameters on some of the test problems.

Acknowledgements We would like to acknowledge support to Prof. Shoemaker from the CISE Directorate in NSF, grant ACI-0305583. We are grateful to the Intelligent Information Systems Institute (IISI) directed by Dr. Carla Gomes for providing GRA funding for Rommel G. Regis (AFOSR, grant F49620-01-1-0076) during the early stages of this project. The computational experiments were performed using several desktop machines from the School of Operations Research & Industrial Engineering at Cornell University. We would also like to thank Dr. John Zollweg of the Cornell Theory Center for helping us write very efficient matlab codes which were helpful in getting the computational results in a reasonable amount of time. Finally, we would like to thank the referees for their valuable comments and suggestions.

References

Björkman, M., Holmström, K.: Global optimization of costly nonconvex functions using radial basis functions. Optim. Eng. **1**(4), 373–397 (2000)

Booker, A.J., Dennis, J.E., Frank, P.D., Serafini, D.B., Torczon, V., Trosset, M.W.: A rigorous framework for optimization of expensive functions by surrogates. Struct. Optim. **17**(1), 1–13 (1999)

Buhmann, M.D.: Radial Basis Functions. Cambridge University Press, UK (2003)

Conn, A.R., Scheinberg, K., Toint, Ph.L.: Recent progress in unconstrained nonlinear optimization without derivatives. Math. Program. **79**(3), 397–414 (1997)

Dixon, L.C.W., Szegö, G.: The global optimization problem: an introduction. In: Dixon, L.C.W., Szegö, G. (eds.) Towards Global Optimization 2 pp. 1–15. North-Holland, Amsterdam (1978)

Gomes, C.P., Selman, B., Crato, N., Kautz, H.: Heavy-tailed phenomena in satisfiability and constraint satisfaction problems. J. Automated Reasoning **24**, 67–100 (2000)

Gutmann, H.-M.: A radial basis function method for global optimization. J. Global Optim. **19**(3), 201–227 (2001a)

Gutmann, H.-M.: Radial basis function methods for global optimization. PhD Thesis, Department of Applied Mathematics and Theoretical Physics, University of Cambridge, UK (2001b)

Holmström, K.: The TOMLAB optimization environment in Matlab. Adv. Model. Optim. **1**(1), 47–69 (1999)

Horst, R., Pardalos, P.M., Thoai, N.V.: Introduction to Global Optimization, 2nd Edn. Kluwer, Dordrecht (2000)

Jones, D.R.: Global Optimization with Response Surfaces, presented at the *Fifth SIAM Conference on Optimization*, Victoria, Canada (1996)

Jones, D.R., Perttunen, C.D., Stuckman, B.E.: Lipschitz optimization without the lipschitz constant. J. Optim. Theor. Appli. **78**(1), 157–181 (1993)

Jones, D.R., Schonlau, M., Welch, W.J. Efficient global optimization of expensive black-box functions. J. Global Optim. **13**(4), 455–492 (1998)

Marazzi, M., Nocedal, J.: Wedge trust region methods for derivative free optimization. Math. Program. Ser. A **91**, 289–305 (2002)

McKay, M., Beckman, R., Conover, W.: A Comparison of three methods for selecting values of input variables in the analysis of output from a computer code. Technometrics **21**, 239–246 (1979)

Powell, M.J.D.: The theory of radial basis function approximation in 1990. In: Light, W. (ed.) Advances in Numerical Analysis, Vol. 2, Wavelets, Subdivision Algorithms and Radial Basis Functions, pp. 105–210. Oxford University Press, (1992)
Powell, M.J.D.: A review of algorithms for thin plate spline interpolation in two dimensions. In: Fontanella, F., Jetter, K., Laurent, P.J. (eds.) Advanced Topics in Multivariate Approximation, pp. 303–322. World Scientific Publishing, River Edge, NJ, (1996)
Powell, M.J.D.: Recent research at cambridge on radial basis functions. In: Müller, M., Buhmann, M., Mache, D. and Felten, M. (eds.) New Developments in Approximation Theory, International Series of Numerical Mathematics, Vol. 132, pp. 215–232. Birkhauser Verlag, Basel, (1999)
Powell, M.J.D.: UOBYQA: unconstrained optimization by quadratic approximation. Math. Program. **92**, 555–582 (2000)
Powell, M.J.D.: On trust region methods for unconstrained minimization without derivatives. Math. Program. **97**, 605–623 (2002)
Regis, R.G., Shoemaker, C.A.: Constrained global optimization using radial basis functions. J. Global Optim. **31**, 153–171 (2005)
Rinnooy Kan, A.H.G., Timmer, G.T.: Stochastic global optimization methods, part II: multi level methods. Math. Program. **39**, 57–78 (1987)
Serafini, D.B.: A framework for managing models in non-linear optimization of computationally expensive functions. Ph.D. Thesis, Department of Computational and Applied Mathematics, Rice University (1998)
Schoen, F.: A wide class of test functions for global optimization. J. Global Optim. **3**, 133–137 (1993)
The Mathworks Inc. Optimization Toolbox for use with MATLAB: User's Guide, Version 3, Natick, MA (2004)
Torczon, V.: On the convergence of pattern search algorithms. SIAM J. Optim. **7**(1), 1–25 (1997)
Torn, A., Zilinskas, A.: Global Optimization. Lecture Notes in Computer Science, Vol. 350, Springer-Verlag, Berlin (1989)
Ye, K.Q., Li, W., Sudjianto, A.: Algorithmic construction of optimal symmetric latin hypercube designs. J. Stat. Plan. Infer. **90**, 145–159 (2000)