Optimizing Radial Basis Functions by D.C. Programming and its use in Direct Search for Global Derivative-Free Optimization

Le Thi Hoai An[*] A. I. F. Vaz[†] L. N. Vicente[‡]

October 6, 2009

Abstract

In this paper we address the global optimization of functions subject to bound and linear constraints without using derivatives of the objective function. We investigate the use of derivative-free models based on radial basis functions (RBFs) in the search step of direct-search methods of directional type. We also study the application of algorithms based on difference of convex (d.c.) functions programming to solve the resulting subproblems which consist of the minimization of the RBF models subject to simple bounds on the variables. Extensive numerical results are reported with a test set of bound and linearly constrained problems.

Keywords: Global optimization, derivative-free optimization, direct-search methods, search step, radial basis functions, d.c. programming, DCA.

1 Introduction

We are interested in solving optimization problems of the form

$$\min_{x\in\mathbb{R}^n} f(x) \quad \text{s.t.} \quad x \in \Omega,$$

without using derivatives of the function f and when the feasible set Ω is a polyhedron. We will consider in more detail the case where Ω is solely defined by lower and upper bounds on the variables

$$\Omega = \{x \in \mathbb{R}^n : \ell \le x \le u\}. \tag{1}$$

[*]Laboratory of Theoretical and Applied Computer Science (LITA EA 3097), Paul Verlaine University, Metz, Ile du Saulcy, 57045 Metz, France (`lethi@univ-metz.fr`).

[†]Department of Systems and Production, University of Minho, Campus de Gualtar, 4710-057, Portugal (`aivaz@dps.uminho.pt`). Support for this author was provided by Algoritmi Research Center.

[‡]CMUC, Department of Mathematics, University of Coimbra, 3001-454 Coimbra, Portugal (`lnv@mat.uc.pt`). Support for this author was provided by FCT under the grant PTDC/MAT/64838/2006.

In (1) the inequalities $\ell \leq x \leq u$ are posed componentwise and $\ell \in (-\infty, \mathbb{R})^n$, $u \in (\mathbb{R}, +\infty,)^n$, and $\ell < u$.

Our approach to address this type of problems is to incorporate the minimization of a radial basis functions (RBF) model in the search step of direct-search methods of directional type. This class of methods has been extensively studied in the literature (see the survey paper by Kolda, Lewis, and Torczon [19] or Chapter 7 of the book by Conn, Scheinberg, and Vicente [6]). Under appropriate assumptions they guarantee global convergence to stationary points. Their ability to find global minima depends on the incorporation of methods or heuristics for global optimization in their so-called search step. Two illustrative examples of such hybridizations are the approaches of Vaz and Vicente [37, 38], where a population-based heuristic was applied or, more recently, the approach of Griffin and Kolda [11], where DIRECT [16] was the chosen global optimization method.

On the other hand, models based on RBFs have been shown to be of interest for global optimization. In fact, derivative-free global optimization methods based on radial basis functions have been proposed by several authors (see the references [5, 12, 17] and the sequence of papers by Regis and Shoemaker [31, 32, 33, 34] which includes extensions to the constrained and parallel cases).

The overall direct-search algorithmic structure chosen for this paper is simple, but relatively efficient and robust as we will show by reporting extensive numerical experiments. In every iteration of direct search, we attempt to form a RBF model with as many points as possible and then to minimize it in the intersection of Ω with a trust region whose radius is proportional to the direct-search step size. We choose an ℓ_∞-shape trust region so that, in the case where Ω is given by (1), this intersection is still a box-constrained set. When Ω is defined by linear constraints not of the simple bound type, we apply some approximation procedure so that we still consider subproblems formed by the minimization of RBFs subject to box constraints. Our main conclusion is that a direct-search method with an RBF model minimization in the search step offers a good compromise between global optimization and computational effort (i.e., number of function evaluations), especially in the linearly constrained case.

RBFs seem to offer a number of natural ways into which can be decomposed as a difference of two convex functions. In this paper, we will introduce two of such d.c. decompositions and adapt the d.c. algorithm (DCA) of [4] to both. The most efficient one is used to solve the subproblems which arise in the search step.

The structure of the paper is as follows. In Sections 2 and 3, we present some background material on radial basis functions and d.c. programming. The application of the DCA to the minimization of RBFs is studied in Section 4. The use of RBFs in the context of derivative-free optimization is described in Section 5. Finally, we provide numerical results with bound and linearly constrained problems in Section 6 and conclude the paper in Section 7.

2 Radial basis functions for optimization

In order to interpolate a function f whose values on a set $Y = \{y^1, \dots, y^{n_p}\} \subset \mathbb{R}^n$ are known, one can use a radial basis functions (RBF) model of the form

$$m(x) \;=\; \sum_{i=1}^{n_p} \lambda_i \phi(\|x - y^i\|), \tag{2}$$

where $\phi(\|\cdot\|)$, with $\phi : \mathbb{R}_+ \to \mathbb{R}$, is a radial basis function and $\lambda_1, \dots, \lambda_{n_p} \in \mathbb{R}$ are parameters to be determined.

For $m(x)$ to be twice continuously differentiable, the function $\phi(x)$ must be both twice continuously differentiable and have a derivative that vanishes at the origin. The cubic RBF, defined by $\phi(r) = r^3$, is among the most popular (twice continuously differentiable) radial basis functions, and has been frequently used for optimization purposes. Other popular RBFs, also twice continuously differentiable, are the Gaussian, the multiquadratic, and the inverse multiquadric.

The term *radial basis* comes from the fact that $\phi(\|x\|)$ is constant on any sphere centered at the origin in $\mathbb{R}^n$. In many applications, it is desirable that the linear space spanned by the basis functions include constant or linear functions. Thus, it turns out to be useful to augment the radial basis function model in (2) by a low-order *polynomial tail* $\sum_{j=0}^{q} \gamma_j p_j(x)$, where p_j, $j = 0, \dots, q$, are some basis functions for the polynomial and $\gamma_0, \dots, \gamma_q \in \mathbb{R}$. The new model is now of the form

$$m(x) \;=\; \sum_{i=1}^{n_p} \lambda_i \phi(\|x - y^i\|) + \sum_{j=0}^{q} \gamma_j p_j(x).$$

Furthermore, the coefficients λ's are required to satisfy

$$\sum_{i=1}^{n_p} \lambda_i p_j(y^i) \;=\; 0, \quad j = 0, \dots, q.$$

These, in conjunction with the interpolation conditions $m(y^i) = f(y^i)$, $i = 1, \dots, n_p$, give the linear system

$$\begin{bmatrix} \Phi & P \\ P^\top & 0 \end{bmatrix} \begin{bmatrix} \lambda \\ \gamma \end{bmatrix} = \begin{bmatrix} f(Y) \\ 0 \end{bmatrix}, \tag{3}$$

where $\Phi_{ij} = \phi(\|y^i - y^j\|)$ for $i, j \in \{1, \dots, n_p\}$, $P_{ij} = p_j(y^i)$ for $i \in \{1, \dots, n_p\}$, $j \in \{0, \dots, q\}$, and $f(Y)$ is the vector formed by the values $f(y^1), \dots, f(y^{n_p})$.

The polynomial tails most frequently used in the context of RBFs are linear, and we will write $t(x) = c + g^\top x$ and

$$m(x) \;=\; \sum_{i=1}^{n_p} \lambda_i \phi(\|x - y^i\|) + t(x). \tag{4}$$

This model has n_p+n+1 parameters, n_p for the radial basis terms and $n+1$ for the linear polynomial terms. However, when the number of points is $n+1$ (or less), the solution of the interpolation system gives rise to a linear polynomial, since all the parameters λ_i, $i=1,\ldots,n_p$, are zero (see the second block equation in (3)). Consequently, the simplest nonlinear model $m(x)$ of the form (4) is based on $n+2$ interpolation points (and has $2n+3$ parameters).

The approaches by Oeuvray and Bierlaire [28, 29] and Wild, Regis, and Shoemaker [39] for derivative-free optimization use cubic radial basis functions and linear polynomial tails:

$$m(x) \;=\; \sum_{i=1}^{n_p} \lambda_i \|x-y^i\|^3 + t(x). \tag{5}$$

3 A brief review of d.c. programming

D.C. programming addresses the problem of minimizing a function f which is a difference of convex functions on the whole space $\mathbb{R}^p$ or on a convex set $C \subset \mathbb{R}^p$. Generally speaking, a d.c. program takes the form

$$\beta \;=\; \inf\{f(x)=g(x)-h(x) : x \in \mathbb{R}^p\}, \qquad (P_{dc})$$

where g and h are in the set $\Gamma_0(\mathbb{R}^p)$ of all lower semicontinuous proper convex functions in $\mathbb{R}^p$. Such a function f is called a d.c. function and $g-h$ the d.c. decomposition of f, while g and h are the d.c. components of f. The convex constraint $x \in C$ can be incorporated in the objective function of (P_{dc}) by using the indicator function on C denoted χ_C which is defined by $\chi_C(x)=0$ if $x \in C$ and by $+\infty$ otherwise.

D.C. programming is one of the most relevant tools in nonsmooth nonconvex programming and global optimization. D.C. algorithms (DCA) were introduced by Pham Dinh Tao (see [7]) in their preliminary form in 1985 and have been extensively developed since 1994 by Le Thi Hoai An and Pham Dinh Tao, see [2, 3, 4] and the references therein. DCA have been successfully applied to many large-scale (smooth or nonsmooth) nonconvex programs in different fields of applied sciences for which they often give global solutions. The algorithms have proved to be more robust and efficient than some of the standard methods.

Recall that, for $\theta \in \Gamma_0(\mathbb{R}^p)$ and $\bar{x} \in dom\,\theta = \{x \in \mathbb{R}^p : \theta(\bar{x}) < +\infty\}$, the subdifferential $\partial\theta(\bar{x})$ of θ at $\bar{x}$ is defined as

$$\partial\theta(\bar{x}) \;=\; \{y \in \mathbb{R}^p : \theta(x) \geq \theta(\bar{x}) + (x-\bar{x})^\top y, \forall x \in \mathbb{R}^p\}.$$

A point x^* is critical or stationary for the minimization of $g-h$ when

$$\partial h(x^*) \cap \partial g(x^*) \;\neq\; \emptyset.$$

Let $g^*(y) := \sup\{x^\top y - g(x) : x \in \mathbb{R}^p\}$ be the conjugate function of g. Then, the following program is called the dual program of (P_{dc}):

$$\beta_D \;=\; \inf\{h^*(y) - g^*(y) : y \in \mathbb{R}^p\}. \qquad (D_{dc})$$

One can prove that $\beta = \beta_D$, (see, e.g., [4]) and there exists a perfect symmetry between the primal and dual d.c. programs: the dual of (D_{dc}) is exactly (P_{dc}).

The transportation of global solutions between (P_{dc}) and (D_{dc}) is expressed by:

$$[\cup_{y^* \in \mathcal{D}_{dc}} \partial g^*(y^*)] \subset \mathcal{P}_{dc}, \quad [\cup_{x^* \in \mathcal{P}_{dc}} \partial h(x^*)] \subset \mathcal{D}_{dc},$$

where $\mathcal{P}_{dc}$ and $\mathcal{D}_{dc}$ denote the solution sets of (P_{dc}) and (D_{dc}) respectively. Under certain conditions, this property also holds for the local solutions of (P_{dc}) and (D_{dc}), in the following sense. Let x^* be a local solution to (P_{dc}) and let $y^* \in \partial h(x^*)$. If g^* is differentiable at y^*, then y^* is a local solution to (D_{dc}). Similarly, let y^* be a local solution to (D_{dc}) and let $x^* \in \partial g^*(y^*)$. If h is differentiable at x^*, then x^* is a local solution to (P_{dc}).

Based on local optimality conditions and duality in DCA, the idea of DCA is quite simple: each iteration k of DCA approximates the concave part $-h$ by its affine majorization (that corresponds to taking $y^k \in \partial h(x^k)$) and minimizes the resulting convex function (that is equivalent to determining $x^{k+1} \in \partial g^*(y^k)$).

```
**Generic DCA scheme**
**Initialization:** Let $x^0 \in \mathbb{R}^p$ be a best guess, $0 \leftarrow k$.
**Repeat**
    Calculate $y^k \in \partial h(x^k)$
    Calculate $x^{k+1} \in \arg\min\{g(x) - h(x^k) - (x - x^k)^\top y^k : x \in \mathbb{R}^p\}$   $(P_k)$
    $k + 1 \leftarrow k$
**Until** convergence of $\{x^k\}$.
```

Convergence properties of the DCA and its theoretical basis are described in [2, 3, 4]. However, for what comes next it is worthwhile to report the following properties:

- DCA is a descent method *without* line search, say the sequence $\{g(x^k) - h(x^k)\}$ is decreasing.
- If $g(x^{k+1}) - h(x^{k+1}) = g(x^k) - h(x^k)$, then x^k is a critical point of $g - h$. In this case, DCA terminates at the k-th iteration.
- If the optimal value β of problem (P_{dc}) is finite and the infinite sequence $\{x^k\}$ is bounded, then every limit point of this sequence is a critical point of $g - h$.
- DCA has a linear convergence for general d.c. programs.

Note that a d.c. function f has *infinitely many* d.c. decompositions and that the choice of a d.c. decomposition influences the speed of convergence, robustness, efficiency, globality of computed solutions of DCA. For a given d.c. program, the choice of the *optimal* d.c. decompositions is still open and depends strongly on the structure of the problem being considered. In practice, one chooses g and h such that the sequences $\{x^k\}$ and $\{y^k\}$ can be easily calculated.

4 Optimizing RBFs using d.c. algorithms

The optimization problem we are addressing in this section is

$$\min \quad m(x) \quad \text{s.t.} \quad x \in \bar{\Omega}, \tag{6}$$

where $\bar{\Omega}$ is the feasible region defined by upper and lower bounds on the variables, *i.e.*, $\bar{\Omega} = \{x \in \mathbb{R}^n : \bar{\ell} \le x \le \bar{u}\}$.

When ϕ is convex in $[0, +\infty)$, one possible d.c. decomposition of the RBF model (4) in $\bar{\Omega}$ is given by

$$m(x) \;=\; g_1(x) - h_1(x),$$

where

$$g_1(x) \;=\; \sum_{\lambda_i \ge 0} \lambda_i \phi(\|x - y^i\|) + t(x) + \chi_{\bar{\Omega}}(x), \qquad h_1(x) \;=\; \sum_{\lambda_i < 0} (-\lambda_i)\phi(\|x - y^i\|),$$

and $\chi_{\bar{\Omega}}(x)$ is the indicator function associated with $\bar{\Omega}$. The d.c. algorithm (DCA) corresponding to this decomposition is as follows.

Algorithm 4.1 (d.c. algorithm 1 (DCA1))

```
Initialization
    Choose $x_0$.

For $k = 0, 1, 2, \ldots$
    1. $y_k = \nabla h_1(x_k)$.
    2. Compute $x_{k+1}$ as the solution of
       $\min \; g_1(x) - \big(h_1(x_k) + (x - x_k)^\top y_k\big) \;\; \text{s.t.} \;\; x \in \bar{\Omega}.$   (7)
```

Another possible d.c. decomposition for the RBF model (4) in $\bar{\Omega}$ is the following

$$m(x) \;=\; g_2(x) - h_2(x),$$

where

$$g_2(x) \;=\; \frac{\rho}{2}\|x\|^2 + t(x) + \chi_{\bar{\Omega}}(x), \qquad h_2(x) = \frac{\rho}{2}\|x\|^2 - (m(x) - t(x)),$$

$\chi_{\bar{\Omega}}(x)$ is, again, the indicator function associated with $\bar{\Omega}$, and

$$\rho \;=\; \max_{x \in \bar{\Omega}} \|\nabla^2 (m(x) - t(x))\|.$$

Denoting by $P_S(s)$ the projection of s onto the set S, the DCA becomes then the following.

Algorithm 4.2 (d.c. algorithm 2 (DCA2))

```
Initialization
    Choose $x_0$. Compute $\rho$.

For $k = 0, 1, 2, \ldots$
        1. $y_k = \nabla h_2(x_k)$.
        2. $x_{k+1} = P_{\bar{\Omega}}(y_k/\rho)$.
```

We have performed a number of numerical experiments in MATLAB R14 (7.0.1) [22] with both algorithms to assess their potential in the case where cubic RBFs are chosen (5). Both algorithms were compared against `fmincon` of MATLAB. We will report below an illustrative snapshot of these experiments. All tests were run in a Pentium Centrino (2.0GHz and 2Gb of RAM).

An upper bound for ρ can be computed as follows when $\phi(r) = r^3$ and $\bar{\Omega}$ is defined as in (6):

$$\rho \;=\; 12 n_p \|\lambda\|_\infty \max\{\|\bar{\ell}\|, \|\bar{u}\|\}. \tag{8}$$

The derivation of this upper bound is based on the fact that when ϕ is twice continuously differentiable, one has

$$\nabla^2 m(x) \;=\; \sum_{i=1}^{n_p} \lambda_i \Theta(x - y^i),$$

with

$$\Theta(r) \;=\; \begin{cases} \frac{\phi'(\|r\|)}{\|r\|} I + \left(\phi''(r) - \frac{\phi'(\|r\|)}{\|r\|}\right) \frac{r}{\|r\|} \frac{r^\top}{\|r\|} & \text{if } r \neq 0, \\ \phi''(0) I & \text{if } r = 0. \end{cases}$$

We generated 10 problems in $[-1,1]^n$ with $n = 50$ by first randomly generating $n_p = 100$ sample points and then computing the corresponding model coefficients λ and γ. The function values were also randomly generated in $[-3,-1]$ and $[1,3]$. The starting point was set to the origin for all the three methods.

We used `fmincon` to solve the subproblems (7) in DCA1. The starting point was the final point obtained in the previous iteration (except in the first iteration where we have used the origin). The `fmincon` stopping tolerances were set to 10^{-1}. The stopping criteria of DCA1 was satisfied when the absolute error of two consecutive iterations does not exceeded 10^{-4} or the number of iterations exceeded 30000.

The stopping criteria of DCA2 was satisfied when the absolute error of two consecutive iterations does not exceeds 10^{-6} or the number of iterations exceeded 30000. We tested two versions. In the first version (DCA2a), the value of ρ is kept constant as in (8). In the second version (DCA2b), we start with an initial value for ρ given by (8) multiplied by 5×10^{-6}, and increase it by a factor of two each time a new point does not lead to a simple improvement in the objective function value, until the upper bound in (8) is reached.

	DCA1		DCA2a		DCA2b		fmincon	
n_g	f final	CPU	f final	CPU	f final	CPU	f final	CPU
25	-19.7301	264.34	-19.7378	0.03	-19.7200	0.02	-19.7334	0.50
50	-23.2822	174.66	-23.2800	0.02	-23.2610	0.01	-23.2641	0.47
75	-20.3773	187.40	-20.3774	0.04	-20.3213	0.01	-20.3774	0.51

Table 1: Results for the minimization of RBFs in box domains ($n = 50$). The number of sample points is $n_p = 100$ (n_g of them have positive function values). The two versions of DCA2 differ only in the values for ρ.

When applying the fmincon to solve problem (6) we set the stopping tolerances to 10^{-5}. Only first-order derivatives were provided.

The results presented in Table 1 summarize the average behavior of the methods for the problems under consideration. We can observe that DCA2 is typically 10-20 times faster than fmincon. The high CPU time taken by DCA1 is due to the necessity of solving a subproblem in each iteration.

5 An application in global DFO

Now we investigate the use of radial basis models in the enhancement of direct-search methods for global derivative-free optimization. Our approach consists of forming and minimizing an RBF model in the search step of direct-search methods of the directional type. The iterations of such methods can be divided into two main steps (a search step and a poll step). For the purposes of global convergence to stationary points, the search step is optional and free of any rules except that it must terminate finitely and yield a point in the underlying integer lattice.

The algorithmic description below applies to the case where Ω is solely defined by bound constraints (1). In this situation, the poll step makes use of a constant positive spanning set that includes the coordinate vectors and their negatives, which conform to the feasible set. The linearly constrained case is discussed afterwards.

Algorithm 5.1 (Direct-search method (RBFs in search step))

Initialization

Choose x_0, $\alpha_0 > 0$, $\Delta_0 > 0$. Let $D = \{e_1, \ldots, e_n, -e_1, \ldots, -e_n, e, -e\}$ (where e_i denotes the i-th column of the identity and e the vector of ones, of dimensions n). Let $n_{min} = n+2$ and $n_{max} = (n+1)(n+2)/2$ be the minimum and maximum number of points used to build the RBF models. (The maximum number of points in cache considered for the model building is $50(n+1)$.)

For $k = 0, 1, 2, \ldots$

1. **Search step:** Skip the search step if the number of points previously evaluated is less than n_{min}.

   Use all the previously evaluated points if its number is lower then n_{max}.

   If there are more previously evaluated points in the cache than n_{max}, 80% of the points for model building are selected as the ones nearest to the current iterate. The last 20% are chosen as the ones further away from the current iterate.

   Form a RBF model (5) based on the selected previously evaluated points.

   Minimize the RBF model by solving the problem

   $$\min_{x \in \Omega \cap B(x_k;\Delta_k)} m(x) \tag{9}$$

   where $B(x_k;\Delta_k) = \{x \in \mathbb{R}^n : \|x - x_k\|_\infty \leq \Delta_k\}$, $\Delta_k = \sigma\alpha_k$, and σ takes the value 1 if the previous search step was unsuccessful, or 2 otherwise.

   If the solution $\bar{x}_k$ of the subproblem satisfies $f(\bar{x}_k) < f(x_k)$ then set $x_{k+1} = \bar{x}_k$, declare the iteration and the search step successful, and skip the poll step.

2. **Poll step:** Optionally order the poll set $P_k = \{x_k + \alpha_k d : d \in D\}$[1]. Start evaluating f at the poll points following the chosen order. If a poll point $x_k + \alpha_k d_k$ is found such that $f(x_k + \alpha_k d_k) < f(x_k)$ then stop polling, set $x_{k+1} = x_k + \alpha_k d_k$, and declare the iteration and the poll step successful. Otherwise declare the iteration (and the poll step) unsuccessful and set $x_{k+1} = x_k$.

3. **Step size update:** If the iteration was successful then maintain the step size parameter ($\alpha_{k+1} = \alpha_k$) or double it ($\alpha_{k+1} = 2\alpha_k$) after two consecutive poll successes along the same direction. If the iteration was unsuccessful, halve the step size parameter ($\alpha_{k+1} = \alpha_k/2$).

We provide some implementation details in the following subsections.

5.1 Linearly constrained problems

The subproblem considered in the search step consists of the minimization of an RBF model on the intersection of Ω with an ℓ_∞-shape trust region. When Ω is defined by bounds, as in (1), this amounts to a box-constrained domain and the DCAs of the previous section can be easily applied. If Ω contains linear constraints which are not simple bounds, then we simplify the subproblem by temporarily removing those constraints. The point to be evaluated in the search step is then defined by $x_k + \min\{1, \tau_k\}(\bar{x}_k - x_k)$, where τ_k is the largest positive scalar such that $x_k + \min\{1, \tau_k\}(\bar{x}_k - x_k) \in \Omega$. This procedure is similar to the one used in [38] to address infeasible points in the search step.

To obtain an feasible initial guess and a set of poll directions which conforms to the feasible set, when Ω contains linear constraints which are not simple bounds, we also use the same strategies as in [38].

[1]The points in P_k can be ordered by increasing values of the RBF model.

5.2 Cache

In order to take the maximum advantage of previous implementations of the algorithm described in [37, 38], the `PSwarm` solver was coupled with a cache for the true function evaluations. This cache can be used by both the search and poll steps when a function evaluation is requested. The cache proved not to be useful in the particle swarm search step as the number of hits in the cache was very low. For the herein proposed version the cache implementation is mandatory, since the RBF model building relies on previous evaluated objective function values.

6 Numerical results for bound and linearly constrained optimization problems

We report numerical results with the `PSwarm` version 2.1 changed to include the setup and minimization of an RBF model in the search step. The solver is implemented in MATLAB R14 (7.0.1) [22]. All tests were run in a Pentium Centrino (2.0GHz and 2Gb of RAM). As the problems are modeled in AMPL [9], an AMPL-MATLAB interface was used, as described in [37, 38]. The test set used in the numerical results includes 119 bound constrained problems and 109 linearly constrained problems coded in the AMPL format.

The bound constrained problems are the ones reported in [37]. These problems are global optimization test problems collected from the literature ([1, 13, 14, 18, 20, 21, 25, 30]) and coded in AMPL. All coded problems have lower and upper bounds on the variables. The problems description and their source are available at `http://www.norg.uminho.pt/aivaz/pswarm`.

The linearly constrained problems were gathered from a total of 1564 problems, collected from the following sources: Vanderbei [36] (given in AMPL, which includes the CUTE [10] collection), GLOBALlib [26] (available in AMPL format at [27]), one problem from [35], three problems from [15], one from [23], and four from [24]. These problems can also be obtained at `http://www.norg.uminho.pt/aivaz/pswarm`.

We provide aggregated results for all the test problems, results for the class of bound constrained problems, and results for the class of linearly constrained problems. Showing the results in this way allows us to suggest different algorithmic options for each class of problems.

We are reporting numerical results for the RBF search step with DCA as the algorithm used to solve problem (9) (RBF-DCA), for the RBF search step with `fmincon` as the algorithm used to solve problem (9) (RBF-fmincon), for an empty search step (Pattern), for the particle swarm search step (PSwarm), and for the RBF search step with DCA as the algorithm to solve problem (9) and the directions used in the poll step sorted by the RBF model value at the poll points (RBF-DCA Sort). In all cases, the stopping criteria consisted of reaching a maximum budget of 2000 function evaluations or driving the step size parameter α_k below 10^{-5}.

The algorithm used in the RBF-DCA versions is the DCA2b described in Section 4 for cubic RBFs. A limit of 3000 iterations for the DCA2 algorithm was set, as an approximation with high accuracy to the problem (9) solution is not required. Even when the DCA algorithm is unable to converge with the requested accuracy, the last iterate is used to check for progress in the objective function in the search step.

For a better visualization, brevity, and clarity of the numerical results, we are providing performance profiles obtained by using the procedure described in [37] (a modification of the performance profiles from [8]). The major advantage of the performance profiles is that they can be presented in one figure, by plotting, for the different solvers, a cumulative distribution function $\upsilon(\tau)$ representing a performance ratio.

The performance ratio is defined by setting $r_{p,s} = \frac{t_{p,s}}{\min\{t_{p,s}: s \in \mathcal{S}\}}$, $p \in \mathcal{P}$, $s \in \mathcal{S}$, where $\mathcal{P}$ is the test set, $\mathcal{S}$ is the set of solvers, and $t_{p,s}$ is the value obtained by solver s on test problem p. Then, define $\upsilon_s(\tau) = \frac{1}{N_p}\text{size}\{p \in \mathcal{P} : r_{p,s} \leq \tau\}$, where N_p is the number of test problems. The value of $\upsilon_s(1)$ is the probability that the solver will win over the remaining ones (meaning that it will yield a value lower than the values of the remaining ones). If we are only interested in determining which solver is the best (in the sense that wins the most), we compare the values of $\upsilon_s(1)$ for all the solvers. At the other end, solvers with the largest probabilities $\upsilon_s(\tau)$ for large values of τ are the most robust ones (meaning that are the ones that solved more problems).

Figure 1 presents a comparison between the DCA algorithm and `fmincon` when using a RBF model in the search step. Additionally we include the case where the search directions of the poll step are sorted accordingly to the RBF model. For this comparison, we have set $t_{p,s} = \frac{b}{a}$, where a is the number of models leading to an improvement in the objective function and b is the number of models built, corresponding to the inverse of the percentage of success in using the RBF model. When none of the model building leads to a success we have $t_{p,s} = +\infty$. Observing the profiles, we can see that in nearly 30% of the problems all the versions where able to obtain the best $t_{p,s}$ value (around 1). Looking at large values of τ, we observe that only in 75% of the problems the model has a nonzero success rate. A careful inspection to problems where the RBF model has a zero number of improvements shows that AMPL is providing an initial guess that is already the problem global optima (AMPL considers the origin as the initial guess when it is not provided by the user and in fact the origin is the global optima for some problems).

For this comparison between success rates, the specific profiles for bound and linearly constrained problems are omitted, since they provide similar results. While the results are comparable, the simplicity of the DCA algorithm completely justifies its use in our software. Furthermore, the inclusion of DCA makes the solver widely available since it does not depend on a license for the MATLAB optimization toolbox. We have also tried to solve the RBF subproblems up to global optimization in an attempt to improve the success rate, but we did not observe any significant difference.

For the remaining analysis, we have removed the problems where the success rate was zero for all the RBF versions (for these problems the search and poll steps always reduce to failures). The problems used are listed in Table 2 for the bound constrained case and in Table 3 for the linearly constrained case.

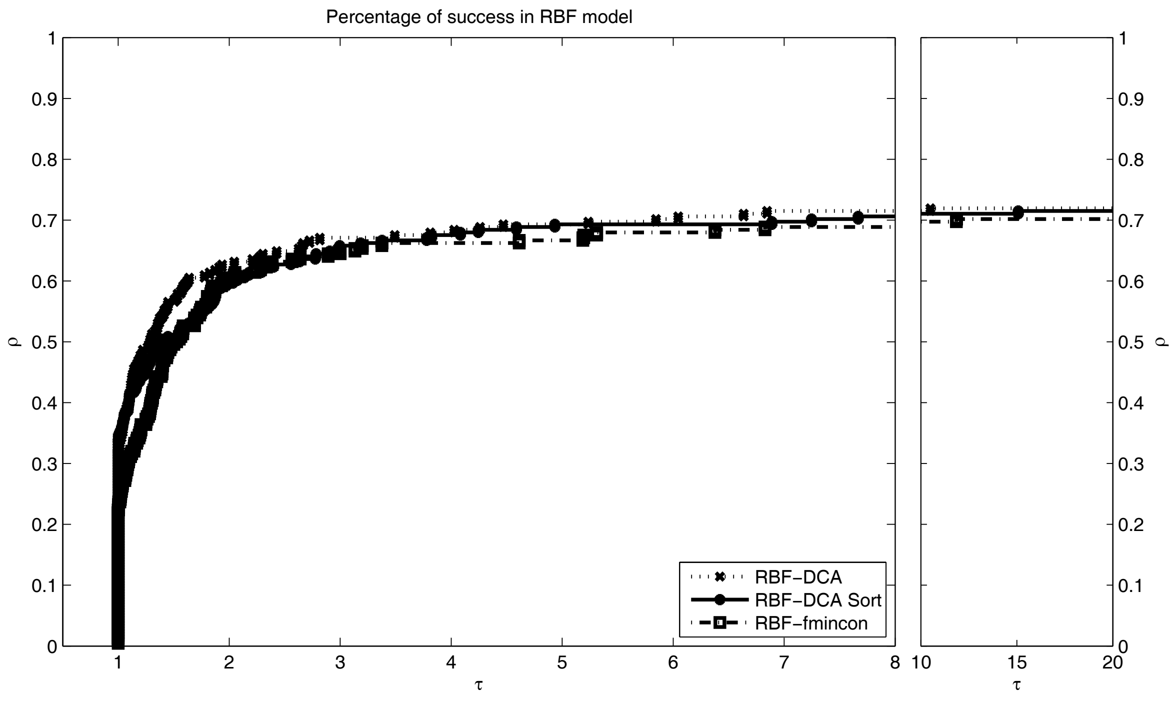

Figure 1: Comparison among and RBF-DCA, RBF-fmincon, and RBF-DCA Sort, for all test problems.

ap	fx_10	ir2	lj3_98	ml_5	osp_20	sin_10	zlk3a
bhs	fx_5	ir5	lm1	mr	plj_38	sin_20	zlk3b
bl	gp	kl	lms1a	ms1	plj_75	st_17	zlk3c
bp	grp	lj1_38	lms1b	ms2	plj_98	st_9	zlk4
cb6	h3	lj1_75	lms2	nf2	ptm	swf	zlk5
da	h6	lj1_98	lms5	nf3_10	rb	sz	zzs
em_10	hm	lj2_38	lv8	nf3_15	s10	szzs	
em_5	hm1	lj2_75	mc	nf3_20	s5	xor	
ep	hm3	lj2_98	mcp	nf3_25	s7	zlk1	
fls	hm4	lj3_38	mgp	nf3_30	sbt	zlk2a	
fr	hv	lj3_75	ml_10	osp_10	shv1	zlk2b	

Table 2: Bound constrained problems used in the numerical results.

antenna2	bunnag12	gtm	hubfit	s231	s392
avgasa	bunnag13	hatfldh	Ji1	s232	simpllpa
biggsc4	ex2_1_10	himmelbi	Ji2	s250	simpllpb
bunnag1	ex2_1_5	hs024	Ji3	s251	sipow1
bunnag2	ex2_1_7	hs035	ksip	s253	sipow1m
bunnag3	expfita	hs036	lowpass	s268	sipow2
bunnag4	expfitb	hs037	lsqfit	s277	sipow2m
bunnag5	fir_linear	hs044	makela4	s278	sipow3
bunnag6	g01	hs076	nuffield_continuum	s279	sipow4
bunnag7	genocop07	hs086	oet1	s280	stancmin
bunnag8	genocop09	hs118	oet3	s331	structure2
bunnag9	genocop10	hs21mod	pentagon	s340	tfi2
bunnag10	genocop11	hs268	pt	s354	weapons
bunnag11	goffin	hs35mod	s224	s359	zecevic2

Table 3: Linearly constrained problems used in the numerical results.

The profiles for the function evaluations are reported in Figures 2-3 comparing PSwarm, Pattern Search (`PSwarm` solver with an empty search step), the RBF model minimized with the DCA algorithm, and the RBF model minimized with the DCA algorithm and the directions in the poll step sorted by using the RBF model. PSwarm is a stochastic solver but we chose to make only one run for each problem, since we are only interested in an indication of its performance.

As expected, Figures 2-3 show a slight advantage in the number of objective function evaluations for the pattern search version where an empty search step is considered (with more significance for the bound constrained problems).

The last three figures depict profiles for the quality of the final objective function value obtained. We can easily verify that PSwarm wins over the remaining versions. This is due to the PSwarm ability to overcome some non-convexity of the objective function. Recall that PSwarm is a population based algorithm that often uses all the objective function evaluations budget.

We can observe that the RBF-DCA Sort version has an advantage over the remaining version (except for PSwarm). The good performance of this version in all the test problems is attained mainly due to the excellent performance in the linearly constrained subset of test problems. Such performance may be justified by the number and type of directions that are considered in the poll step when linear constraints are present. Such number tends to be larger than the one for the bound constrained case which is constant and equal to $2n + 2$.

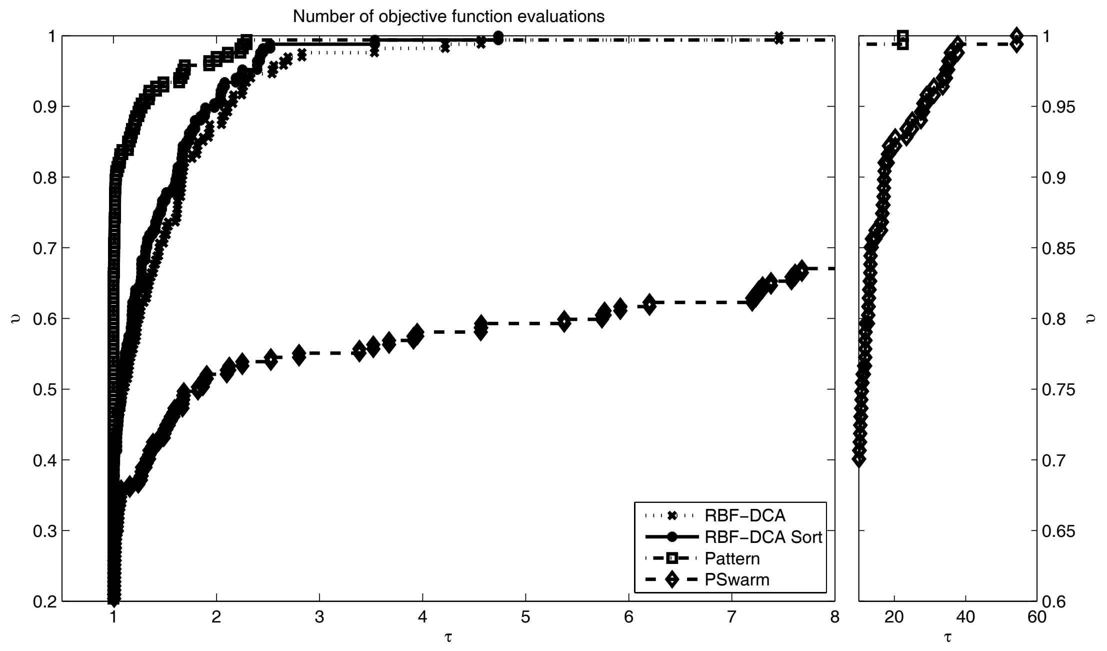

Figure 2: Comparison among RBF-DCA, RBF-DCA Sort, Pattern and PSwarm for all problems.

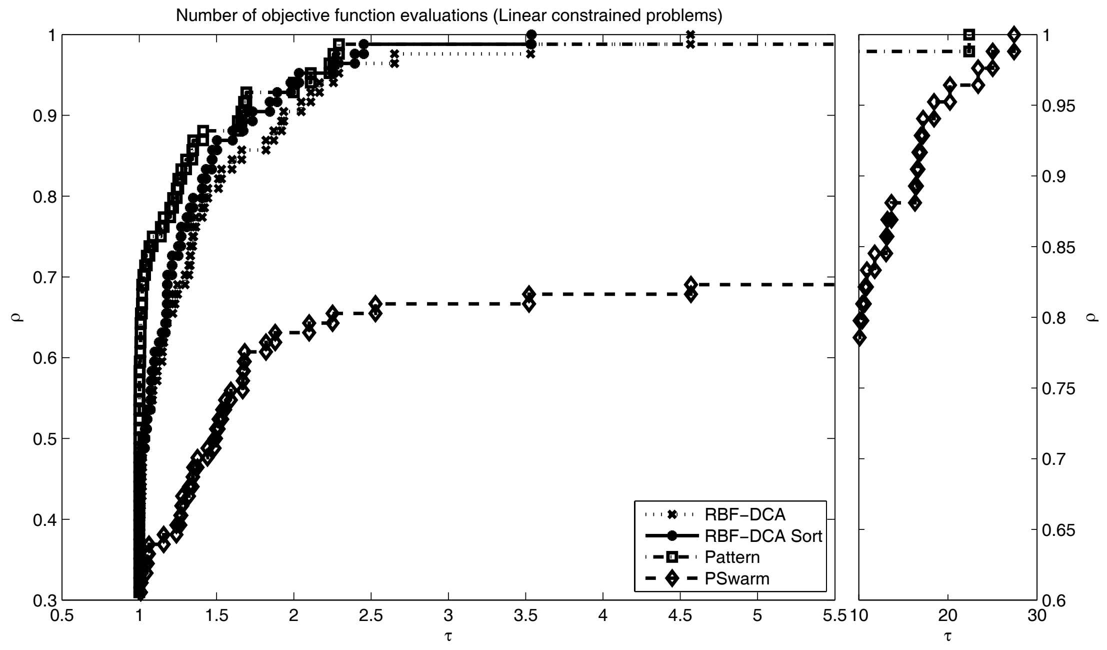

Figure 3: Comparison among RBF-DCA, RBF-DCA Sort, Pattern, and PSwarm for linearly constrained problems.

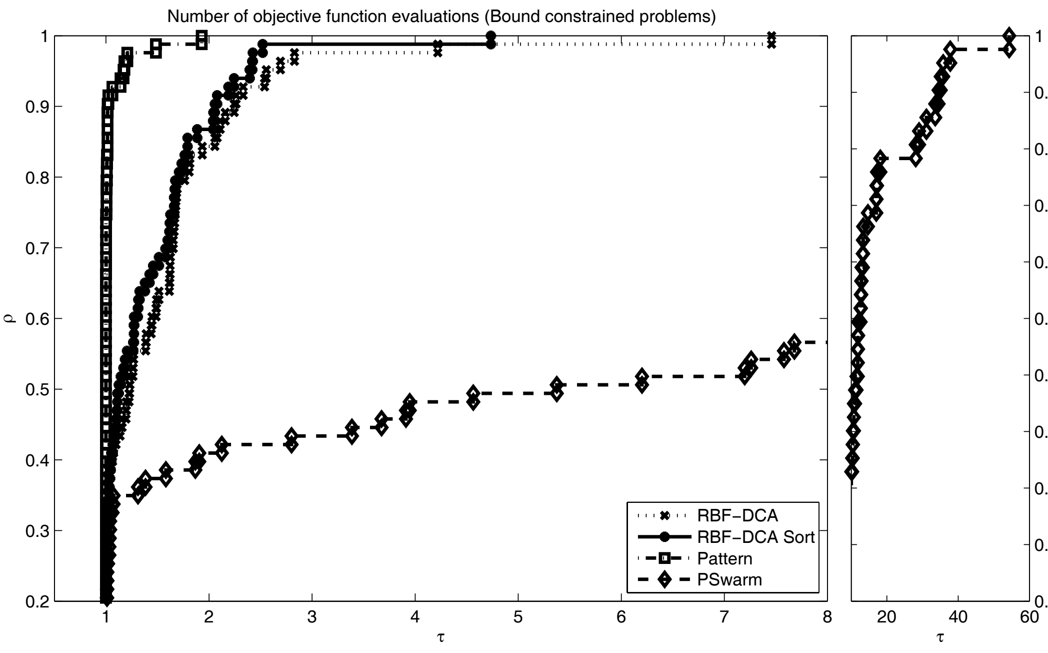

Figure 4: Comparison among RBF-DCA, RBF-DCA Sort, Pattern, and PSwarm for bound constrained problems.

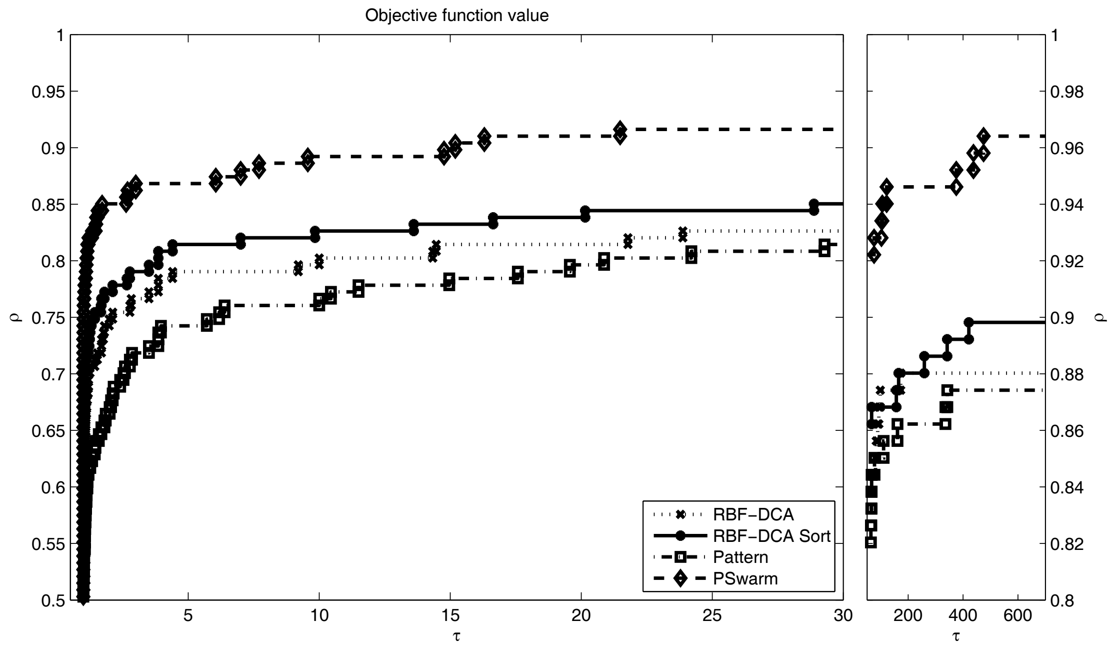

Figure 5: Comparison among RBF-DCA, RBF-DCA Sort, Pattern, and PSwarm for all problems.

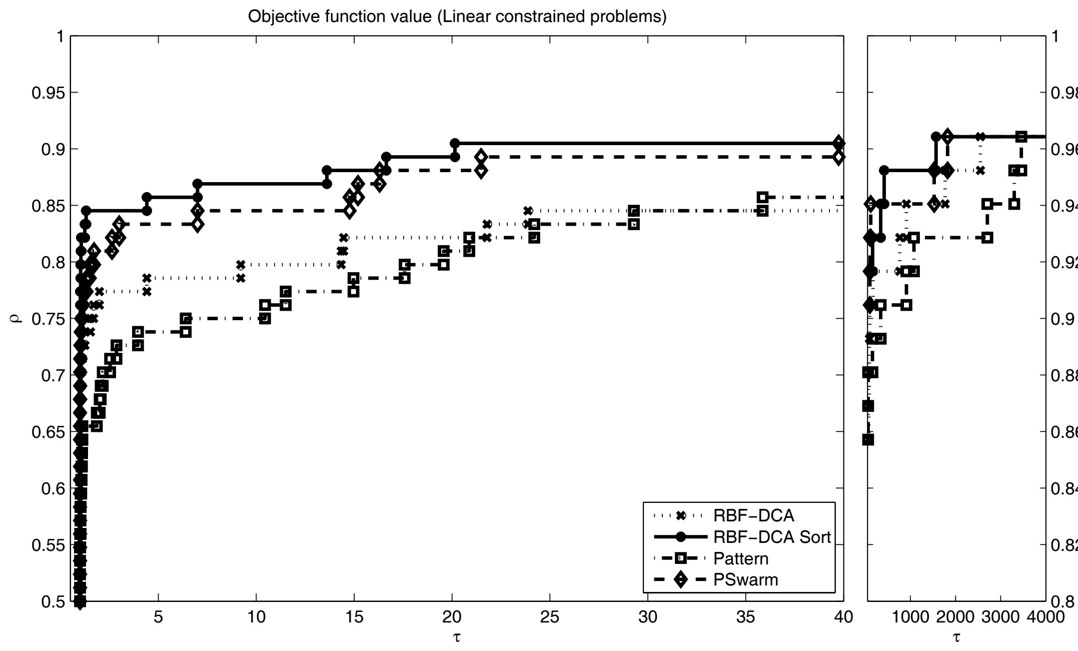

Figure 6: Comparison among RBF-DCA, RBF-DCA Sort, Pattern, and PSwarm for linearly constrained problems.

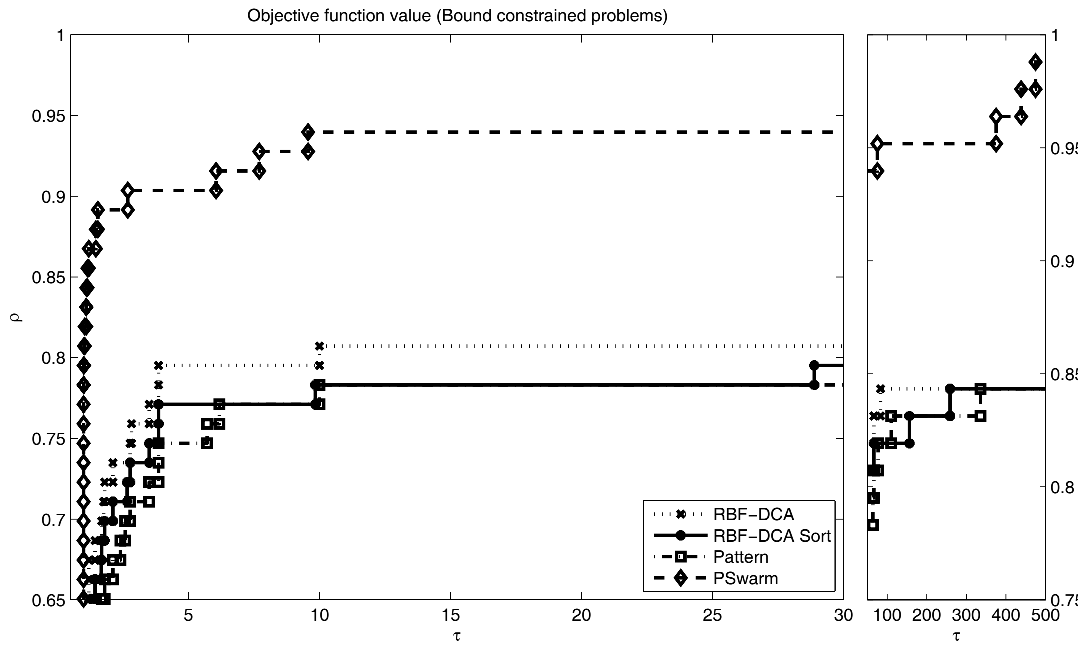

Figure 7: Comparison among RBF-DCA, RBF-DCA Sort, Pattern, and PSwarm for bound constrained problems.

7 Conclusions

In this paper we proposed the use of radial function basis (RBF) models to improve the efficiency of a direct-search type method for the global optimization of functions subject to bound and linear constraints. The RBF models are known to model well multimodal functions and proved to be useful in the context of black-box optimization of functions expensive to evaluate.

A minimizer of the RBF model is used to obtain an incumbent point where the objective function is evaluated. The RBF model minimization problem consists in a bound constrained optimization problem where the objective function (the RBF model) is cheap to evaluate. To solve the minimization problem we proposed to apply an algorithm based on difference of convex (d.c.) functions. The proposed d.c. algorithm (DCA) was compared to the MATLAB `fmincon` solver and proved to be competitive and simultaneously simpler.

Extensive numerical results were reported for a test set of bound and linearly constrained problems in order to access the overall performance of the resulting derivative-free optimization algorithms. The reported results confirmed the utility of the RBF in driving the algorithm for a better objective function value at the expense of only a moderate increase in the number of objective function evaluations.

References

[1] M. M. Ali, C. Khompatraporn, and Z. B. Zabinsky. A numerical evaluation of several stochastic algorithms on selected continuous global optimization test problems. *J. Global Optim.*, 31:635–672, 2005.

[2] Le Thi Hoai An and Pham Dinh Tao. Convex analysis approach to d.c. programming: Theory, algorithms and applications. *Acta Math. Vietnam.*, 22:289–355, 1997.

[3] Le Thi Hoai An and Pham Dinh Tao. D.C. optimization algorithms for solving the trust-region subproblem. *SIAM J. Optim.*, 8:476–505, 1998.

[4] Le Thi Hoai An and Pham Dinh Tao. The DC (difference of convex functions) programming and DCA revisited with DC models of real world nonconvex optimization problems. *Ann. Oper. Res.*, 133:23–46, 2005.

[5] M. Björkman and K. Holmström. Global optimization of costly nonconvex functions using radial basis functions. *Optim. Eng.*, 1:373–397, 2000.

[6] A. R. Conn, K. Scheinberg, and L. N. Vicente. *Introduction to Derivative-Free Optimization.* MPS-SIAM Series on Optimization. SIAM, Philadelphia, 2009.

[7] T. Pham Dinh and S. Elbernoussi. Duality in d.c. (difference of convex functions) optimization. In *Subgradient Methods*, volume 84, pages 276–294. Birkhäuser, Basel, 1988.

[8] E.D. Dolan and J.J. Moré. Benchmarking optimization software with performance profiles. *Math. Program.*, 91:201–213, 2002.

[9] R. Fourer, D. M. Gay, and B. W. Kernighan. A modeling language for mathematical programming. *Management Sci.*, 36:519–554, 1990.

[10] N. I. M. Gould, D. Orban, and Ph. L. Toint. Contrained and unconstrainted test environement, revisited. `http://cuter.rl.ac.uk/cuter-www`.

[11] J. D. Griffin and T. G. Kolda. Asynchronous parallel hybrid optimization combining DIRECT and GSS. *Optim. Methods Softw.*, 2009, to appear.

[12] H.-M. Gutmann. A radial basis function method for global optimization. *J. Global Optim.*, 19:201–227, 2001.

[13] A.-R. Hedar and M. Fukushima. Heuristic pattern search and its hybridization with simulated annealing for nonlinear global optimization. *Optim. Methods Softw.*, 19:291–308, 2004.

[14] L. Ingber and B. Rosen. Genetic algorithms and very fast simulated reannealing: A comparison. *Math. Comput. Modelling*, 16:87–100, 1992.

[15] Y. Ji, K.-C. Zhang, and S.-J. Qu. A deterministic global optimization algorithm. *Appl. Math. Comput.*, 185:382–387, 2006.

[16] D. Jones, C. Perttunen, and B. Stuckman. Lipschitzian optimization without the Lipschitz constant. *J. Optim. Theory Appl.*, 79:157–181, 1993.

[17] J.-E. Käck. Constrained global optimization with radial basis functions. Technical Report Research Report MdH-IMa-2004, Department of Mathematics and Physics, Mälardalen University, Västerås, Sweden, 2004.

[18] E. Kiseleva and T. Stepanchuk. On the efficiency of a global non-differentiable optimization algorithm based on the method of optimal set partitioning. *J. Global Optim.*, 25:209–235, 2003.

[19] T. G. Kolda, R. M. Lewis, and V. Torczon. Optimization by direct search: New perspectives on some classical and modern methods. *SIAM Rev.*, 45:385–482, 2003.

[20] M. Locatelli. A note on the Griewank test function. *J. Global Optim.*, 25:169–174, 2003.

[21] M. Locatelli and F. Schoen. Fast global optimization of difficult Lennard-Jones clusters. *Comput. Optim. Appl.*, 21:55–70, 2002.

[22] MathWorks. MATLAB. `http://www.mathworks.com`.

[23] Z. Michalewicz. Evolutionary computation techniques for nonlinear programming problems. *International Transactions in Operational Research*, 1:223–240, 1994.

[24] Z. Michalewicz. *Genetic Algorithms+ Data Structures= Evolution Programs.* Springer, third edition, 1996.

[25] M. Mongeau, H. Karsenty, V. Rouzé, and J.-B. Hiriart-Urruty. Comparison of public-domain software for black box global optimization. *Optim. Methods Softw.*, 13:203–226, 2000.

[26] NETLIB. GLOBAL library. `http://www.gamsworld.org/global/globallib.htm`.

[27] A. Neumaier. The COCONUT benchmark. `www.mat.univie.ac.at/~neum/glopt/coconut/Benchmark/Benchmark.html`.

[28] R. Oeuvray. *Trust-Region Methods Based on Radial Basis Functions with Application to Biomedical Imaging.* PhD thesis, Institut de Mathématiques, École Polytechnique Fédérale de Lausanne, Switzerland, 2005.

[29] R. Oeuvray and M. Bierlaire. BOOSTERS: A derivative-free algorithm based on radial basis functions. *International Journal of Modelling and Simulation*, 29:4634–4636, 2009.

[30] K. E. Parsopoulos, V. P. Plagianakos, G. D. Magoulas, and M. N. Vrahatis. Stretching technique for obtaining global minimizers through particle swarm optimization. In *Proc. Of the Particle Swarm Optimization Workshop*, pages 22–29, Indianapolis, USA, 2001.

[31] R. G. Regis and C. A. Shoemaker. Constrained global optimization of expensive black box functions using radial basis functions. *J. Global Optim.*, 31:153–171, 2005.

[32] R. G. Regis and C. A. Shoemaker. Improved strategies for radial basis function methods for global optimization. *J. Global Optim.*, 37:113–135, 2007.

[33] R. G. Regis and C. A. Shoemaker. Parallel radial basis function methods for the global optimization of expensive functions. *European J. Oper. Res.*, 182:514–535, 2007.

[34] R. G. Regis and C. A. Shoemaker. A stochastic radial basis function method for the global optimization of expensive functions. *INFORMS J. Comput.*, 19:497–509, 2007.

[35] T.P. Runarsson and X. Yao. Stochastic ranking for constrained evolutionary optimization. *IEEE Transactions on Evolutionary Computation*, 4:284–294, 2000.

[36] R. Vanderbei. Nonlinear optimization models. `www.sor.princeton.edu/~rvdb/ampl/nlmodels/index.html`.

[37] A. Ismael F. Vaz and L. N. Vicente. A particle swarm pattern search method for bound constrained global optimization. *J. Global Optim.*, 39:197–219, 2007.

[38] A. Ismael F. Vaz and L. N. Vicente. Pswarm: A hybrid solver for linearly constrained global derivative-free optimization. *Optim. Methods Softw.*, 24:669–685, 2009.

[39] S. M. Wild, R. G. Regis, and C. A. Shoemaker. ORBIT: Optimization by radial basis function interpolation in trust-regions. *SIAM J. Sci. Comput.*, 30:3197–3219, 2008.