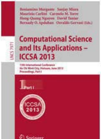

Beniamino Murgante Sanjay Misra
Maurizio Carlini Carmelo M. Torre
Hong-Quang Nguyen David Taniar
Bernady O. Apduhan Osvaldo Gervasi (Eds.)

LNCS 7971

Computational Science and Its Applications – ICCSA 2013

13th International Conference
Ho Chi Minh City, Vietnam, June 2013
Proceedings, Part I

Part I

ICCSA 2013

Springer

Multilocal Programming: a Derivative-Free Filter Multistart Algorithm

Florbela P. Fernandes[1,3], M. Fernanda P. Costa[2,3], and Edite M.G.P. Fernandes[4]

[1] Polytechnic Institute of Bragança, ESTiG, 5301-857 Bragança, Portugal,
fflor@ipb.pt
[2] Department of Mathematics and Applications, University of Minho, 4800-058 Guimarães, Portugal,
mfc@math.uminho.pt
[3] Centre of Mathematics,
[4] Algoritmi R&D Centre,
University of Minho, 4710-057 Braga, Portugal
emgpf@dps.uminho.pt

Abstract. Multilocal programming aims to locate all the local solutions of an optimization problem. A stochastic method based on a multistart strategy and a derivative-free filter local search for solving general constrained optimization problems is presented. The filter methodology is integrated into a coordinate search paradigm in order to generate a set of trial approximations that might be acceptable if they improve the constraint violation or the objective function value relative to the current one. Preliminary numerical experiments with a benchmark set of problems show the effectiveness of the proposed method.

Keywords: Multilocal programming; multistart; derivative-free; coordinate search; filter method

1 Introduction

Multilocal programming has a wide range of applications in the engineering field [8,10,18,19] and aims to compute all the global and local/non-global solutions of constrained nonlinear optimization problems. The goal of most multistart methods presented in the literature is to locate multiple solutions of bound constrained optimization problems [1,21,23,24] (see also [15] and the references therein included). Multistart may also be used to explore the search space and converge to a global solution of nonlinear optimization problems [6]. When a multistart strategy is implemented, a local search procedure is applied to randomly generated (sampled) points of the search space aiming to converge to the multiple solutions of the problem. However, the same solutions may be found over and over again. To avoid convergence to an already computed solution, some multistart methods use clustering techniques to define prohibited regions based on the closeness to the previously located solutions. Sampled points from these prohibited regions are discarded since the local search procedure would converge most certainly to an already located solution. MinFinder is an example of a clustering algorithm that competes with multistart when global and some local minimizers are required [21,22]. Alternatively, niching, deflecting and stretching techniques may be combined with global optimization methods, like the simulated annealing, evolutionary algorithm and the particle swarm optimization, to discover the global and some specific local minimizers of a problem [17,18,20]. A glowworm swarm optimization approach has been proposed to converge to multiple optima of multimodal functions [13].

The purpose of this paper is to present a method based on a multistart technique and a derivative-free deterministic local search procedure to obtain multiple solutions of an optimization problem. The novelty here is that a direct search method and the filter methodology, as outlined in [3,7], are combined to construct a local search procedure that does not require any derivative information. The filter methodology is implemented to handle the constraints by forcing the local search towards the feasible region. Bound, as well as linear and nonlinear inequality and equality constraints may be treated by the proposed local search procedure.

Direct search methods are popular because they are straightforward to implement and do not use or approximate derivatives. Like the gradient-based methods, direct search methods also have their niche. For example, the maturation of simulation-based optimization has led to optimization problems in which derivative-free methods are mandatory. There are also optimization problems where derivative-based methods cannot be used since the objective function is not numerical in nature [11].

The problem to be addressed is of the following type:

$$\begin{array}{rl} \min & f(x) \\ \text{subject to} & g_j(x) \leq 0, \quad j = 1, ..., m \\ & l_i \leq x_i \leq u_i, \, i = 1, ..., n \end{array} \tag{1}$$

where, at least one of the functions $f, g_j : \mathbb{R}^n \longrightarrow \mathbb{R}$ is nonlinear and $\mathrm{F} = \{x \in \mathbb{R}^n : g(x) \leq 0\,,\ l \leq x \leq u\}$ is the feasible region. Problems with general equality constraints can be reformulated in the above form by introducing $h(x) = 0$ as an inequality constraint $|h(x)| - \tau \leq 0$, where τ is a small positive relaxation parameter. This kind of problems may have many global and local optimal solutions and so, it is important to develop a methodology that is able to explore the entire search space and find all the minimizers guaranteeing, in some way, that convergence to a previously found minimizer is avoided.

This paper is organized as follows. In Section 2, the algorithm based on the multistart strategy and on the filter methodology is presented. In Section 3, we report the results of our numerical experiments with a set of benchmark problems. In the last section, conclusions are summarized and recommendations for future work are given.

2 Multistart Coordinate Search Filter Method

The methodology used to compute all the optimal solutions of problem (1), hereafter called MCSFilter method, is a multistart algorithm coupled with a clustering technique to avoid the convergence to previously detected solutions. The exploration feature of the method is carried out by a multistart strategy that aims at generating points randomly spread all over the search space. Exploitation of promising regions is made by a simple local search approach. In contrast to the line search BFGS method presented in [24], the local search proposal, a crucial procedure inside a multistart paradigm, relies on a direct search method, known as coordinate search (CS) method [11], that does not use any analytical or numerical derivative information.

Since the goal of the local search is to converge to a solution of a constrained optimization problem, started from a sampled approximation, progress towards an optimal solution is measured by a filter set methodology, as outlined in [7], which is integrated into the local search procedure. The filter methodology appears naturally from the observation that an optimal solution of the problem (1) minimizes both constraint violation and objective function [3,4,7,9]. Thus, the proposed CS method is combined with a (line search) filter method that aims at generating trial iterates that might be acceptable if they improve the constraint violation or the objective function relative to the current iterate.

2.1 The Multistart Strategy

Multistart is a stochastic algorithm that repeatedly applies a local search to sampled points (randomly generated inside $[l, u]$) $x_i = l_i + \lambda(u_i - l_i)$ for $i = 1, \ldots, n$, where λ is a random number uniformly distributed in $[0, 1]$, aiming to converge to the solutions of a multimodal problem. When a multistart strategy is applied to converge to the multiple solutions, some or all of the minimizers may be found over and over again. To avoid convergence to a previously computed solution, a clustering technique based on computing the regions of attraction of previously identified minimizers is to be integrated in the algorithm. The region of attraction of a local minimizer, y_i, associated with a local search procedure $\mathbf{L}$, is defined as:

$$A_i \equiv \{x \in [l, u] : \mathbf{L}(x) = y_i\}, \tag{2}$$

where $\mathbf{L}(x)$ is the minimizer obtained when the local search procedure $\mathbf{L}$ starts at point x. The ultimate goal of a multistart algorithm is to invoke the local search procedure N times, where N is the number of solutions of (1). If a sampled point $x \in [l, u]$ belongs to a region of attraction A_j then the minimizer y_j would be obtained when $\mathbf{L}$ is started from x. Ideally, the local search procedure is to be applied only to a sampled point that does not belong to any of the regions of attraction of already computed minimizers, or equivalently to the union of those regions of attraction, since they do not overlap. However, computing the region of attraction A_i of a minimizer y_i is not an easy task. Alternatively, a stochastic procedure may be used to estimate the probability, p, that a sampled point will not belong to the union of the regions of attraction of already computed minimizers, i.e.,

$$p = Prob[x \notin \cup_{i=1}^{k} A_i] = \prod_{i=1}^{k} Prob[x \notin A_i] \approx Prob[x \notin A_n]$$

where A_n is the region of attraction of the nearest to x minimizer y_n (see details in [24]). The value of p may be approximated using $Prob[x \notin B(y_n, R_n)]$, where $B(y, R)$ represents a sphere centered at y with radius R.

Let the maximum attractive radius of the minimizer y_i be defined by:

$$R_i = \max_j \left\{ \left\| x_i^{(j)} - y_i \right\| \right\}, \tag{3}$$

where $x_i^{(j)}$ is one of the sampled points that led to the minimizer y_i. Given x, let $d_i = \|x - y_i\|$ be the distance of x to y_i. If $d_i < R_i$ then x is likely to be inside the region of attraction of y_i. However, if the direction from x to y_i is ascent then x is likely to be outside the region of attraction of y_i and the local search procedure is to be implemented started from x, since a new minimum could be computed with high probability. Thus, using these arguments, similarly to [24], the probability that $x \notin A_i$ is herein estimated by:

$$Prob(x \notin A_i) = \begin{cases} 1, & \text{if } z \geq 1 \text{ or the direction from } x \text{ to } y_i \text{ is ascent} \\ \varrho\, \phi(z, r), & \text{otherwise} \end{cases} \tag{4}$$

where $\varrho \in [0, 1]$ is a fixed parameter, $z = d_i/R_i \in (0, 1)$, r is the number of times y_i has been recovered so far and $\phi(z, r) \in (0, 1)$ is taken as

$$\phi(z, r) = z \exp(-r^2(z-1)^2), \text{ where } \begin{array}{l} \lim_{z \to 0} \phi(z, r) \to 0,\ \lim_{z \to 1} \phi(z, r) \to 1, \\ \lim_{r \to \infty} \phi(z, r) \to 0. \end{array}$$

In this derivative-free approach, the direction from x to y_i is considered ascent when $f(x + \beta(y_i - x)) - f(x) > 0$ for a small $\beta > 0$. In the gradient-based approach [24], ϱ is a function that depends on the directional derivative of f along the direction from x to y_i.

Figure 1 illustrates the behavior of the multistart method when converging to four minimizers. The point represented by a $\star$ (in black) is the first sampled point that converges to a minimizer. The points represented by $\diamond$ (in full blue) lie outside the region of attraction and thus the local search procedure is applied to be able to converge to a minimizer. For example, the global minimizer (leftmost and bottom of the figure) has been recovered seven times. The other points (represented by $\circ$) are sampled points inside the region of attraction of a minimizer. They have been discarded, i.e., the local search has not been applied to them.

Algorithm 1 shows the multistart algorithm. Although this type of methods is simple, they would not be effective if a bad stopping rule is used. The main goal of a stopping rule is to make the algorithm to stop when all minimizers have

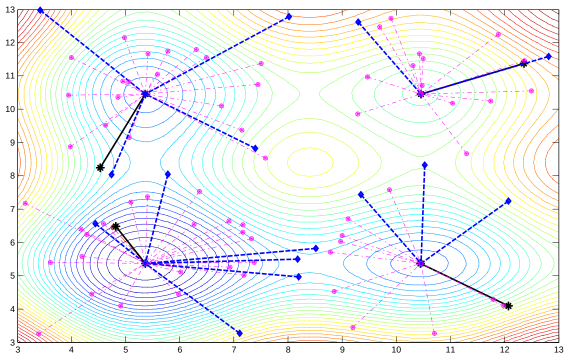

Fig. 1. Illustration of the multistart method

been located with certainty. Further, it should not require a large number of local searches to decide that all minimizers have been found (see [14]). A simple stopping rule uses the estimate of the fraction of uncovered space $P(k) = \frac{k(k+1)}{t(t-1)}$, where k is the number of recovered minimizers after having performed t local search procedures. The multistart algorithm then stops if $P(k) \leq \epsilon$, for a small $\epsilon > 0$.

2.2 The Coordinate Search Filter Procedure

The coordinate search filter (CSFilter) method, combining a derivative-free paradigm with the filter methodology, is proposed as the local search procedure **L**, to compute a minimizer y starting from a sampled point $x \in [l, u]$. Briefly, a minimizer y of the constrained optimization problem (1) is to be computed, starting from x. The basic idea behind this approach is to interpret (1) as a bi-objective optimization problem aiming to minimize both the objective function $f(x)$ and a nonnegative continuous aggregate constraint violation function $\theta(x)$ defined by

$$\theta(x) = \|g(x)_+\|^2 + \|(l - x)_+\|^2 + \|(x - u)_+\|^2 \tag{5}$$

where $v_+ = \max\{0, v\}$. Therefore, the proposed CSFilter approach computes an approximate minimizer, y, to the bi-objective optimization problem

$$\min_x \left(\theta(x), f(x)\right). \tag{6}$$

The filter technique incorporates the concept of nondominance, present in the field of multi-objective optimization, to build a filter that is able to accept trial approximations if they improve the constraint violation or objective function value. A filter $\mathcal{F}$ is a finite set of points y, corresponding to pairs $(\theta(y), f(y))$, none of which is dominated by any of the others. A point y is said to dominate a point y' if only if $\theta(y) \leq \theta(y')$ and $f(y) \leq f(y')$.

Algorithm 1 Multistart algorithm

```
Require: Parameter values; set $Y^* = \emptyset$[†], $k = 1$, $t = 1$;
 1: Randomly generate $x \in [l, u]$; compute $A_{\min} = \min_{i=1,\dots,n}\{u_i - l_i\}$;
 2: Compute $y_1 = \mathbf{L}(x)$, $R_1 = \|x - y_1\|$; set $r_1 = 1$, $Y^* = Y^* \cup y_1$;
 3: repeat
 4:   Randomly generate $x \in [l, u]$;
 5:   Set $o = \arg\min_{j=1,\dots,k} d_j \equiv \|x - y_j\|$;
 6:   if $d_o < R_o$ then
 7:     if the direction from $x$ to $y_o$ is ascent then
 8:       Set $p = 1$;
 9:     else
10:       Compute $p = \varrho\,\phi(\frac{d_o}{R_o}, r_o)$;
11:     end if
12:   else
13:     Set $p = 1$;
14:   end if
15:   if $\zeta$[‡] $< p$ then
16:     Compute $y = \mathbf{L}(x)$; set $t = t + 1$;
17:     if $\|y - y_j\| > \gamma^* A_{\min}$, for all $j = 1, \dots, k$[§] then
18:       Set $k = k + 1$, $y_k = y$, $r_k = 1$, $Y^* = Y^* \cup y_k$; compute $R_k = \|x - y_k\|$;
19:     else
20:       Set $R_l = \max\{R_l, \|x - y_l\|\}$[♮]; $r_l = r_l + 1$;
21:     end if
22:   else
23:     Set $R_o = \max\{R_o, \|x - y_o\|\}$; $r_o = r_o + 1$;
24:   end if
25: until the stopping rule is satisfied
```

† - Y^* is the set containing the computed minimizers.

‡ - ζ is a uniformly distributed number in $(0, 1)$.

§ - $y \notin Y^*$.

♮ - $\|y - y_l\| \leq \gamma^* A_{\min}$.

A rough outline of a coordinate search filter is as follows. At the beginning of the optimization, the filter is initialized to $\mathcal{F} = \{(\theta, f) : \theta \geq \theta_{\max}\}$, where $\theta_{\max} > 0$ is an upper bound on the acceptable constraint violation.

Let $\mathcal{D}_\oplus$ denote de set of $2n$ coordinate directions, defined as the positive and negative unit coordinate vectors, $\mathcal{D}_\oplus = \{e_1, e_2, \dots, e_n, -e_1, -e_2, \dots, -e_n\}$. The search begins with a central point, the current approximation $\tilde{x}$, as well as $2n$ trial approximations $y_c^i = \tilde{x} + \alpha d_i$, for $d_i \in \mathcal{D}_\oplus$, where $\alpha > 0$ is a step size. The constraint violation value and the objective function value of all $2n + 1$ points are computed. If some trial approximations improve over $\tilde{x}$, reducing θ or f by a certain amount (see (7) and (8) below), and are acceptable by the filter, then the best of these non-dominated trial approximations, y_c^{best}, is selected, and the filter is updated (adding the corresponding entries to the filter and removing any dominated entries). Then, this best approximation becomes the new central point in the next iteration, $\tilde{x} \leftarrow y_c^{best}$. If, on the other hand, all trial approximations y_c^i are dominated by the current filter, then all y_c^i are rejected, and a restoration phase is invoked.

To avoid the acceptance of a point y_c^i, or the corresponding pair $\left(\theta(y_c^i), f(y_c^i)\right)$, that is arbitrary close to the boundary of $\mathcal{F}$, the trial y_c^i is considered to improve over $\tilde{x}$ if one of the conditions

$$\theta(y_c^i) \leq (1-\gamma_\theta)\ \theta(\tilde{x}) \ \text{ or } \ f(y_c^i) \leq f(\tilde{x}) - \gamma_f\, \theta(\tilde{x}) \tag{7}$$

holds, for fixed constants $\gamma_\theta, \gamma_f \in (0,1)$.

However, the filter alone cannot ensure convergence to optimal points. For example, if a sequence of trial points satisfies $\theta(y_c^i) \leq (1-\gamma_\theta)\ \theta(\tilde{x})$ then it could converge to an arbitrary feasible point. Therefore, when $\tilde{x}$ is nearly feasible, $\theta(\tilde{x}) \leq \theta_{\min}$ for a small positive $\theta_{\min}$, the trial approximation y_c^i has to satisfy only the condition

$$f(y_c^i) \leq f(\tilde{x}) - \gamma_f\, \theta(\tilde{x}) \tag{8}$$

instead of (7), in order to be acceptable.

The best non-dominated trial approximation is selected as follows. The best point y_c^{best} of a set $Y = \{y_c^i : y_c^i = \tilde{x} + \alpha d_i, d_i \in \mathcal{D}_\oplus\}$ is the point that satisfies one of two following conditions:

– if there are some feasible points in Y, y_c^{best} is the point that has the less objective function value among the feasible points:

$$\theta\left(y_c^{best}\right) = 0 \text{ and } f\left(y_c^{best}\right) < f\left(y_c^i\right) \ \text{ for all } y_c^i \in Y \text{ such that } \theta\left(y_c^i\right) = 0; \tag{9}$$

– otherwise, y_c^{best} is the point that has less constraint violation among the non-dominated infeasible points

$$0 < \theta\left(y_c^{best}\right) < \theta\left(y_c^i\right) \ \text{ and } y_c^i \notin \mathcal{F}. \tag{10}$$

We remark that the filter is updated whenever the trial approximations y_c^i verify conditions (7) or (8) and are non-dominated.

When it is not possible to find a non-dominated best trial approximation, and before declaring the iteration unsuccessful, a restoration phase is invoked. In this phase, the most nearly feasible point in the filter, $x_{\mathcal{F}}^{inf}$, is recuperated and the search along the $2n$ coordinate directions is carried out about it to find the set $Y = \{y_c^i : y_c^i = x_{\mathcal{F}}^{inf} + \alpha d_i, d_i \in \mathcal{D}_\oplus\}$. If a non-dominated best trial approximation is found, this point becomes the central point of the next iteration and the iteration is successful. Otherwise, the iteration is unsuccessful, the search returns back to the current $\tilde{x}$, the step size is reduced, for instance $\alpha = \alpha/2$, and new $2n$ trial approximations y_c^i are generated about it. If a best non-dominated trial approximation is still not found, the step size reduction is repeated since another unsuccessful iteration has occurred. When α falls below $\alpha_{\min}$, a small positive tolerance, the search terminates since first-order convergence has been attained [11]. At each unsuccessful iteration, the CSFilter algorithm reduces the step size and tries again the coordinate search about the current point $\tilde{x}$.

Thus, to judge the success of the CSFilter algorithm, the below presented conditions are applied

$$\begin{cases} \alpha \leq \alpha_{\min} \\ \theta(y_c^{best}) < 0.01\,\theta_{\min} \\ \left|f(y_c^{best}) - f(y)\right| < 10^{-6}\,|f(y)| + 10^{-8}, \end{cases} \tag{11}$$

where $0 < \alpha_{\min} << 1$ and y is the previous current point. The proposed algorithm for the local procedure is presented in Algorithm 2.

Algorithm 2 CSFilter algorithm

```
Require: $x$ (sampled in the Multistart algorithm) and parameter values; set $\tilde{x} = x$,
    $x_{\mathcal{F}}^{inf} = x$, $y = \tilde{x}$;
 1: Initialize the filter;
 2: Set $\alpha = \min\{1, 0.05\frac{\sum_{i=1}^{n} u_i - l_i}{n}\}$;
 3: repeat
 4:   Compute the trial approximations $y_c^i = \tilde{x} + \alpha d_i$, for all $d_i \in \mathcal{D}_{\oplus}$;
 5:   repeat
 6:     Check acceptability of trial points $y_c^i$ using (7) and (8);
 7:     if there are some $y_c^i$ acceptable by the filter then
 8:       Update the filter;
 9:       Choose $y_c^{best}$ using (9) or (10);
10:       Set $y = \tilde{x}$, $\tilde{x} = y_c^{best}$; update $x_{\mathcal{F}}^{inf}$ if appropriate;
11:     else
12:       Compute the trial approximations $y_c^i = x_{\mathcal{F}}^{inf} + \alpha d_i$, for all $d_i \in \mathcal{D}_{\oplus}$;
13:       Check acceptability of trial points $y_c^i$ using (7) and (8);
14:       if there are some $y_c^i$ acceptable by the filter then
15:         Update the filter;
16:         Choose $y_c^{best}$ using (9) or (10);
17:         Set $y = \tilde{x}$, $\tilde{x} = y_c^{best}$; update $x_{\mathcal{F}}^{inf}$ if appropriate;
18:       else
19:         Set $\alpha = \alpha/2$;
20:       end if
21:     end if
22:   until new trial $y_c^{best}$ is acceptable
23: until the conditions (11) are satisfied
```

3 Numerical Results

To analyze the performance of the MCSFilter algorithm, a set of 30 test problems is used (see Table 1). The set contains bound constrained problems, inequality and equality constrained problems, multimodal objective functions, with one global and some local, more than one global, and a unimodal optimization problem. Table 1 reports the acronym of the tested problems, under ‘Prob.’, references with details of the models and the known number of solutions, ‘Min’. Five minimization problems, 2Dt+1, 2Dt+2, MMO+1, CB6+1 and BR+1 are defined from well-known problems by adding constraints: 2Dt+1 comes from 2Dt by adding the constraint $(x_1+5)^2+(x_2-5)^2-100 \le 0$; 2Dt+2 is 2Dt+1 with an additional linear constraint $-x_1-x_2-3 \le 0$; MMO+1 comes from MMO by adding the linear constraint $-2x_1-3x_2+27 \le 0$; CB6+1 comes from CB6 with the constraint $(x_1+1)^2+(x_2-1)^2 \le 2.25$; BR+1 comes from BR with the additional constraint $(x_1-5)^2+2(x_2-10)^2 \le 100$. Problem g8 is a maximization problem that was rewritten as a minimization problem. g11 has an equality constraint which was transformed into an inequality constraint using $\tau = 10^{-5}$.

Table 1. Problem, reference and known number of solutions.

Prob.		Min	Prob.		Min	Prob.		Min
ADJ	[8]	3	SHK10	[2,14,16]	10	MMO+1	[5]	4
CB6	[2,8,14]	6	2Dt	[14,18]	4	CB6+1		4
BR	[2,14,18]	3	3Dt	[14,18]	8	BR+1		3
GP	[2,8,14]	4	4Dt	[14,18]	16	g8	[9,12,18]	2
H3	[2,8,14]	3	5Dt	[14,18]	32	g9	[9,18]	1
H6	[2,8,14]	2	6Dt	[14,18]	64	g11	[9]	2
MMO	[18]	4	8Dt	[14,18]	256	EX. 2.2	[10]	2
SBT	[16]	760	10Dt	[14,18]	1024	EX. 3.3	[10]	2
SHK5	[14,16]	5	2Dt+1	[5]	4	EX. 6.17	[10]	4
SHK7	[14,16]	7	2Dt+2	[5]	5	EX. 1	[19]	2

The MCSFilter method was coded in MatLab and the results were obtained in a PC with an Intel Core i7-2600 CPU (3.4GHz) and 8 GB of memory. In the CSFilter method, we set after an empirical study: $\gamma_\theta = \gamma_f = 10^{-5}$, $\alpha_{\min} = 10^{-5}$, $\theta_{\min} = 10^{-3}$, $\theta_{\max} = 10^3 \max\{1, 1.25\theta(x_{in})\}$, where x_{in} is the initial point in the local search. We also set $\varrho = 0.5$, $\beta = 0.001$, $\gamma^* = 0.1$ and $\epsilon = 0.1$ in the stopping rule of the multistart algorithm. Each problem was solved 10 times and the average values are reported.

Table 2 contains the results obtained when solving the bound constrained problems, where the columns show:

– the average number of computed minimizers, 'Min$_{av}$';
– the average number of function evaluations, 'nfe$_{av}$';
– the average time (in seconds) 'T$_{av}$'.

For comparative purposes, Table 2 also reports:

(i) in columns 5–7, the results presented in [18], relative to the problems BR and nDt with $n = 2, 4, 6, 8, 10$;
(ii) in columns 8–9, the results presented in [14], relative to the problems CB6, BR, GP, H3, H6, SHK5, SHK7, SHK10 and nDt with $n = 4, 5, 6$.

Table 2. Numerical results obtained with bound constrained problems.

	MCSFilter algorithm			results in [18]			results in [14]	
Prob. (n)	Min$_{av}$	nfe$_{av}$	T$_{av}$	Min$_{av}$	nfe$_{av}$	T$_{av}$	Min$_{av}$	nfe$_{av}$
ADJ (2)	2.5	5768	27.970	-	-	-	-	-
CB6 (2)	6†	1869.1	0.262	-	-	-	6	5642
BR (2)	3†	1571.1	0.221	3	2442	0.45	3	2173
GP (2)	4†	13374.9	2.021	-	-	-	4	5906
H3 (3)	2.9	2104.3	0.313	-	-	-	3	3348
H6 (6)	2†	6559.2	0.840	-	-	-	2	3919
MMO (2)	4†	1328.3	0.200	-	-	-	-	-
SBT (2)	25.2	9276.2	5.738	-	-	-	-	-
SHK5 (4)	4.6	6240.3	0.782	-	-	-	5	8720
SHK7 (4)	6.4	8335.2	1.036	-	-	-	7	11742
SHK10 (4)	8.6	10312.6	1.293	-	-	-	10	16020
2Dt (2)	4†	1372.6	0.193	2	1067	0.17	-	-
3Dt (3)	8†	3984.4	0.521	-	-	-	-	-
4Dt (4)	16†	11718.5	1.471	2	3159	0.29	16	17373
5Dt (5)	31.9	32881.7	4.373	-	-	-	32	37639
6Dt (6)	63.8	102490.3	16.105	2	10900	0.75	64	81893
8Dt (8)	254.3	659571.6	368.674	1	36326	2.28	-	-
10Dt (10)	1016	3863756	18563	1	58838	3.71	-	-

† - All the minimizers were computed in all runs.

- Not available.

The algorithm presented in [18] implements a function stretching technique combined with a simulated annealing approach, known as SSA method. We observe that the MCSFilter algorithm has a good performance and is able to find almost all minimizers of the problems with acceptable number of function evaluations. The algorithm finds all minimizers in all runs when solving eight of the 18 tested bound constrained problems and an average of 85% of the minimizers in the remaining ones (94% when the problem SBT is excluded from these statistics). In terms of time needed to find all the solutions, the worst cases are observed with the problems 8Dt and 10Dt. An average of 1.45 seconds and 2593.7 function evaluations are required to compute each solution of problem 8Dt, and an average of 18.3 seconds, with an average of 3802.9 function evaluations, to compute each solution in 10Dt. We remark that the global minimum has been always identified in all runs and in all problems of Table 2. We observe from the comparison with the results of [14] that MCSFilter is slightly more efficient although the multistart method implemented in [14] seems to retrieve a greater number of minimizers.

When a comparison is made with the number of solutions reported in [18], we find that for problems 2Dt and 4Dt, our method finds all the minimizers while SSA identifies only two in each problem. The average number of function evaluations and time to locate each minimizer required by the MCSFilter algorithm for problems 6Dt and 8Dt are smaller than those of the SSA method. Furthermore, MCSFilter finds almost all the minimizers, while SSA finds only two and one respectively. For problem BR, both methods obtained the same number of minimizers, although MCSFilter requires a smaller number of function evaluations and time than SSA. Results for the problem MMO, with 50 and 100 variables are presented in [18], with 4 and 6 found minimizers respectively. Their results were obtained after 1000000 function evaluations. For comparative purposes, we implemented this condition to stop the MCSFilter algorithm and obtain the following results: Min_{av}=48, when $n = 50$, and Min_{av}=14, when $n = 100$. In both cases MCSFilter method finds more minimizers than SSA. A total of 100000 function evaluations were used by SSA (in [18]) to find one minimizer of problem 10Dt. Using the same condition to stop MCSFilter, we obtain Min_{av}=85.3. We note that the problems of the set nDt have 2^n minimizers, where n is the number of variables. We may conclude that the MCSFilter method consistently finds more minimizers.

Table 3. Results obtained with inequality and equality constrained problems.

	MCSFilter algorithm			other results			
Prob. (n, m)	Min_{av}	nfe_{av}	T_{av}		Min_{av}	nfe_{av}	T_{av}
2Dt+1 (2,1)	3.9	10127.6	7.986	[5]	3.8	9752.5	51.6
2Dt+2 (2,2)	4.6	28065.4	24.242	[5]	3.1	13417.4	69.9
MMO+1 (2,1)	$4^{\dagger}$	1858.5	0.527	[5]	4	7630.1	8.8
CB6+1 (2,1)	3.4	12319.1	40.626				
BR+1 (2,1)	$3^{\dagger}$	4128.9	2.525				
g8 (2,2)	1	1930	3.087	[6]	$1^{\S}$	4999	-
				[9]	$1^{\S}$	56476	-
				[18]	5	67753	-
g9 (7,4)	$1.4^{\ddagger}$	5767.7	0.737	[6]	$1^{\S}$	38099	-
				[9]	$1^{\S}$	324569	-
				[18]	1	183806	-
g11 (2,1)	2	84983.3	191.629	[6]	$1^{\S}$	139622	-
				[9]	$1^{\S}$	23722	-
EX. 2.2 (2,2)	$2^{\dagger}$	2469.2	2.315				
EX. 3.3 (2,1)	1.7	6435.9	9.403				
EX. 6.17 (2,3)	3.7	53292.5	141.114				
EX. 1 (2,1)	$2.1^{\ddagger}$	65133.0	593.220				

† - All the minimizers were computed in all runs.

‡ Some cases of premature convergence have been observed, thus identifying a new minimizer.

§ Only one global minimizer was required to be found.

- Not available.

Table 3 lists the results obtained with inequality and equality constrained problems. A comparison is made with the results reported:

(i) in [5,6], where a multistart method coupled with a stochastic approach to derive approximate descent directions and a filter technique is used;
(ii) in [9], which implements a filter simulated annealing method;
(iii) in [18], which implements the SSA method with a penalty function technique.

We may conclude that the presented MCSFilter has a good performance since the average number of function evaluations required to locate each minimizer is smaller than those of the other methods in comparison. We also observe that the algorithm finds all minimizers in all runs when solving MMO+1, BR+1 and EX. 2.2 and an average of 94% of the minimizers in the remaining nine problems. Thus, we may conclude that MCSFilter is able to consistently find almost all minimizers within a reduced time.

4 Conclusions

We present a multistart algorithm based on a derivative-free filter method to solve multilocal programming problems. The method is based on a multistart strategy which relies on the concept of regions of attraction in order to avoid convergence to previously found local minimizers. The proposal for the local procedure is a coordinate search combined with a filter method to generate a sequence of approximate solutions that improve either the constraint violation or the objective function relative to the previous approximation. A set of benchmark problems was used to test the algorithm and the results are very promising. One problematic issue of the proposed MCSFilter method is related to the large number of search directions in the set $\mathcal{D}_{\oplus}$. For large dimensional problems, the computational effort in terms of number of function evaluations and consequently CPU time greatly increases with n. We have observed that the proposed method consistently locates all or almost all minimizers of a problem. To improve the effectiveness of the algorithm, a stopping rule that balances the number of estimated minimizers with local search calls is to be devised in the future.

Acknowledgments. This work was financed by FEDER funds through COMPETE (Operational Programme Thematic Factors of Competitiveness) and by portuguese funds through FCT (Foundation for Science and Technology) within the projects FCOMP-01-0124-FEDER-022674 and PEst-C/MAT/UI0013/2011.

References

1. Ali, M.M., Gabere, M.N.: A simulated annealing driven multi-start algorithm for bound constrained global optimization. J. Comput. Appl. Math., 233, 2661–2674 (2010)

2. Ali, M.M., Khompatraporn, C., Zabinsky, Z.B.: A numerical evaluation of several stochastic algorithms on selected continuous global optimization test problems. J. Glob. Optim., 31, 635–672 (2005)
3. Audet, C., Dennis Jr., J.E.: A pattern search filter method for nonlinear programming without derivatives. SIAM J. Optimiz., 14(4), 980–1010 (2004)
4. Costa, M.F.P., Fernandes, E.M.G.P.: Assessing the potential of interior point barrier filter line search methods: nonmonotone versus monotone approach, Optimization, 60(10-11), 1251–1268 (2011)
5. Fernandes, F.P., Costa, M.F.P., Fernandes, E.M.G.P.: Stopping rules effect on a derivative-free filter multistart algorithm for multilocal programmnig, In: ICACM 2012, file:131-1395-1-PB.pdf (6 pp) http://icacm.iam.metu.edu.tr/all-talks (2012)
6. Fernandes, F.P., Costa, M.F.P., Fernandes, E.M.G.P.: A derivative-free filter driven multistart technique for global optimization, In: ICCSA 2012 Part III, Lect. Notes Comput. Sc., Vol. 7335, Murgante, B. et al. (Eds.) 103–118 (2012)
7. Fletcher, R., Leyffer, S.: Nonlinear programming without a penalty function. Math. Program., 91, 239–269 (2002)
8. Floudas, C.A., Pardalos, P.M., Adjiman, C.S., Esposito, W.R., Gumus, Z.H., Harding, S.T., Klepeis, J.L., Meyer, C.A., Schweiger, C.A.: Handbook of Test Problems in Local and Global Optimization, Kluwer Academic Publishers (1999)
9. Hedar, A.R., Fukushima, M.: Derivative-Free Filter Simulated Annealing Method for Constrained Continuous Global Optimization, J. Glob. Optim., 35, 521–549 (2006)
10. Hendrix, E.M.T., G.-Tóth, B.: Introduction to Nonlinear and Global Optimization, Springer Optimization and Its Applications Vol 37 (2010)
11. Kolda, T.G., Lewis, R.M., Torczon, V.: Optimization by Direct Search: New Perspectives on Some Classical and Moddern Methods, SIAM Rev., 45(3), 385–482 (2003)
12. Koziel, S., Michalewicz, Z.: Evolutionary algorithms, homomorphous mappings, and constrained parameter optimization, Evol. Comput., 7(1), 19–44 (1999)
13. Krishnanand, K.N., Ghose, D.: Glowworm swarm optimization for simultaneous capture of multiple local optima of multimodal functions, Swarm Intell., 3, 87–124 (2009)
14. Lagaris, I.E., Tsoulos, I.G.: Stopping rules for box-constrained stochastic global optimization, Appl. Math. Comput., 197, 622–632 (2008)
15. Marti, R.: Multi-start methods, In: Handbook of Metaheuristics, Glover, F., Kochenberger, G. (Eds), Kluwer Academic Publishers, 355–368 (2003)
16. Ozdamar, L., Demirhan, M.: Experiments with new stochastic global optimization search techniques, Comput. Oper. Res., 27, 841–865 (2000)
17. Parsopoulos, K.E., Vrahatis, M.N.: On the computation of all global minimizers through particle swarm optimization, IEEE T. Evolut. Comput., 8(3), 211–224 (2004)
18. Pereira, A., Ferreira, O., Pinho, S. P., Fernandes, E.M.G.P.: Multilocal Programming and Applications, In: Handbook of Optimization, Intelligent Systems series, Zelinka, I. et al. (Eds.), Springer-Verlag, 157–186 (2013)
19. Ryoo, H.S., Sahinidis, N.V.: Global optimization of nonconvex NLPs and MINLPs with applications in process design, Comput. Chem. Eng., 19(5), 551–566 (1995)
20. Singh, G., Deb, K.: Comparison of multi-modal optimization algorithms based on evolutionary algorithms, In: GECCO' 06, ACM Press, 1305–1312 (2006)
21. I.G. Tsoulos, I.E. Lagaris, MinFinder: Locating all the local minima of a function, Computer Phys Com., 174, 166–179 (2006)

22. I.G. Tsoulos, A. Stavrakoudis, On locating all roots of systems of nonlinear equations inside bounded domain using global optimization methods, Nonlinear Anal. Real, 11, 2465–2471 (2010)
23. W. Tu, R.W. Mayne, Studies of multi-start clustering for global optimization, Int. J. Numer. Meth. Eng., 53(9), 2239–2252 (2002)
24. C. Voglis, I.E. Lagaris, Towards "Ideal Multistart". A stochastic approach for locating the minima of a continuous function inside a bounded domain, Appl. Math. Comput., 213, 1404–1415 (2009)