A Derivative-Free Filter Driven Multistart Technique for Global Optimization

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@mct.uminho.pt`
[3] Mathematics R&D Centre,
[4] Algoritmi R&D Centre,
University of Minho, 4710-057 Braga, Portugal
`emgpf@dps.uminho.pt`

Abstract. A stochastic global optimization method based on a multistart strategy and a derivative-free filter local search for general constrained optimization is presented and analyzed. In the local search procedure, approximate descent directions for the constraint violation or the objective function are used to progress towards the optimal solution. The algorithm is able to locate all the local minima, and consequently, the global minimum of a multi-modal objective function. The performance of the multistart method is analyzed with a set of benchmark problems and a comparison is made with other methods.

Keywords: Global optimization, Multistart, Descent Direction, Filter Method

1 Introduction

Global optimization problems arise in many engineering applications. Owing to the existence of multiple minima, it is a challenging task to solve a multilocal optimization problem and to identify all the global minima.

The purpose of this paper is to present a technique for solving constrained global optimization problems based on a multistart method that uses a filter methodology to handle the constraints of the problem. The problem to be addressed is of the following type

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

where, at least one of the functions $f, g_j : \mathbb{R}^n \longrightarrow \mathbb{R}$ is nonlinear and $\mathrm{F} = \{x \in \mathbb{R}^n : l_i \leq x_i \leq u_i, i = 1, \ldots, n, g_j(x) \leq 0, j = 1, \ldots, m\}$ is the feasible region.

Problems with general nonlinear equality constraints can be reformulated in the above form by introducing $h(x) = 0$ as inequality constraints $|h(x)| - \tau \leq 0$, where τ is a small positive relaxation parameter. Since this kind of problems may have many global and local (non-global) optimal solutions (convexity is not assumed), it is important to develop a methodology that is able to explore the entire search space, find all the (local) minima guaranteeing, in some way, that convergence to a previously found minimum is avoided, and identify the global ones.

The two major classes of methods for solving problem (1) globally are the deterministic and the stochastic one. One of the most known stochastic algorithms is the multistart. In the last decade some research has been focused on this type of methods [1, 6, 9–11]; see also [7] and the references therein included. The underlying idea of this method is to sample uniformly a point from the search region and to perform a local search, starting from this point, to obtain an optimal (local) solution, using a local technique. This is repeated until the stop conditions are met. One of the advantages of multistart is that it has the potential of finding all local minima; although, it has the drawback of locating the same solution more than once.

Here, we are specially interested in developing a simple to implement and efficient method for the identification of at least one global optimal solution of problem (1) that is based on a multistart paradigm. A multistart strategy is chosen due to its simplicity and previously observed practical good performance.

The herein proposed method does not compute or approximate any derivatives or penalty parameters. Our proposal for the local search relies on a procedure, namely, the approximate descent direction (ADD) method, which is a derivative-free procedure with high ability of producing a descent direction. The ADD method is combined with a (line search) filter method to generate trial solutions that might be acceptable if they improve the constraint violation or the objective function. Hence, the progress towards a solution that is feasible and optimal is carried out by a filter method. This is a recent strategy that has shown to be highly competitive with penalty function methods [2–4].

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 The Filter Driven Multistart Method

This section describes a multistart approximate descent direction filter-based approach, hereafter denoted by MADDF, that relies on a derivative-free local search procedure to converge to the local solutions of the problem. 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 are made by a simple local search approach. A derivative-free technique

that computes approximate descent directions, for either the constraint violation or the objective function, is implemented with reduced computational costs. To measure progress towards an optimal solution a filter methodology, as outlined in [4], 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 [2, 3, 5, 4].

2.1 A Multistart Strategy

The basic multistart algorithm starts by randomly generating a point x from the search space $S \subset \mathbb{R}^n$, and a local search procedure is applied from x to converge to a local minimum y. We will denote the implementation of the local procedure to provide the minimum y by $y = \mathcal{L}(x)$. Subsequently, another point is randomly generated from the search space and the local search is again applied to give another local minimum. This process is repeated until a stopping rule is satisfied. The pseudo-code of this procedure is presented below in Algorithm 1.

Algorithm 1 Basic multistart algorithm

1: Set $k = 1$;
2: Randomly generate x from S;
3: Compute $y_1 = \mathcal{L}(x)$;
4: **while** the stopping rule is not satisfied **do**
5:     Randomly generate x from S;
6:     Compute $y = \mathcal{L}(x)$;
7:     **if** $y \notin \{y_i, i = 1, \ldots, k\}$ **then**
8:         $k = k + 1$;
9:         Set $y_k = y$;
10:    **end if**
11: **end while**

Unfortunately, this multistart strategy has a drawback since the same local minimum may be found over and over again. To prevent the repetitive invoking of the local search procedure, converging to previously found local minima, clustering techniques have been incorporated into the multistart strategy. To guarantee that a local minimum is found only once, the concept of region of attraction of a local minimum is introduced.

Definition 1. *The region of attraction of a local minimum associated with a local search procedure $\mathcal{L}$ is defined as:*

$$A_i \equiv \{x \in \mathcal{S}, y_i = \mathcal{L}(x)\}, \tag{2}$$

where $y_i = \mathcal{L}(x)$ is the minimizer obtained when the local search procedure $\mathcal{L}$ is started at point x.

This concept is very important because it guarantees that the local search applied to any point x from the region of attraction A_i will converge eventually to the same minimizer y_i. Thus, after y_i has been found there is no point in starting the local search from any other point in that region of attraction.

Let N be the number of local minima in $\mathcal{S}$. From the previous definition it follows that

$$\mathcal{S} = \bigcup_{i=1}^{N} A_i \text{ and } A_i \cap A_j = \emptyset, \text{ for } i \neq j. \tag{3}$$

A multistart method that uses the concept of region of attraction proceeds as follows: it starts by randomly generating a point from $\mathcal{S}$, and applies a local search to obtain the first minimum y_1 with the region of attraction A_1. Afterwards, other points are randomly generated from $\mathcal{S}$ until a point is found that does not belong to A_1. Next, the local search is performed and a new minimizer y_2 is obtained, with the region of attraction A_2. The next point from which a local search will start does not belong to $A_1 \cup A_2$. This procedure continues until a stopping rule is satisfied. The corresponding multistart algorithm is presented in the Algorithm 2:

Algorithm 2 Multistart Clustering algorithm

1: Set $k = 1$;
2: Randomly generate x from S;
3: Compute $y_1 = \mathcal{L}(x)$ and the corresponding A_1;
4: **while** the stopping rule is not satisfied **do**
5:     Randomly generate x from S;
6:     **if** $x \notin \cup_{i=1}^{k} A_i$ **then**
7:         Compute $y = \mathcal{L}(x)$;
8:         $k = k + 1$;
9:         Set $y_k = y$ and compute the corresponding A_k;
10:    **end if**
11: **end while**

Theoretically, this algorithm invokes the local search procedure only N times, where N is the number of existing minima of (1). In practice, the regions of attraction A_k of the minima found so far are not easy to compute. A simple stochastic procedure is used to estimate the probability, p, that a randomly generated point will not belong to a specific set, which is the union of a certain number of regions of attraction, i.e., $p = P[x \notin \cup_{i=1}^{k} A_i]$. Using this reasoning, the new steps of the regions of attraction based multistart algorithm are described in Algorithm 3.

The probability p is estimated as follows [11]. Let the maximum attractive radius of the minimizer y_i be defined by:

$$R_i = \max_{j} \left\{ \left\| x_i^{(j)} - y_i \right\| \right\}, \tag{4}$$

Algorithm 3 Ideal Multistart algorithm

1: Set $k = 1$;
2: Randomly generate x from S;
3: Compute $y_1 = \mathcal{L}(x)$ and the corresponding A_1;
4: **while** the stopping rule is not satisfied **do**
5:     Randomly generate x from S;
6:     Compute $p = P[x \notin \cup_{i=1}^{k} A_i]$;
7:     Let ζ be a uniform distributed number in $(0, 1)$;
8:     **if** $\zeta < p$ **then**
9:         Compute $y = \mathcal{L}(x)$;
10:        **if** $y \notin \{y_i, i = 1, ..., k\}$ **then**
11:            $k = k + 1$;
12:            Set $y_k = y$ and compute the corresponding A_k;
13:        **end if**
14:     **end if**
15: **end while**
---

where $x_i^{(j)}$ are the generated points which led to the minimizer y_i. Given a randomly generated point x, let $z = \frac{\|x - y_i\|}{R_i}$. Clearly, if $z \leq 1$ then x is likely to be inside the region of attraction of y_i. On the other hand, if the direction from x to y_i is ascent then x is likely to be outside the region of attraction of y_i. Based on a suggestion presented in [11], an estimate of the probability that $x \notin A_i$ is herein computed by:

$$p(x \notin A_i) = \begin{cases} 1, & \text{if } z > 1 \text{ or the direction from } x \text{ to } y_i \text{ is ascent} \\ \varrho \, \phi(z, l), & \text{otherwise} \end{cases} \quad (5)$$

where $0 \leq \varrho \leq 1$ is a factor that depends on the directional derivative of f along the direction from x to y_i, l is the number of times y_i has been identified/recovered so far and the function $\phi(z, l)$ satisfies the properties:

$$\lim_{z \to 0} \phi(z, l) \to 0, \ \lim_{z \to 1} \phi(z, l) \to 1, \ \lim_{l \to \infty} \phi(z, l) \to 0 \ \text{ and } \ 0 < \phi(z, l) < 1.$$

In the Ideal Multistart method [11], Voglis and Lagaris propose the

$$\phi(z, l) = z \exp\left(-l^2(z - 1)^2\right) \ \text{ for all } \ z \in (0, 1). \quad (6)$$

Since the Algorithm 3 has the potential of finding all local minima and one global solution is to be required, each solution is compared with the previously identified solutions and the one with the most extreme value is always saved.

2.2   The Derivative-Free Filter Local Procedure

The local search procedure is an iterative method that is applied to a randomly generated point x and provides a trial point y that is an approximate minimizer of problem (1). Our proposal for the local search $\mathcal{L}$ is an Approximate Descent

Direction Filter (ADDF) method. The point y is computed based on a direction d and a step size $\alpha \in (0, 1]$ in such a way that

$$y = x + \alpha\,d. \tag{7}$$

The procedure that decides which step size is accepted to generate an acceptable approximate minimizer is a filter method. The herein proposed multistart method uses the filter set concept [4] that has the ability to explore both feasible and infeasible regions. This technique incorporates the concept of nondominance, present in the field of multiobjective optimization, to build a filter that is able to accept a trial point if it improves either the objective function or the constraint violation, relative to the current point. Filter-based algorithms treat the optimization problem as a biobjective problem aiming to minimize both the objective function and the nonnegative constraint violation function. In this way, the previous constrained problem (1) is reformulated as a biobjective problem involving the original objective function f and the constraint violation function θ, as follows:

$$\min_{x \in \mathcal{S}} \left(f(x), \theta(x) \right) \tag{8}$$

where for $\beta \in \{1, 2\}$

$$\theta(x) = \sum_{i=1}^{m} \left(\max_i \{0, g_i(x)\} \right)^{\beta} + \sum_{i=1}^{n} \left(\left(\max_i \{0, x_i - u_i\} \right)^{\beta} + \left(\max_i \{0, l_i - x_i\} \right)^{\beta} \right). \tag{9}$$

After a search direction d has been computed, a step size α is determined by a backtracking line search technique. A decreasing sequence of α values is tried until a set of acceptance conditions are satisfied. The trial point y, in (7), is acceptable if sufficient progress in θ or in f is verified, relative to the current point x, as shown:

$$\theta(y) \leq (1 - \gamma_\theta)\,\theta(x) \ \text{ or } \ f(y) \leq f(x) - \gamma_f\,\theta(x) \tag{10}$$

where $\gamma_\theta, \gamma_f \in (0, 1)$. However, when x is (almost) feasible, i.e., in practice when $\theta(x) \leq \theta_{\min}$, the trial point y has to satisfy only the condition

$$f(y) \leq f(x) - \gamma_f\,\theta(x) \tag{11}$$

to be acceptable, where $0 < \theta_{\min} \ll 1$. To prevent cycling between points that improve either θ or f, at each iteration, the algorithm maintains the filter $\mathcal{F}$ which is a set of pairs (θ, f) that are prohibited for a successful trial point. During the backtracking line search procedure, the y is acceptable only if $(\theta(y), f(y)) \notin \mathcal{F}$. If the stopping conditions are not satisfied (see (14) ahead), $x \leftarrow y$ and this procedure is repeated.

The filter is initialized with pairs (θ, f) that satisfy $\theta \geq \theta_{\max}$, where $\theta_{\max} > 0$ is the upper bound on θ. Furthermore, whenever y is accepted because condition (10) is satisfied, the filter is updated by the formula

$$\mathcal{F} = \mathcal{F} \cup \left\{ (\theta, f) \in \mathbb{R}^2 : \theta > (1 - \gamma_\theta)\theta(x) \text{ and } f > f(x) - \gamma_f\theta(x) \right\}.$$

When it is not possible to find a point y with a step size $\alpha > \alpha_{\min}$ ($0 < \alpha_{\min} \ll 1$) that satisfy one of the conditions (10) or (11), a restoration phase is invoked. In this phase, the algorithm recovers the best point in the filter, herein denoted by $x_{\mathcal{F}}^{best}$, and a new trial point is determined according to the strategy based on equation (7).

The algorithm implements the ADD method [5] to compute the direction d, required in (7). This strategy has a high ability of producing a descent direction for a specific function. The ADD method is a derivative-free procedure which uses several points around a given point $x \in \mathbb{R}^n$ to generate an approximate descent direction for a function ψ at x [5]. More specifically, the ADD method chooses r exploring points close to x, in order to generate an approximate descent direction $d \in \mathbb{R}^n$ for ψ at x. Hence, the direction $d = \frac{v}{\|v\|}$ is computed at x, after generating r points $\{a_i\}_{i=1}^r$ close to x, as shown:

$$v = \sum_{i=1}^{r} w_i \, e_i \tag{12}$$

where

$$w_i = \frac{\Delta\psi_i}{\displaystyle\sum_{j=1}^{r} |\Delta\psi_j|}, \; \Delta\psi_i = \psi(a_i) - \psi(x), \, i = 1, \dots, r \tag{13}$$

$$e_i = -\frac{a_i - x}{\|a_i - x\|} \;\; i = 1, \dots, r.$$

In the ADDF context, the ADD method generates the search direction d, at a given point x, according to the following rules:

- If x is feasible (in practice, if $\theta(x) < \theta_{tol}$), the ADD method computes an approximate descent direction d for the objective function f at x and then $\psi = f$ in (13);
- If x is infeasible, the ADD method is used to compute an approximate descent direction d for the constraint violation function θ, at x, and in this case $\psi = \theta$.

To judge the success of the ADDF algorithm, the three below presented conditions are applied simultaneously, i.e., if

$$|f(y) - f(x)| \leq 10^{-4}\,|f(y)| + 10^{-6} \; \wedge \; |\theta(y) - \theta(x)| \leq 10^{-4}\theta(y) + 10^{-6}$$
$$\wedge \; \|y - x\| \leq 10^{-4}\|y\| + 10^{-6} \tag{14}$$

hold, the local search procedure stops with a successful approximate local minimizer of problem (1). The proposed algorithm for the local procedure is presented in Algorithm 4.

Algorithm 4 ADDF algorithm

Require: x (sampled in multistart); Set $x_{\mathcal{F}}^{best} = x$ and $\tilde{x} = x$;
1: Initialize the filter;
2: **while** the stopping conditions are not satisfied **do**
3:    Set $x = \tilde{x}$;
4:    Use ADD to compute v by (12);
5:    Set $\alpha = 1$;
6:    Compute y using (7);
7:    **while** new trial y is not acceptable **do**
8:       Check acceptability of trial point, using (10) and (11);
9:       **if** acceptable by the filter **then**
10:          Update the filter if appropriate;
11:          Set $\tilde{x} = y$; Update $x_{\mathcal{F}}^{best}$;
12:       **else**
13:          Set $\alpha = \alpha/2$;
14:          **if** $\alpha < \alpha_{\min}$ **then**
15:             Set $\alpha = 1$; Set $x = x_{\mathcal{F}}^{best}$;
16:             Invoke restoration phase;
17:          **end if**
18:          Compute y using (7);
19:       **end if**
20:    **end while**
21: **end while**

2.3 Stopping Rule

Good stopping rules to identify multiple optimal solutions should combine reliability and economy. A reliable rule is one that stops only when all minima have been identified with certainty. An economical rule is one that invokes the local search the least number of times to verify that all minima have been found. A lot of research about stopping rules has been carried out in the past (see [6] and the references therein included). There are three established rules that have been successfully used [6].

We choose to use the following stopping condition [6]. If s denotes de number of recovered local minima after having performed t local search procedures, then the estimate of the fraction of the uncovered space is given by

$$P(s) = \frac{s(s+1)}{t(t-1)} \tag{15}$$

and the stopping rule is

$$P(s) \leq \epsilon \tag{16}$$

with ϵ being a small positive number.

3 Experimental Results

The MADDF method was coded in MatLab and the results were obtained in a PC with an Intel(R) Core(TM)2 Duo CPU P7370 2.00GHz processor and 3 GB of memory.

To perform some comparisons between other methods, it is necessary to set the MADDF parameters. The parameter τ used to reformulate equality into inequality constraints was set to $\tau = 10^{-5}$. Since derivatives are not provided to the algorithm, the factor ϱ is estimated and set to 0.05. The closer the direction $(y - x)$ is to the greatest decrease of f, the smaller is ϱ. The power factor used in equation (9) was set to 2 and to generate the approximate descent directions we set $r = 2$ and $r_{ADD} = 10^{-3}$ (the radius of the neighborhood in which the exploring points are generated), as suggested in [5]. In ADDF method, $\gamma_\theta = \gamma_f = 10^{-5}$, $\alpha_{\min} = 10^{-6}$, $\theta_{tol} = 10^{-5}$, $\theta_{\min} = 10^{-3} \max\{1, 1.25\theta(x_{initial})\}$, $\theta_{\max} = \max\{1, 1.25\theta(x_{initial})\}$, where $x_{initial}$ is the initial point in the local search.

In this section, we report the performance of the MADDF algorithm on 14 well-known test problems, which are shown in the Appendix of this paper, in an effort to make the article as self-contained as possible. The MADDF code was applied 30 times to solve each problem.

In the first set of experiments, summarized in Table 1, the stopping rule (16) with $\epsilon = 0.06$ is used. Table 1 summarizes the MADDF results obtained for each

Table 1. Numerical results obtained with MADDF and FSA [5].

Prob.	f_{OPT}	Method	Best	Average	Worst	S.D.	Av. f.eval.
g3	-1	MADDF	-1.0000968	-0.9998019	-0.9993183	0.000208	45466
		in [5]	-1.0000015	-0.9991874	-0.9915186	0.001653	314938
g6	-6961.81388	MADDF	-6961.23915	-6957.99845	-6954.65040	1.92544	15544
		in [5]	-6961.81388	-6961.81388	-6961.81388	0.00000	44538
g8	-0.095825	MADDF	-0.095825	-0.095825	-0.095825	0.000000	4999
		in [5]	-0.095825	-0.095825	-0.095825	0.000000	56476
g9	680.630057	MADDF	681.08698	683.31319	685.49488	1.38392	38099
		in [5]	680.63008	680.63642	680.69832	0.014517	324596
g11	0.75	MADDF	0.749980	0.750204	0.751048	0.000295	139622
		in [5]	0.749999	0.749999	0.749999	0.000000	23722

test problem as well as the best known objective function value for each problem ('f_{OPT}'). In order to show more details concerning the quality of the obtained solution, the best ('Best'), the average ('Average'), the worst ('Worst'), as well as the standard deviation ('S.D.') of the obtained objective function values are also reported in Table 1. The average number of function evaluations required to converge to the solution ('Av. f.eval.') is also reported. In this table, the results for each problem using the Filter Simulated Annealing Method (FSA) proposed in [5] are also reported.

Problems g3 and g8 were originally maximization problems. They were rewritten as minimization problems. As it can be seen, for all five problems, MADDF method finds the global minimum. The quality of the solution is good. The worst results are obtained with problems g6 and g9. The average number of function evaluations is much smaller than the one reported by FSA method, for all test problems, except g11. In [5], a comparison with four evolutionary algorithms (EA) was made. These EA methods need a higher number of function evaluations than FSA and, consequently, our proposed method. Hence, the MADDF is better than the EA methods used in [5] as far as the number of function evaluations is concerned. These four EA-based methods are: Homomorphous Mappings (HM) method, Stochastic Ranking (SR) method, Adaptive Segregational Constraint Handling EA (ASCHEA) method and Simple Multimembered Evolution Strategy (SMES) method. In Table 2, the results of the proposed MADDF method are repeated, in order to compare them with those of the EA methods. We may observe that the MADDF method is competitive with the EA methods relative to the quality of the solution.

Table 2. Numerical results obtained with MADDF and EA methods [5].

Prob.	Method	Best	Average	Worst
g3	MADDF	-1.0000968	-0.9998019	-0.9993183
	HM	-0.9997	-0.9989	-0.9978
	SR	-1.000	-1.000	-1.000
	ASCHEA	-1	-0.99989	N.A
	SMES	-1.001038	-1.000989	-1.000579
g6	MADDF	-6961.23915	-6957.99845	-6954.65040
	HM	-6952.1	-6342.6	-5473.9
	SR	-6961.814	-6875.940	-6350.262
	ASCHEA	-6961.81	-6961.81	N.A
	SMES	-6961.813965	-6961.283984	-6961.481934
g8	MADDF	-0.095825	-0.095825	-0.095825
	HM	-0.0958250	-0.0891568	-0.0291438
	SR	-0.095825	-0.095825	-0.095825
	ASCHEA	-0.09582	-0.09582	N.A
	SMES	-0.095826	-0.095826	-0.095826
g9	MADDF	681.08698	683.31319	685.49488
	HM	680.91	681.16	683.18
	SR	680.630	680.656	680.763
	ASCHEA	680.630	680.641	N.A
	SMES	680.631592	680.643410	680.719299
g11	MADDF	0.749980	0.750204	0.751048
	HM	0.75	0.75	0.75
	SR	0.750	0.750	0.750
	ASCHEA	0.75	0.75	N.A
	SMES	0.749090	0.749358	0.749830

To establish other comparisons with other stochastic global methods, we applied the following conditions that appear in [8] to the next set of nine problems and the results are shown in the next two tables. Two conditions to judge the success of the run were applied. First,

$$\left|f(x^{best}) - f_{OPT}\right| \leq 10^{-4} \left|f_{OPT}\right| \tag{17}$$

where $f(x^{best})$ is the best solution found so far and f_{OPT} is the known optimal solution available in the literature, is used instead of the stopping rule (16).

In practice, when solving any benchmark problem whose global optimal solution is known, the Algorithm 3 is stopped as soon as a sufficiently accurate solution is found, according to the condition in (17). We remark that in multistart clustering methods based on the region of attraction, the stopping rule of the algorithm is crucial to promote convergence to all local optimal solutions (cf. [6]). In the presented algorithm, the likelihood of choosing a point that does not belong to the regions of attraction of previously identified optimal solutions is very high, although convergence to a local minimum that has not been located before is not guaranteed. Convergence to an optimal solution more than once may happen. So far, during the herein presented experiments this situation has occurred although not frequently.

Table 3 contains the average number of function evaluation obtained after the 30 runs. A comparison is made with the results reported in [8] - two artificial fish swarm based methods (AFS and m-AFS) and an electromagnetism-like mechanism algorithm (EM).

Table 3. Average number of function evaluations, using (17).

Prob.	f_{OPT}	MADDF	AFS	m-AFS	EM
BR	0.39789	493	550	475	315
CB6	-1.03160	660	331	247	233
GP	3.00000	787	676	417	420
H3	-3.86278	6022	2930	1891	1114
H6	-3.32237	5001	7091	2580	2341
S5	-10.1532	2396	3928	1183	3368
S7	-10.4029	2655	4033	1103	1782
S10	-10.5364	3514	2069	1586	5620
SBT	-186.731	938	472	523	358

From the table we may conclude that the performance of the proposed MADDF is similar to the AFS algorithm, in terms of efficiency (number of function evaluations), while m-AFS and EM are slightly better than MADDF.

The results shown in Table 4 were obtained using the following stopping condition,

$$\left|f(x^{best}) - f_{OPT}\right| \leq 10^{-3} \tag{18}$$

instead of (16). This set of experiments is compared with the results obtained by AFS, m-AFS, two particle swarm algorithms, PSO-RPB and PSO-HS, and a differential evolution method, DE, available in [8].

Table 4. Average number of function evaluations, using (18).

Prob.	f_{OPT}	MADDF	AFS	m-AFS	PSO-RPB	PSO-HS	DE
BR	0.39789	506	651	438	2652	2018	1305
CB6	-1.03160	660	246	245	2561	2390	1127
GP	3.00000	1063	562	485	2817	1698	884
H3	-3.86278	5845	1573	1142	3564	2948	1238
H6	-3.32237	7559	7861	2845	8420	8675	7053
S5	-10.1532	2929	3773	1150	6641	6030	5824
S7	-10.4029	4428	2761	1240	6860	6078	5346
S10	-10.5364	4489	2721	1190	6747	5602	4822
SBT	-186.731	1867	659	516	4206	6216	2430

As it can be seen, the MADDF method has a similar performance to AFS method, outperforms the two variants of the particle swarm optimization and the differential evolution methods, although is less efficient than m-AFS.

4 Conclusions and Future Work

We present a multistart technique based on a derivative-free filter method to solve constrained global optimization problems. The multistart strategy relies on the concept of region of attraction to prevent the repetitive use of the local search procedure in order to avoid convergence to previously found local minima. Our proposal for the local search computes approximate descent directions combined with a (line search) filter method to generate a sequence of approximate solutions that improve either the constraint violation or the objective function value.

A set of 14 well-known test problems was used and the results obtained are very promising. In all problems we could reach the global minimum and the performance of the algorithm, in terms of number of function evaluations and the quality of the solution is quite satisfactory.

In the future, we aim to extend MADDF method to multilocal programming, so that all global as well as local (non-global) minimizers are obtained. This is an interesting and promising area of research due to their real applications in the chemical engineering field.

References

1. Ali, M.M., Gabere, M.N.: A simulated annealing driven multi-start algorithm for bound constrained global optimization. Journal of Computational and Applied Mathematics. 233, 2661-2674 (2010)
2. Audet, C., Dennis, Jr., J.E.: A pattern search filter method for nonlinear programming without derivatives. SIAM Journal on Optimization. 14(4) 980–1010 (2004)
3. 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)
4. Fletcher, R., Leyffer, S.: Nonlinear programming without a penalty function. Mathematical Programming. 91, 239–269 (2002)
5. Hedar, A.R., Fukushima, M.: Derivative-Free Filter Simulated Annealing Method for Constrained Continuous Global Optimization. Journal of Global Optimization. 35, 521–549 (2006)
6. Lagaris, I.E., Tsoulos, I.G.: Stopping rules for box-constrained stochastic global optimization. Applied Mathematics and Computation. 197, 622–632 (2008)
7. Marti, R.: Multi-start methods. In: Glover F, Kochenberger G (eds) Handbook of metaheuristics. pp 355–368. Kluwer, Dordrech (2003)
8. Rocha, A.M.A.C., Fernandes, E.M.G.P.: Mutation-Based Artificial Fish Swarm Algorithm for Bound Constrained Global Optimization. In: Numerical Analysis and Applied Mathematics ICNAAM 2011 AIP Conf. Proc. 1389, 751–754 (2011)
9. Tsoulos, I.G., Lagaris, I.E.: MinFinder: Locating all the local minima of a function, Computer Physics Communications. 174, 166–179 (2006)
10. Tu, W., Mayne, R.W.: Studies of multi-start clustering for global optimization. International Journal for Numerical Methods in Engineering. 53(9), 2239–2252 (2002)
11. Voglis, C., Lagaris, I.E.: Towards "Ideal Multistart". A stochastic approach for locating the minima of a continuous function inside a bounded domain. Applied Mathematics and Computation. 213, 1404–1415 (2009)

Appendix - Test problems

- **Branin (BR)**

$$\min f(x) \equiv (x_2 - \frac{5.1}{4\pi^2}x_1^2 + \frac{5}{\pi}x_1 - 6)^2 + 10\left(1 - \frac{1}{8\pi}\right)\cos(x_1) + 10$$
subject to $\quad -5 \leq x_1 \leq 10$
$\qquad\qquad\quad 0 \leq x_2 \leq 15$

- **Camel (CB6)**

$$\min f(x) \equiv \left(4 - 2.1x_1^2 + \frac{x_1^4}{3}\right)x_1^2 + x_1 x_2 - 4(1 - x_2^2)x_2^2$$
subject to $\quad -2 \leq x_i \leq 2, \; i = 1, 2$

- **Goldestein and Price (GP)**

$$\min f(x) \equiv (1 + (x_1 + x_2 + 1)^2(19 - 14x_1 + 3x_1^2 - 14x_2 + 6x_1 x_2 + 3x_2^2)) \times$$
$$\times (30 + (2x_1 - 3x_2)^2(18 - 32x_1 + 12x_1^2 + 48x_2 - 36x_1 x_2 + 27x_2^2))$$
subject to $\quad -2 \leq x_i \leq 2, \; i = 1, 2$

– **Hartman3 (H3)**

$$\min f(x) \equiv -\sum_{i=1}^{4} c_i \exp\left(-\sum_{j=1}^{3} a_{ij}(x_j - p_{ij})^2\right)$$

subject to $\quad 0 \le x_i \le 1, i = 1, 2, 3$

with

$$a = \begin{bmatrix} 3 & 10 & 30 \\ 0.1 & 10 & 35 \\ 3 & 10 & 30 \\ 0.1 & 10 & 35 \end{bmatrix}, \ c = \begin{bmatrix} 1 \\ 1.2 \\ 3 \\ 3.2 \end{bmatrix} \text{ and } p = \begin{bmatrix} 0.3689 & 0.117 & 0.2673 \\ 0.4699 & 0.4387 & 0.747 \\ 0.1091 & 0.8732 & 0.5547 \\ 0.03815 & 0.5743 & 0.8828 \end{bmatrix}$$

– **Hartman6 (H6)**

$$\min f(x) \equiv -\sum_{i=1}^{4} c_i \exp\left(-\sum_{j=1}^{6} a_{ij}(x_j - p_{ij})^2\right)$$

subject to $\quad 0 \le x_i \le 1, i = 1, \ldots, 6$

with $a = \begin{bmatrix} 10 & 3 & 17 & 3.5 & 1.7 & 8 \\ 0.05 & 10 & 17 & 0.1 & 8 & 14 \\ 3 & 3.5 & 1.7 & 10 & 17 & 8 \\ 17 & 8 & 0.05 & 10 & 0.1 & 14 \end{bmatrix}$,

$$c = \begin{bmatrix} 1 \\ 1.2 \\ 3 \\ 3.2 \end{bmatrix} \text{ and } p = \begin{bmatrix} 0.1312 & 0.1696 & 0.5569 & 0.0124 & 0.8283 & 0.5886 \\ 0.2329 & 0.4135 & 0.8307 & 0.3736 & 0.1004 & 0.9991 \\ 0.2348 & 0.1451 & 0.3522 & 0.2883 & 0.3047 & 0.6650 \\ 0.4047 & 0.8828 & 0.8732 & 0.5743 & 0.1091 & 0.0381 \end{bmatrix}$$

– **Shekel-5 (S5)**

$$\min f(x) \equiv -\sum_{i=1}^{5} \frac{1}{(x - a_i)(x - a_i)^T + c_i}$$

subject to $\quad 0 \le x_i \le 10, i = 1, \ldots, 4$

with

$$a = \begin{bmatrix} 4 & 4 & 4 & 4 \\ 1 & 1 & 1 & 1 \\ 8 & 8 & 8 & 8 \\ 6 & 6 & 6 & 6 \\ 3 & 7 & 3 & 7 \end{bmatrix} \text{ and } c = \begin{bmatrix} 0.1 \\ 0.2 \\ 0.2 \\ 0.4 \\ 0.4 \end{bmatrix}$$

– **Shekel-7 (S7)**

$$\min f(x) \equiv -\sum_{i=1}^{7} \frac{1}{(x - a_i)(x - a_i)^T + c_i}$$

subject to $\quad 0 \le x_i \le 10, i = 1, \ldots, 4$

with

$$a = \begin{bmatrix} 4 & 4 & 4 & 4 \\ 1 & 1 & 1 & 1 \\ 8 & 8 & 8 & 8 \\ 6 & 6 & 6 & 6 \\ 3 & 7 & 3 & 7 \\ 2 & 9 & 2 & 9 \\ 5 & 3 & 5 & 3 \end{bmatrix} \quad \text{and} \quad c = \begin{bmatrix} 0.1 \\ 0.2 \\ 0.2 \\ 0.4 \\ 0.4 \\ 0.6 \\ 0.3 \end{bmatrix}$$

– **Shekel-10 (S10)**

$$\min f(x) \equiv -\sum_{i=1}^{10} \frac{1}{(x - a_i)(x - a_i)^T + c_i}$$

subject to $\quad 0 \le x_i \le 10, i = 1, \ldots, 4$

with

$$a = \begin{bmatrix} 4 & 4 & 4 & 4 \\ 1 & 1 & 1 & 1 \\ 8 & 8 & 8 & 8 \\ 6 & 6 & 6 & 6 \\ 3 & 7 & 3 & 7 \\ 2 & 9 & 2 & 9 \\ 5 & 5 & 3 & 3 \\ 8 & 1 & 8 & 1 \\ 6 & 2 & 6 & 2 \\ 7 & 3.6 & 7 & 3.6 \end{bmatrix} \quad \text{and} \quad c = \begin{bmatrix} 0.1 \\ 0.2 \\ 0.2 \\ 0.4 \\ 0.4 \\ 0.6 \\ 0.3 \\ 0.7 \\ 0.5 \\ 0.5 \end{bmatrix}$$

– **Shubert (SBT)**

$$\min f(x) \equiv \left(\sum_{i=1}^{5} i \cos((i+1)x_1 + i) \right) \left(\sum_{i=1}^{5} i \cos((i+1)x_2 + i) \right)$$

subject to $\quad -10 \le x_i \le 10, i = 1, 2$

– **Problem G3**

$\min f(x) \equiv -(\sqrt{n})^n \prod_{i=1}^{n} x_i$

subject to $\quad \sum_{i=1}^{n} x_i^2 - 1 = 0$

$\qquad\qquad 0 \le x_i \le 1, i = 1, 2$

– **Problem G6**

$\min f(x) \equiv (x_1 - 10)^3 + (x_2 - 20)^3$

subject to $\quad -(x_1 - 5)^2 - (x_2 - 5)^2 + 100 \le 0$

$\qquad\qquad (x_1 - 6)^2 + (x_2 - 5)^2 - 82.81 \le 0$

$\qquad\qquad 13 \le x_1 \le 100$

$\qquad\qquad 0 \le x_2 \le 100,$

– **Problem G8**

$$\min f(x) \equiv -\frac{\sin^3(2\pi x_1)\sin(2\pi x_2)}{x_1^3(x_1+x_2)}$$

subject to $x_1^2 - x_2 + 1 \le 0$

$1 - x_1 + (x_2 - 4)^2 \le 0$

$0 \le x_i \le 10, i = 1, 2$

– **Problem G9**

$$\min f(x) \equiv (x_1 - 10)^2 + 5(x_2 - 12)^2 + x_3^4 + 3(x_4 - 11)^2 + \dots$$

$$+10x_5^6 + 7x_6^2 + x_7^4 - 4x_6 x_7 - 10x_6 - 8x_7$$

subject to $v1 + 3v2^2 + x_3 + 4x_4^2 + 5x_5 - 127 \le 0$

$7x_1 + 3x_2 + 10x_3^2 + x_4 - x_5 - 282 \le 0$

$23x_1 + v2 + 6x_6^2 - 8x_7 - 196 \le 0$

$2v1 + v2 - 3x_1 x_2 + 2x_3^2 + 5x_6 - 11x_7 \le 0$

$-10 \le x_i \le 10, i = 1, \dots, 7$

with $v1 = 2x_1^2; v2 = x_2^2$

– **Problem G11**

$$\min f(x) \equiv -x_1^2 + (x_2 - 1)^2$$

subject to $x_2 - x_1^2 - 1 = 0$

$-1 \le x_i \le 1, i = 1, 2$