A Two-Phase Heuristic Coupled DIRECT Method for Bound Constrained Global Optimization

M. Fernanda P. Costa, Edite M. G. P. Fernandes and Ana Maria A. C. Rocha

Abstract In this paper, we investigate the use of a simple heuristic in the DIRECT method context, aiming to select a set of the hyperrectangles that have the lowest function values in each *size* group. For solving bound constrained global optimization problems, the proposed heuristic divides the region where the hyperrectangles with the lowest function values in each *size* group lie into three subregions. From each subregion, different numbers of hyperrectangles are selected depending on the subregion they lie. Subsequently, from those selected hyperrectangles, the potentially optimal ones are identified for further division. Furthermore, the two-phase strategy aims to firstly encourage the global search and secondly enhance the local search. Global and local phases differ on the number of selected hyperrectangles from each subregion. The process is repeated until convergence. Numerical experiments carried out until now show that the proposed two-phase heuristic coupled DIRECT method is effective in converging to the optimal solution.

M. Fernanda P. Costa
Centre of Mathematics,
Department of Mathematics,
University of Minho, Campus de Gualtar, 4710-057 Braga, Portugal
e-mail: mfc@math.uminho.pt

Edite M.G.P. Fernandes
ALGORITMI Center
e-mail: emgpf@dps.uminho.pt

Ana Maria A.C. Rocha
ALGORITMI Center,
Department of Production and Systems,
University of Minho, Campus de Gualtar, 4710-057 Braga, Portugal
e-mail: arocha@dps.uminho.pt

1 Introduction

This paper addresses the use of a DIRECT-type method that coupled with a simple heuristic and a two-phase strategy aims to globally solve non-smooth and non-convex bound constrained optimization problems. The bound constrained global optimization (BCGO) problem can de stated as:

$$\min_{\mathbf{x}\in\Omega} f(\mathbf{x}) , \tag{1}$$

where $f : \mathbb{R}^n \to \mathbb{R}$ is a nonlinear function and $\Omega = \{\mathbf{x} \in \mathbb{R}^n : -\infty < l_i \leq x_i \leq u_i < \infty, i = 1,\ldots,n\}$ is a bounded feasible region. We assume that the optimal set $\mathbf{X}^*$ of the problem (1) is nonempty and bounded, $\mathbf{x}^*$ is a global minimizer and f^* represents the global optimal value.

When the function f is non-smooth, or its evaluation requires different simulations, and those simulations may add noise to the problem, analytical or numerical gradient-based methods may fail to solve the problem (1). Derivative-free methods, like the DIRECT method [1, 2], can solve it. The main idea in the DIRECT method is the partition of the feasible region into an increasing number of each time smaller hyperrectangles. At each iteration, a set of the most promising hyperrectangles are identified for further division. DIRECT needs to store all the information about all the generated hyperrectangles. This means that for larger dimensional problems, computational requirements may prevent DIRECT to find a high quality solution. DIRECT has strong convergence properties and produces a good coverage of the feasible region [3]. For the hyperrectangle division, DIRECT uses two criteria: the *size* of the hyperrectangle to favor the global search feature of the algorithm and the *value* of the hyperrectangle, translated by the objective function value at the center point of the hyperrectangle, to give preference to its local search feature. DIRECT-type algorithms that are more biased toward local search are proposed in [4, 5]. They are mostly suitable for small problems with one global minimizer and a few local minimizers. In [3], the deterministic partition strategy of the DIRECT method is used, in a multi-start context, to perform local minimizations starting from the center points of the most promising hyperrectangles. Globally biased searches are also reinforced in DIRECT by making use of a new technique for selecting the hyperrectangles to be divided [6, 7, 8].

For further details on the original DIRECT and other recent interesting modifications, we refer the reader to [6, 7, 8, 9, 10].

This paper investigates the use of a DIRECT-type method coupled with a heuristic aiming to potentiate the exploration of the most promising regions in the DIRECT method context. The heuristic categorizes the hyperrectangles with the lowest function values in each *size* group into three subregions for further sampling and division. Additionally, a two-phase strategy aims to cyclically encourage the global search phase (first phase) and enhance the local search one (second phase). Our proposal reinforces the global search capabilities of the DIRECT by avoiding the selection of the hyperrectangles that were mostly divided and choosing all the hyperrectangles with largest sizes (first phase). Conversely, when the new algorithm enters the second phase, the hyperrectangles with largest sizes are mostly prevented from being selected and the ones with smallest sizes are all included in the selection.

The paper is organized as follows. Section 2 briefly presents the main ideas of the DIRECT method and Sect. 3 describes the heuristic and the two-phase strategy in the DIRECT method context. Finally, Sect. 4 contains the results of our preliminary numerical experiments and we conclude the paper with the Sect. 5.

2 DIRECT Method

The DIRECT (DIviding RECTangles) algorithm has been originally proposed to solve BCGO problems like (1) where f is assumed to be a continuous function, by producing finer and finer partitions of the hyperrectangles generated from Ω [1]. The algorithm is a modification of the standard Lipschitzian approach, in which f must satisfy the Lipschitz condition

$$|f(\mathbf{x}_1) - f(\mathbf{x}_2)| \leq K\|\mathbf{x}_1 - \mathbf{x}_2\| \text{ for all } \mathbf{x}_1, \mathbf{x}_2 \in \Omega \ ,$$

where $K > 0$ is the Lipschitz constant. DIRECT is a derivative-free and deterministic global optimizer since it is able to explore potentially optimal regions in order to converge to the global optimum solution, thus avoiding to be trapped in a local optimum solution. It does not require any derivative information or the value of the Lipschitz constant [2]. DIRECT views the Lipschitz constant as a weighting parameter that balances global and local search. These searches are carried out by exploring some of the hyperrectangles in the current partition of Ω, in order to divide them further [5, 11]. First, the method organizes hyperrectangles by groups of the same *size* and considers dividing in each group the hyperrectangles that have the lowest value of the objective function – herein denoted by *candidate* hyperrectangles. However, not all of these *candidate* hyperrectangles are divided. The selection falls on the hyperrectangles that satisfy the following two criteria that define a potentially optimal hyperrectangle (POH):

Definition 1 Given the partition $\{P^i : i \in I\}$ of Ω, let ε be a positive constant and let $f_{\min}$ be the current best function value. A hyperrectangle j is said to be potentially optimal if there exists some rate-of-change constant $\hat{K}_j > 0$ such that

$$f(\mathbf{c}_j) - \frac{\hat{K}_j}{2}\|\mathbf{u}^j - \mathbf{l}^j\| \leq f(\mathbf{c}_i) - \frac{\hat{K}_j}{2}\|\mathbf{u}^i - \mathbf{l}^i\|, \forall i \in I \tag{2}$$

$$f(\mathbf{c}_j) - \frac{\hat{K}_j}{2}\|\mathbf{u}^j - \mathbf{l}^j\| \leq f_{\min} - \varepsilon|f_{\min}| \ , \tag{3}$$

where $\mathbf{c}_j$ is the center and $\|\mathbf{u}^j - \mathbf{l}^j\|/2$ is a measure of the *size* of hyperrectangle j.

The use of $\hat{K}_j$ intends to show that it is not the Lipschitz constant but it is just a rate-of-change constant [1]. Condition in (2) aims to check if the lower bound on the minimum of f on the hyperrectangle j is lower than the lower bounds on the minima of the other hyperrectangles of the partition P^i (for the hyperrectangle j to be potentially optimal). Condition (3) aims to balance the local and global search and prevents the algorithm from searching locally a region where very small improvements are obtained. The parameter ε aims to ensure that a sufficient improvement of f for the hyperrectangle j will be potentially found based on the current $f_{\min}$ [12, 13]. The value of $f_{\min} - \varepsilon|f_{\min}|$ (in contrast to $f_{\min}$) prevents the hyperrectangle with the smallest objective function value from being a POH.

DIRECT can be briefly described by Algorithm 1 [1].

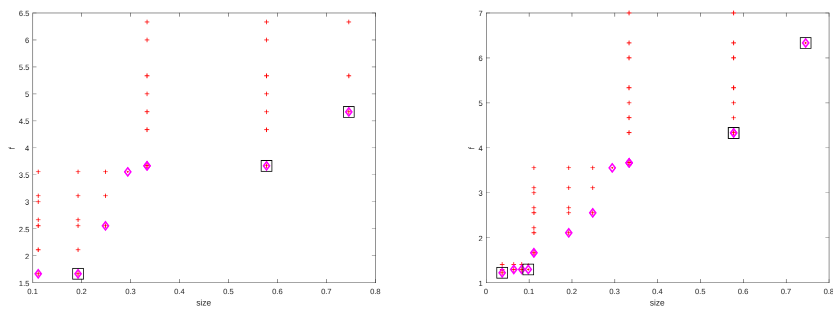
```
Input: f, Ω.
Output: (x_min, f_min).
Normalize Ω to be the unit hypercube and compute f(c) where c is the center;
Set k = 0, f_min = f(c), x_min = c;
while Stopping condition is not satisfied do
    Define the set I_k of the candidate hyperrectangles;
    Identify the set O_k ⊆ I_k of POH;
    while O_k ≠ ∅ do
        Select a hyperrectangle j ∈ O_k;
        Identify the set L_j of dimensions with maximum size δ_max; Set δ = (1/3)δ_max;
        for all i ∈ L_j do
            Sample f at c_j ± δe_i;
            Divide hyperrectangle j into thirds along the dimensions in L_j starting with
              the dimension with lowest w_i = min{f(c_j + δe_i), f(c_j − δe_i)} and
              continue until the dimension with highest w_i;
        end
        Set O_k = O_k \ {j};
    end
    Update f_min = min_{i∈I_k} f(c_i);
    Set x_min = arg min_{i∈I_k} f(c_i);
    Set k = k + 1;
end
```

Algorithm 1: DIRECT algorithm

Identifying the set of POH can be regarded as a problem of finding the extreme points on the lower right convex hull of a set of points in the plane [1]. A 2D-plot can be used to identify the set of POH. The horizontal axis corresponds to the *size* of the hyperrectangle and the vertical axis corresponds to the f value at the center of the hyperrectangle. Figures 1(a) and 1(b) show the center points of the hyperrectangles (marked with 'red' '+' in the plots) generated up to iteration 4 (after 47 function evaluations) and iteration 7 (after 159 function evaluations) respectively, of DIRECT when solving the problem:

$$\min_{\mathbf{x}\in\Omega} \sum_{i=1}^{4} |x_i| + 1 \;, \tag{4}$$

where $\Omega = \{\mathbf{x} \in \mathbb{R}^4 : -2 \leq x_i \leq 3\}$ [14]. The mark that identifies a *candidate* hyperrectangle is a 'magenta' *diamond* and the mark to identify a POH is a 'black' *square*. The identified POH at iteration 4 were divided and generated smaller hyperrectangles. They are no longer hyperrectangles of that size at iteration 7, although other hyperrectangles with the same sizes and higher function values are identified as POH.



(a) (b)

Fig. 1 Points representing hyperrectangles, *candidate* hyperrectangles and POH, when solving the problem (4) by DIRECT. **a** Iteration 4. **b** Iteration 7

3 Two-Phase Heuristic Coupled DIRECT Method

In this section, we reveal how the DIRECT algorithm is modified to incorporate a heuristic that aims to divide a promising search region into three subregions. The implementation of the two-phase strategy aims to drastically reduce the selection of the mostly divided hyperrectangles and, in contrast, select all the hyperrectangles that have the lowest function values in each group of the largest sizes, when a global search phase seems convenient. Conversely, for the local search phase, all the hyperrectangles that have the lowest function values in each group of the smallest sizes are selected and, at the same time, the selection of the largest hyperrectangles are greatly reduced.

3.1 Heuristic

POH either have center points with low function values or are large enough to provide good and unexplored regions for the global search [14]. Hyperrectangles with the smallest *sizes* are the ones that were mostly divided so far. On the other hand, hyperrectangles with large *sizes* were the least divided. Avoiding the identification of POH that were mostly divided can enhance the global search capabilities of DIRECT [7]. Conversely, identifying POH that are close to the hyperrectangle which corresponds to $f_{\min}$ may improve the local search process in DIRECT. Thus, at any iteration k, the present heuristic incorporated into the DIRECT method aims to divide the region of the *candidate* hyperrectangles (the ones with least function values at each *size* group) into three subregions. The leftmost subregion includes hyperrectangles with indices based on *size* that are larger than $i_l = \lfloor 2/3 i_{\min} \rfloor$ (denoting the set by I_k^3), where $i_{\min}$ is the index based on the *size* of the hyperrectangle that corresponds to $f_{\min}$. The rightmost subregion contains the hyperrectangles with indices that are smaller than $i_u = \lfloor 1/3 i_{\min} \rfloor$ (denoting the set by I_k^1). The middle subregion contains hyperrectangles with indices i that satisfy $i_l \leq i \leq i_u$ (denoting the set by I_k^2). (We remark that the larger the *size*, the smaller is the index based on *size*.)

We present in Algorithm 2 the main steps of the proposed heuristic to be integrated into the DIRECT method, coupled with the two-phase strategy (see details in the next subsection).

3.2 Two-Phase Strategy

Since the balance between global and local information must be provided with caution so that convergence to the global solution is guaranteed and stagnation in a local solution is avoided, the two-phase strategy performs a cycling process between a globally biased set of iterations and locally biased iterations. The first phase (identified in the algorithm by 'phase=global') runs for $G_{\max}$ iterations and aims to potentiate the exploration of the hyperrectangles with the largest *sizes*. Here, all *candidate* hyperrectangles with indices based on *size* in I_k^1 are selected. From the middle region, 50% of the indices in the set I_k^2 are randomly selected and the corresponding *candidate* hyperrectangles are used in the selection. From the leftmost subregion, 10% of the indices in the set I_k^3 are randomly selected and the corresponding *candidate* hyperrectangles are selected. Thereafter, the set of POH are identified (following Definition 1) from all these selected hyperrectangles.

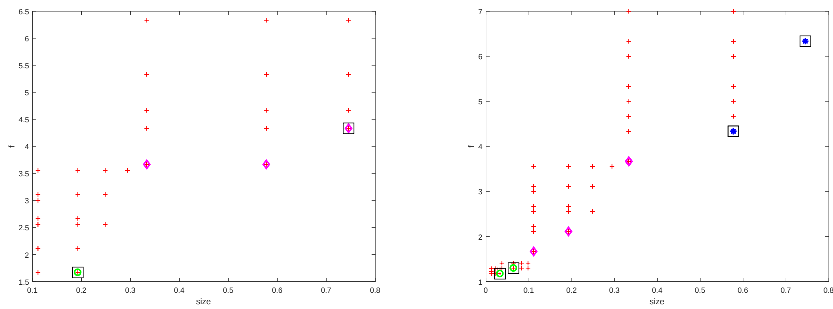
The second phase runs for $L_{\max}$ iterations. Now, all *candidate* hyperrectangles that have indices in the set I_k^3 are selected, 50% of randomly selected indices from I_k^2 are used to choose the corresponding *candidate* hyperrectangles, and 10% of randomly selected indices from I_k^1 are used to pick the corresponding *candidate* hyperrectangles. Then, based on all these selected hyperrectangles, the set of POH are identified. This process is repeated until convergence.

Figures 2(a) and 2(b) show the centers of the hyperretangles generated by Algorithm 2 up to iteration 4 (after 43 function evaluations) and iteration 7 (after 79 function evaluations) respectively, when solving the problem (4). In each plot, the 'green' *circles* correspond to the selected *candidate* hyperrectangles from the set I^3, the 'magenta' *diamonds* correspond to the selected *candidate* hyperrectangles from I^2, and the 'blue' '*' correspond to the selected *candidate* hyperrectangles from

```
Input: f, Ω, G_max, L_max.
Output: (x_min, f_min).
Normalize Ω to be the unit hypercube and compute f(c) where c is the center;
Set k = 0, f_min = f(c), x_min = c; phase=global; k_G = 0, k_L = 0
while Stopping condition is not satisfied do
    Identify the indices based on size i_l = ⌊2/3 i_min⌋ and i_u = ⌊1/3 i_min⌋ and define the sets
      of indices I_k^1, I_k^2, I_k^3 of candidate hyperrectangles;
    if phase=global then
        Set H_k^1 = I_k^1; Randomly select 50% of indices in I_k^2 to define H_k^2; Randomly select
          10% of indices in I_k^3 to define H_k^3;
        Set k_G = k_G + 1;
    else
        Set H_k^3 = I_k^3; Randomly select 50% of indices in I_k^2 to define H_k^2; Randomly select
          10% of indices in I_k^1 to define H_k^1;
        Set k_L = k_L + 1;
    end
    Set H_k = H_k^3 ∪ H_k^2 ∪ H_k^1;
    Identify the set O_k ⊆ H_k of POH;
    while O_k ≠ ∅ do
        Select a hyperrectangle j ∈ O_k;
        Identify the set L_j of dimensions with maximum size δ_max; Set δ = (1/3)δ_max;
        for all i ∈ L_j do
            Sample f at c_j ± δe_i;
            Divide hyperrectangle j into thirds along the dimensions in L_j starting with
              the dimension with lowest w_i = min{f(c_j + δe_i), f(c_j − δe_i)} and
              continue until the dimension with highest w_i;
        end
        Set O_k = O_k \ {j}
    end
    Update f_min = min_{i∈H_k} f(c_i);
    Set x_min = arg min_{i∈H_k} f(c_i);
    if phase=global and k_G ≥ G_max then
        Set phase=local; k_G = 0;
    else
        if phase=local and k_L ≥ L_max then
            Set phase=global; k_L = 0;
        end
    end
    Set k = k + 1;
end
```

Algorithm 2: Two-phase heuristic coupled DIRECT algorithm

I^1. The identified POH are marked with the 'black' *squares*. Comparing with the previous Figs. 1(a) and 1(b) obtained from DIRECT, it may be concluded that the heuristic and the two-phase strategy have reduced the number of selected *candidate* hyperrectangles from which POH are identified, without affecting the convergence to a global solution.



(a) (b)

Fig. 2 Points representing hyperrectangles, selected *candidate* hyperrectangles and POH, when solving the problem (4) by Algorithm 2. **a** Iteration 4. **b** Iteration 7

4 Numerical Experiments

Numerical experiments were carried out to analyze the performance of the presented two-phase heuristic coupled DIRECT method, when compared with other DIRECT-type methods. The MATLAB® (MATLAB is a registered trademark of the MathWorks, Inc.) programming language is used to code the algorithm and the tested problems. The parameter ε is set to $1E-04$. Because there are some elements of randomness in the algorithm, each problem was solved 20 times by the algorithm.

4.1 Termination Based on a Budget

First, we want to analyze what would be the most favorable set of $G_{\max}$ and $L_{\max}$ to be used in the Algorithm 2. The following three sets are tested:

- $G_{\max} = 10$ and $L_{\max} = 10$ giving the Variant V_1;
- $G_{\max} = 10$ and $L_{\max} = 5$ giving the Variant V_2;
- $G_{\max} = 5$ and $L_{\max} = 10$ giving the Variant V_3.

The algorithm runs for a budget of 100 function evaluations. This type of stopping condition is what would be used in practice [4].

The well-known Jones set of benchmark problems [1, 4, 8, 9, 10, 11, 14, 15, 16] is used to compare the above defined three variants of the Algorithm 2. The Jones set contains nine problems: Shekel 5 (S5) with $n = 4$, Shekel 7 (S7) with $n = 4$, Shekel 10 (S10) with $n = 4$, Hartman 3 (H3) with $n = 3$, Hartaman 6 (H6) with $n = 6$, Branin (BR) with $n = 2$, Goldstein and Price (GP) with $n = 2$, Six-Hump Camel (C6) with $n = 2$, Schubert (SHU) with $n = 2$.

Table 1 shows the *perror* value given by

$$perror \equiv \frac{(f_{\min} - f^*)}{|f^*|} , \tag{5}$$

where $f_{\min}$ is the best obtained function value and f^* is the best known global minimum. Our results are compared to those reported in [4]. The *perror* value reported from our algorithm is obtained by using the average value of the solutions $f_{\min}$ obtained over the 20 runs. Although the differences in the performance of the Variants V_1 and V_2 are almost negligible, Variant V_1 is slightly superior, and both outperform the Variant V_3. We may conclude that adopting a larger maximum number of global search iterations gives a better advance in the convergence issue. The comparison with the results in [4] is slightly favorable to the therein locally-biased form of the DIRECT algorithm since it finds slightly better solutions for S5, H3 and H6. However, the results for the remaining six test problems are almost identical to our results.

Table 1 Achieved *perror* for 100 function evaluations, using three variants of Algorithm 2

Problem	Variant V_1 *perror*	Variant V_2 *perror*	Variant V_3 *perror*	DIRECT-l[a] *perror*
S5	$0.12E+00$	$0.17E+00$	$0.21E+00$	$0.59E-02$
S7	$0.58E-02$	$0.58E-02$	$0.62E-01$	$0.58E-02$
S10	$0.57E-02$	$0.57E-02$	$0.81E-01$	$0.41E-02$
H3	$0.66E-03$	$0.62E-03$	$0.77E-03$	$0.85E-04$
H6	$0.13E+00$	$0.13E+00$	$0.13E+00$	$0.23E-01$
BR	$0.16E-03$	$0.19E-03$	$0.20E-03$	$0.39E-03$
GP	$0.27E-03$	$0.27E-03$	$0.14E-02$	$0.27E-03$
C6	$0.10E-01$	$0.11E-01$	$0.63E-02$	$0.16E-01$
SHU	$0.83E+00$	$0.83E+00$	$0.83E+00$	$0.82E+00$

[a] Results (locally-biased form) reported in [4].

4.2 Termination Based on the Known Global Minimum

We now test the Algorithm 2 with a stopping condition that uses the knowledge of the global minimum f^*. The algorithm aims to guarantee a solution as close as possible to the f^*. Thus, the algorithm stops when

$$perror \leq \tau , \tag{6}$$

where *perror* has been defined in (5) and τ is a positive small tolerance. It is assumed that $f^* \neq 0$. However, if condition (6) is not satisfied, the algorithm runs until a specified number of function evaluations is reached. When $f^* = 0$, the *perror* becomes $f_{\min}$.

Based on the previous results, we compare Variant V_1 and Variant V_2 of Algorithm 2 with other DIRECT-type algorithms and some stochastic heuristics. The nine problems of the Jones set are used. Table 2 shows the number of function evaluations required to achieve a solution with accuracy given by $\tau = 1E-04$ and $\tau = 1E-06$, in the context of the stopping condition (6). The results reported from the two variants of Algorithm 2 correspond to the average value of the required number of function evaluations of the 20 runs. The results from the other DIRECT-type algorithms are taken from their original papers [1, 8, 9, 10, 14], unless otherwise stated. The maximum number of function evaluations is set to $1E+05$.

Firstly, we note that using the stopping condition (6) with a higher accuracy demand (0.01% and 0.0001%), the results favor Variant V_2. (This conclusion is different from what would be expected after the comparisons in Table 1.) In fact, when $\tau = 1E-04$, Variant V_2 is better, i.e., reaches the required accuracy with fewer function evaluations than Variant V_1 on 6 problems (out of 9) and is a tie in one problem. When a higher accuracy is demanded ($\tau = 1E-06$), Variant V_2 is still better on 7 problems.

When we compare the results of both variants of Algorithm 2 with DIRECT [1] and the solver RDIRECT-b [9], we may conclude that the results for a 0.01% accuracy is favorable to [9] on four problems, but is favorable to our algorithm on five problems. On the other hand, for a higher accuracy demand (0.0001%), the overall balance is six against three. From the comparison with the original DIRECT, we conclude that our algorithm wins (requires less function evaluations) for a 0.01% accuracy solution on five problems and wins for a 0.0001% accuracy solution on six problems. RDIRECT-b is a robust (insensitive to linear scaling of f) version of DIRECT with a bilevel strategy to accelerate convergence to a higher accurate result. The table also shows the results obtained by DIRECT-GL [8], that includes new strategies for the identification of an extended set of POH, a modified DIRECT version that uses an update to the condition (3) [14], and those reported in [10] of the two versions DISIMPL-V and DISIMPL-C of a DIRECT-like method that uses simplices instead of hyperrectangles. The first evaluates f at 2^n vertices and divides a simplex into two new simplices, the second evaluates f at $n!$ centroid points and divides a simplex into three new simplices. For a 0.01% accuracy solution, we may conclude that our algorithm outperforms DIRECT-GL [8] on seven (out of eight common problems), the modified DIRECT [14] on six (out of nine problems), the DISIMPL-V [10] on eight (out of nine problems), and the DISIMPL-C [10] also on eight problems.

Finally, we compare our results with three stochastic algorithms. In the directed tabu search with the adaptive pattern search in the intensification phase (DTS_{APS}) [15], the average number of function evaluations therein reported are related only to successful trials. For completeness, we also report the corresponding success rates (shown in the table as “% succ”). The other stochastic algorithm used in the comparison is the mutation-based artificial fish swarm algorithm (m-AFSA) [16]. It is a population based algorithm that uses a local search procedure to refine the search around the best point found so far. Another population-based algorithm is selected for the comparison. It uses a stochastic version of the coordinate descent method (St-

Coord_D) [17] and the results are from the variant "hscore_w" with success rates of 100%. We may conclude that both variants of the Algorithm 2 outperform the three selected algorithms. Only for the problem SHU, DTS_{APS} reaches the solution with the required accuracy in fewer function evaluations than our variants.

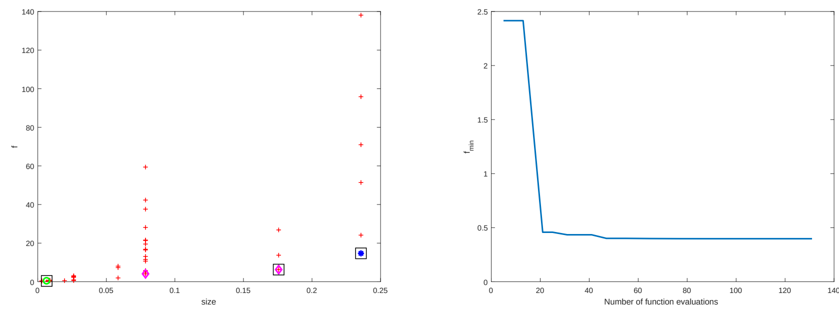
Table 2 Number of function evaluations required by the algorithms, with τ as shown in each row

Algorithm	τ	S5	S7	S10	H3	H6	BR	GP	C6	SHU
Variant V_1	$1E-04$	256	173	171	141	488	145	129	190	2093
	$1E-06$	329	538	580	1140	6908	258	208	362	2684
Variant V_2	$1E-04$	201	170	171	137	454	147	127	179	2409
	$1E-06$	704	430	480	1027	5587	246	209	317	2567
RDIRECT-b[a]	$1E-04$	159	157	157	173	559	181	175	115	3501
	$1E-06$	251	325	325	853	1209	287	373	115	4259
DIRECT[b]	$1E-04$	155	145	145	199	571	195	191	145[c]	2967
	$1E-06$	255	4879	4939	751	182623	377	305	211	3867
DIRECT-GL[d]	$1E-04$	1227	1141	1151	379	4793	333	223	–	425
mDIRECT[e]	$1E-04$	155	145	145	199	571	259	191	285	3663
DISIMPL-V[f]	$1E-04$	2454	723	750	261	6799	242	17	337	4509
DISIMPL-C[f]	$1E-04$	90948	(fail)	(fail)	334	25334	292	180	308	518
DTS_{APS}[g]	$1E-04$	819	812	828	438	1787	212	230	–	274
(% succ)		(75)	(65)	(52)	(100)	(83)	(100)	(100)	–	(92)
m-AFSA[h]	$1E-04$	1183	1103	1586	1891	2580	475	417	247	–
St-Coord_D[i]	$1E-04$	–	–	–	–	–	239	1564	512	–

[a] Results reported in [9]; [b] Results reported in [9], for both values of τ;
[c] Different from result in [1] (285) for $\tau = 1E-04$; [d] Results in [8]; – Not available;
[e] Results in [14] (with a modified update to (3)); [f] Results reported in [10];
[g] Results reported in [15]; [h] Results reported in [16]; [i] Results reported in [17].

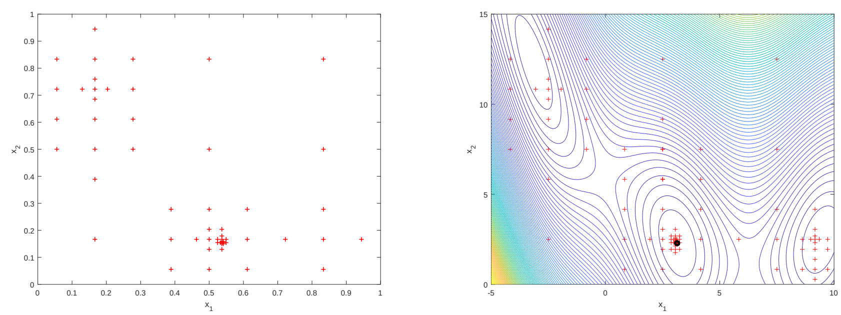
With Fig. 3(a) we aim to illustrate the influence of the heuristic coupled DIRECT on the selected *candidate* hyperrectangles and the POH, at iteration 8 of the global phase, when solving the problem BR, a two-dimensional problem with three global minima. As previously reported the 'green' *circles* correspond to the selected *candidate* hyperrectangles from the set I^3, the 'magenta' *diamonds* are from I^2, and the 'blue' '*' are from I^1. The 'black' *squares* mark the identified POH. Figure 3(b) displays the progress of $f_{\min}$ as the number of function evaluations increases, when solving the problem BR by Algorithm 2 with $G_{\max} = 10$ and $L_{\max} = 10$. The value of $f_{\min}$ rapidly drops (after 20 function evaluations) to a value near the global minimum (0.398).

Figure 4(a) shows the center points of the hyperrectangles generated at iteration 9 when Algorithm 2 uses $G_{\max} = 10$ and $L_{\max} = 10$ (corresponding to the Variant V_1) to solve the problem BR. Figure 4(b) shows the center points at the final iteration where the reported solution is within 0.01% of the global minimum (shown by a 'black' *full circle*). Similar information is shown in Figs. 5(a) and 5(b), but now $G_{\max} = 10$ and $L_{\max} = 5$ (Variant V_2) are used instead. Finally, Figs. 6(a) and 6(b) show the center points of the generated hyperrectangles when $G_{\max} = 5$



(a) (b)

Fig. 3 Solving the problem BR by Algorithm 2. **a** Center points of generated hyperrectangles, selected *candidate* hyperrectangles and identified POH. **b** Progress of $f_{\min}$



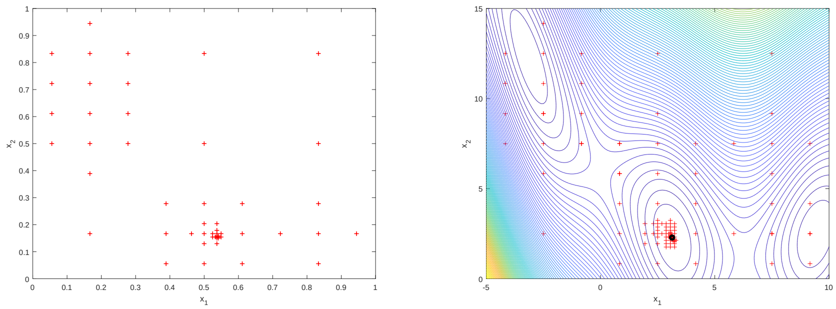
(a) (b)

Fig. 4 Generated hyperrectangles when solving the problem BR by Variant V_1 of Algorithm 2. **a** Iteration 9 (55 function evaluations). **b** Final iteration (131 function evaluations)

and $L_{\max} = 10$ (Variant V_3). It can be seen that the points cluster around the three global solutions, being Variant V_2 the one that concentrates the search the most. After exploring the feasible region looking for promising regions, the Variant V_2 gathers around one of the global solutions instead of jumping and gathering around the other global optima.

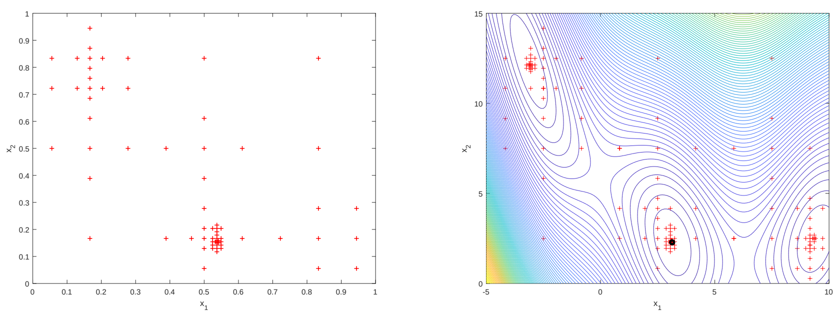
4.3 Experiments with Larger Dimensional Problems

Another set of six larger dimensional benchmark problems from the Hedar test set [18] is also used for comparative purposes: Griewank (GW), Levy (L), Rastrigin (RG), Sphere (S), Sum squares (SS), Trid (T) (also available in [19]). We note that



(a) (b)

Fig. 5 Generated hyperrectangles when solving the problem BR by Variant V_2 of Algorithm 2. **a** Iteration 9 (51 function evaluations). **b** Final iteration (137 function evaluations)



(a) (b)

Fig. 6 Generated hyperrectangles when solving the problem BR by Variant V_3 of Algorithm 2. **a** Iteration 9 (69 function evaluations). **b** Final iteration (163 function evaluations)

the search domain (S. Domain) was modified for some problems in order to avoid that the global minimum lies in the centroid of the feasible region [8, 9].

First, the problem SS is used to analyze the performance of the Variants V_1 and V_2 of the Algorithm 2, when compared to other DIRECT-type methods, as the number of variables increases. The maximum number of function evaluations is now set to $1E+06$ and $\tau = 1E-04$ in the stopping condition (6). See Table 3. The results are compared to those reported in [8], DIRECT, DIRECT-G (DIRECT with a strategy that globally enhances the set of POH), DIRECT-GL (DIRECT with strategies that globally and locally enhance the set of POH). Since numerical data for this problem are not available in [9], a direct comparison is not possible (marked as '–' in the table). (The authors use performance profiles to compare four DIRECT-type methods.) Between Variants V_1 and V_2, the latter is more efficient and from the results it can be concluded that the S. Domain affects the performance of the algorithm, in particular for the largest problem.

Table 3 Number of function evaluations required by Variants V_1 and V_2 to solve problem SS

Problem	S. Domain	Variant V_1	Variant V_2	DIRECT[a]	DIRECT-G[a]	DIRECT-GL[a]
SS $n=2$	$[-10, 15]^n$	84	86	107	143	191
SS $n=5$		2546	1765	833	1951	2919
SS $n=10$		86122	29861	7795	16523	24763
SS $n=2$	$[-8, 12.5]^n$	136	133	–	–	–
SS $n=5$		3209	3135	–	–	–
SS $n=10$		7695	5710	–	–	–

[a] Results reported in [8].

The number of function evaluations achieved by Variants V_1 and V_2 of Algorithm 2 when solving the problems GW, L, RG, S and T for $n = 10$ are shown in Table 4. Between the two tested variants, V_2 outperforms V_1 since it solves the largest problems in general with less function evaluations.

Table 4 Number of function evaluations of Variants V_1 and V_2 (problems with $n = 10$)

Problem	S. Domain	Variant V_1	Variant V_2	DIRECT[a]	DIRECT-G[a]	DIRECT-GL[a]
GW	$[-480, 750]^{10}$	14475	10389	–	–	–
L	$[-10, 10]^{10}$	70437	34067	5589	11149	16179
RG	$[-4.1, 6.4]^{10}$	524921	605391	–	–	–
S	$[-4.1, 6.4]^{10}$	192140	63155	–	–	–
T	$[-100, 100]^{10}$	77653	27075	$> 1E+06$	$> 1E+06$	115073

[a] Results reported in [8]; '–' Not available.

5 Conclusions

The DIRECT method is coupled with a heuristic aiming to divide the region of promising hyperrectangles into three subregions for a discerned selection of a reduced number of hyperrectangles. Furthermore, a two-phase strategy that aims to cyclically encourage the global search capabilities (first phase) and enhance the local search (second phase) is implemented.

During the first phase, the heuristic DIRECT avoids the selection of the hyperrectangles that were mostly divided and chooses all the hyperrectangles with largest sizes. Conversely, during the second phase, the hyperrectangles with largest sizes are mostly avoided and the ones with smallest sizes are all included in the selection. The numerical experiments carried out until now show that a cycle of a global search phase of ten iterations and a local search phase of five iterations provides in general a more efficient process even when solving the largest dimensional problems.

Acknowledgements This work has been supported by European Union's Horizon 2020 research and innovation programme under the Marie Skłodowska-Curie grant agreement No. 734205 - H2020-MSCA-RISE-2017 and by national funds through FCT Fundação para a Ciência e Tecnologia within the Projects Scope: UID/CEC/00319/2019 and UID/MAT/ 00013/2013.

References

1. Jones DR, Perttunen CD, Stuckman BE (1993) Lipschitzian optimization without the Lipschitz constant. J Optimiz Theory App 79(1):157–181
2. Jones DR (2008) Direct global optimization algorithm. In: Floudas C, Pardalos P (eds) Encyclopedia of Optimization, 431–440 (Springer, Boston MA)
3. Liuzzi G, Lucidi S, Piccialli V (2010) A DIRECT-based approach exploiting local minimizations for the solution of large-scale global optimization problems. Comput Optim Appl 45(1):353–375
4. Gablonsky JM, Kelley CT (2001) A locally-biased form of the DIRECT algorithm. J Global Optim 21(1):27-37
5. Liu Q, Zeng J (2015) Global optimization by multilevel partition. J Global Optim 61(1):47–69
6. Sergeyev YD, Kvasov DE (2006) Global search based on efficient diagonal partitions and a set of Lipschitz constants. SIAM J Optimiz 16(3):910–937
7. Paulavičius R, Sergeyev YD, Kvasov DE et al (2014) Globally-biased DISIMPL algorithm for expensive global optimization. J Global Optim 59(2–3):545–567
8. Stripinis L, Paulavičius R, Žilinskas J (2018) Improved scheme for selection of potentially optimal hyper-rectangles in DIRECT. Optim Lett 12(7):1699–1712
9. Liu Q, Cheng W (2014) A modified DIRECT algorithm with bilevel partition. J Global Optim 60(3):483–499
10. Paulavičius R, Žilinskas J (2014) Simplicial Lipschitz optimization without the Lipschitz constant. J Global Optim 59(1):23–40
11. Liu Q (2013) Linear scaling and the DIRECT algorithm. J Global Optim 56(3):1233–1245
12. Liu Q, Zeng J, Yang G (2015) MrDIRECT: a multilevel robust DIRECT algorithm for global optimization problems. J Global Optim 62(2):205–227
13. Liu H, Xu S, Chen X et al (2017) Constrained global optimization via a DIRECT-type constraint-handling technique and an adaptive metamodeling strategy. Struct Multidiscip O 55(1):155–177
14. Finkel DE, Kelley CT (2006) Additive scaling and the DIRECT algorithm. J Global Optim 36(4):597-608
15. Hedar A-R, Fukushima M (2006) Tabu search directed by direct search methods for nonlinear global optimization. Eur J Oper Res 170(2):329–349
16. Rocha AMAC, Costa MFP, Fernandes EMGP (2011) Mutation-based artificial fish swarm algorithm for bound constrained global optimization. AIP Conf Proc 1389:751–754
17. Rocha AMAC, Costa MFP, Fernandes EMGP (2020) A population-based stochastic coordinate descent method. Advances in Intelligent Systems and Computing vol. 991, Le Thi H, Le H, Pham Dinh T (eds) Optimization of Complex Systems: Theory, Models, Algorithms and Applications, 16–25, Springer
18. Hedar A-R `http://www-optima.amp.i.kyoto-u.ac.jp/member/student/hedar/Hedar_files/TestGO.htm`.
19. Ali MM, Khompatraporn C, Zabinsky ZB (2005) A numerical evaluation of several stochastic algorithms on selected continuous global optimization test problems. J Global Optim 31(4):635–672