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

M. Fernanda P. Costa*
Centre of Mathematics
University of Minho, 4710-057 Braga, Portugal
Email:mfc@math.uminho.pt

Edite M. G. P. Fernandes
ALGORITMI Center
University of Minho, 4710-057 Braga, Portugal
Email:emgpf@dps.uminho.pt

Ana Maria A. C. Rocha
ALGORITMI Center
University of Minho, 4710-057 Braga, Portugal
Email:arocha@dps.uminho.pt

Summary

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. Preliminary numerical experiments show that the proposed two-phase heuristic coupled DIRECT method is effective in converging to the optimal solution.

Keywords: *Global optimization, DIRECT algorithm, heuristic, two-phase.*

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:

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

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 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[4]. DIRECT-type algorithms that are more biased toward local search are proposed in[5,6]. 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[7–9].

For further details on the original DIRECT and other recent interesting modifications, we refer the reader to[7–11].

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 Section 3 describes the heuristic and the two-phase strategy in the DIRECT method context. Finally, Section 4 contains the results of our preliminary numerical experiments and we conclude the paper with the Section 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[6,12]. 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*

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

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

where $\boldsymbol{c}_j$ is the center and $\|\boldsymbol{u}^j - \boldsymbol{l}^j\|/2$ is a measure of the size *of the 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 $f_{\min}$[13,14]. 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].

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 and 2 show the points (marked with 'red' '+' in the plots), each one representing the center point of the hyperrectangle, generated at iterations 4 (after 47 function evaluations) and 7 (after 159 function evaluations) respectively, of DIRECT when solving the problem:

$$\min_{\mathbf{x}\in\Omega} \sum_{i=1}^{4} |x_i| + 1 \quad (4)$$

where $\Omega = \{\mathbf{x} \in \mathbb{R}^4 : -2 \leq x_i \leq 3\}$[15]. The mark that identifies a *candidate* hyperrectangle is a 'magenta' circle and the mark to identify a POH is a 'black' square. The identified POH at iteration 4 have been divided and generated smaller hyperrectangles. They are no longer

Input: f, Ω.
Output: $(\mathbf{x}_{\min}, f_{\min})$.
Normalize Ω to be the unit hypercube and compute $f(\mathbf{c})$ where $\mathbf{c}$ is the center;
Set $k = 0$, $f_{\min} = f(\mathbf{c})$, $\mathbf{x}_{\min} = \mathbf{c}$;
while *Stopping condition is not satisfied* **do**
  Define the set I_k of the *candidate* hyperrectangles;
  Identify the set $O_k \subseteq I_k$ of POH;
  while $O_k \neq \emptyset$ **do**
    Select a hyperrectangle $j \in O_k$;
    Identify the set L_j of dimensions with maximum size δ_{max}; Set $\delta = (1/3)\delta_{max}$;
    for *all* $i \in L_j$ **do**
      Sample f at $\mathbf{c}_j \pm \delta \mathbf{e}_i$;
      Divide hyperrectangle j into thirds along the dimensions in L_j starting with the dimension with lowest $w_i = \min\{f(\mathbf{c}_j + \delta \mathbf{e}_i), f(\mathbf{c}_j - \delta \mathbf{e}_i)\}$ and continue until the dimension with highest w_i;
    Set $O_k = O_k \setminus \{j\}$;
  Update $f_{min} = \min_{i \in I_k} f(\mathbf{c}_i)$;
  Set $\mathbf{x}_{\min} = \arg\min_{i \in I_k} f(\mathbf{c}_i)$;
  Set $k = k + 1$;

Algorithm 1: DIRECT algorithm

hyperrectangles of that size at iteration 7, although other hyperrectangles with the same sizes and higher function values are identified as POH.

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[15] . 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[8] . Conversely, identifying POH that are close to

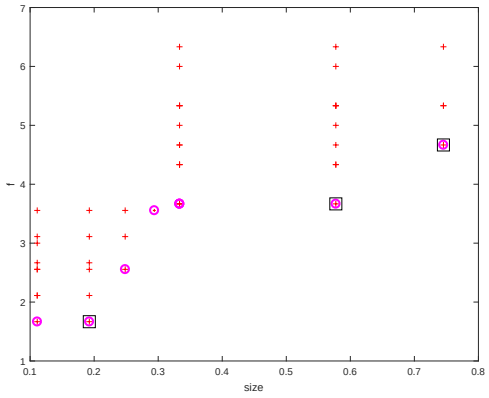

Figure 1: Points representing hyperrectangles, *candidate* hyperrectangles and POH at iteration 4 of DIRECT, when solving the problem (4).

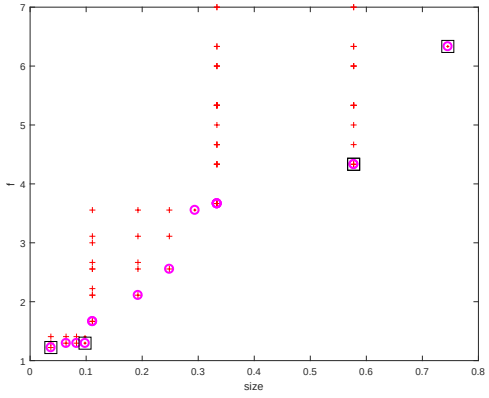

Figure 2: Points representing hyperrectangles, *candidate* hyperrectangles and POH at iteration 7 of DIRECT, when solving the problem (4).

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).

Input: f, Ω, $G_{\max}$, $L_{\max}$.
Output: $(\mathbf{x}_{\min}, f_{\min})$.
Normalize Ω to be the unit hypercube and compute $f(\mathbf{c})$ where $\mathbf{c}$ is the center;
Set $k = 0$, $f_{\min} = f(\mathbf{c})$, $\mathbf{x}_{\min} = \mathbf{c}$; phase=global; $k_G = 0$, $k_L = 0$
while *Stopping condition is not satisfied* **do**
  Identify the indices based on *size* $i_l = \lfloor 2/3 i_{\min} \rfloor$ and $i_u = \lfloor 1/3 i_{\min} \rfloor$ 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 $k_L = k_L + 1$;
  Set $H_k = H_k^3 \cup H_k^2 \cup H_k^1$;
  Identify the set $O_k \subseteq H_k$ of POH;
  while $O_k \neq \emptyset$ **do**
    Select a hyperrectangle $j \in O_k$;
    Identify the set L_j of dimensions with maximum size δ_{max}; Set $\delta = (1/3)\delta_{max}$;
    for *all* $i \in L_j$ **do**
      Sample f at $\mathbf{c}_j \pm \delta \mathbf{e}_i$;
      Divide hyperrectangle j into thirds along the dimensions in L_j starting with the dimension with lowest $w_i = \min\{f(\mathbf{c}_j + \delta \mathbf{e}_i), f(\mathbf{c}_j - \delta \mathbf{e}_i)\}$ and continue until the dimension with highest w_i;
    Set $O_k = O_k \setminus \{j\}$
  Update $f_{min} = \min_{i \in H_k} f(\mathbf{c}_i)$;
  Set $\mathbf{x}_{\min} = \arg\min_{i \in H_k} f(\mathbf{c}_i)$;
  if *phase=global and* $k_G \geq G_{\max}$ **then**
    Set phase=local; $k_G = 0$;
  else
    if *phase=local and* $k_L \geq L_{\max}$ **then**
      Set phase=global; $k_L = 0$;
  Set $k = k + 1$;

Algorithm 2: Two-phase heuristic coupled DIRECT algorithm

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 3 and 4 show the hyperretangles generated by Algorithm 2 at iterations 4 (after 43 function evaluations) and 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' circles correspond to the selected *candidate* hyperrectangles from I^2, and the 'blue' circles correspond to the selected *candidate* hyperrectangles from I^1. The identified POH are marked with the 'black' squares. Comparing with the previous Figures 1 and 2 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.

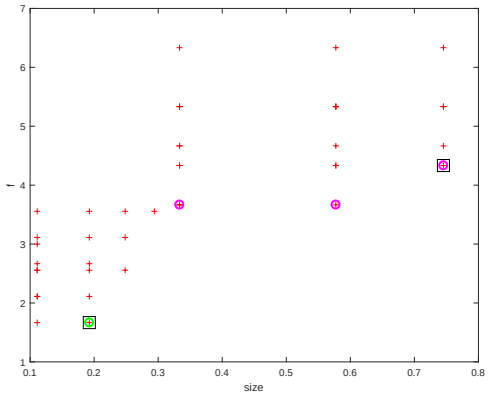

Figure 3: Points representing hyperrectangles, selected *candidate* hyperrectangles and POH at iteration 4 of Algorithm 2, when solving the problem (4).

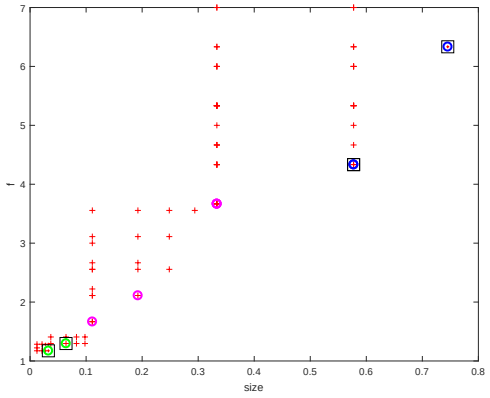

Figure 4: Points representing hyperrectangles, selected *candidate* hyperrectangles and POH at iteration 7 of Algorithm 2, when solving the problem (4).

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 used benchmark problems are the well-known Jones test set (also used in[1,5,10–12,15]). It 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 RCOS (BR) with $n = 2$, Goldstein and Price (GP) with $n = 2$, Six-Hump Camel (C6) with $n = 2$, 2D Schubert (SHU) with $n = 2$.

The Matlab^{TM} (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 is 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[5]. 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[5]. 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, the 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[5] 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* with a budget of 100 function evaluations.

	variant V_1	variant V_2	variant V_3	results in[5]
	perror	*perror*	*perror*	*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	$0.83E+00$	$0.83E+00$	$0.83E+00$	$0.82E+00$

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 $f_{\min} - f^*$.

Based on the previous results, we set now $G_{\max} = 10$, and used $L_{\max} = 10$ or $L_{\max} = 5$. Tables 2 and 3 show 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 from the Algorithm 2 are the average value of the required number of function evaluations of 20 runs. Our results are compared to those reported in[1,10,11,15] and the maximum number of function evaluations is set to $1E+05$.

The first of these two tables compares our results from variant V_1 with DIRECT[1] and the solver RDIRECT-b[10]. 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 comparison

Table 2: Number of function evaluations reached when $G_{max} = 10$ and $L_{max} = 10$, with τ as shown in each column.

	Algorithm 2		results in[10] †		DIRECT §	
	$1E-04$	$1E-06$	$1E-04$	$1E-06$	$1E-04$	$1E-06$
S5	256	329	159	251	155	255
S7	173	538	157	325	145	4879
S10	171	580	157	325	145	4939
H3	141	1140	173	853	199	751
H6	488	6908	559	1209	571	182623
BR	145	258	181	287	195	377
GP	129	208	175	373	191	305
C6	190	362	115	115	145‡	211
SHU	2093	2684	3501	4259	2967	3867

† results from the solver RDIRECT-b.
§ results reported in[10], for both values of τ.
‡ different from result in[1] (285) for $\tau = 1E-04$.

Table 3: Number of function evaluations reached when $G_{max} = 10$ and $L_{max} = 5$, with τ as shown in each column.

	Algorithm 2		results in[15]†	DISIMPL-V§	DISIMPL-C§
	$1E-04$	$1E-06$	$1E-04$	$1E-04$	$1E-04$
S5	201	704	155	2454	90948
S7	170	430	145	723	(fail)
S10	171	480	145	750	(fail)
H3	137	1027	199	261	334
H6	454	5587	571	6799	25334
BR	147	246	259	242	292
GP	127	209	191	17	180
C6	179	317	285	337	308
SHU	2409	2567	3663	4509	518

† modified DIRECT based on an update to (3).
§ results reported in[11].

with the results for a 0.01% accuracy is favorable to[10] on four problems, but is favorable to our Algorithm 2 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.

In the second of these two tables, we also report the results of our variant V_2, since the stopping condition (6) with a higher accuracy demand (0.01% and 0.0001%) provided results different from what would be expected after the comparisons in Table 1. In fact, when $\tau = 1E-04$, V_2 is better – reaches the required accuracy with fewer function evaluations – than V_1 on 6 problems (out of 9) and is a tie in one problem. When a higher accuracy is demanded ($\tau = 1E-06$), V_2 is still better on 7 problems. Table 3 also shows the results obtained by a modified DIRECT version that uses an update to the condition (3)[15], and those reported in[11] 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. From the results in Table 3, for a 0.01% accuracy solution, we may conclude that our algorithm outperforms the modified DIRECT[15] on six problems, the DISIMPL-V[11] on eight problems, and the DISIMPL-C[11] also on eight problems.

With the next figure – Figure 5 – 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' circles are from I^2, and the 'blue' circles are from I^1. The 'black' squares mark the identified POH.

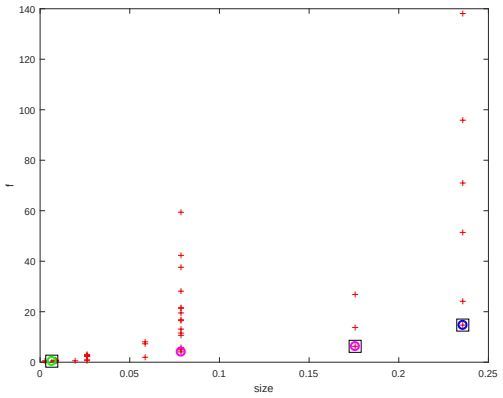

Figure 5: Center points of generated hyperrectangles, selected *candidate* hyperrectangles and identified POH at iteration 8 of Algorithm 2, when solving the problem BR.

In Figure 6 we can see 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 7 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 8 shows the center points at the final iteration where the reported solution is within 0.01% of the global minimum. Figures 9 and 10 display similar information, but when $G_{max} = 10$ and $L_{max} = 5$ (variant V_2) are used instead. Finally, Figures 11 and 12 show the center points of the generated hyperrectangles when $G_{max} = 5$ 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

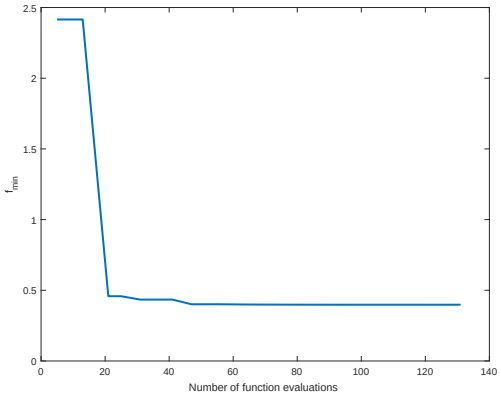

Figure 6: Progress of $f_{\min}$ when solving the problem BR by Algorithm 2.

regions, the variant V_2 gathers around one of the global solutions instead of jumping and gathering around the other global optima.

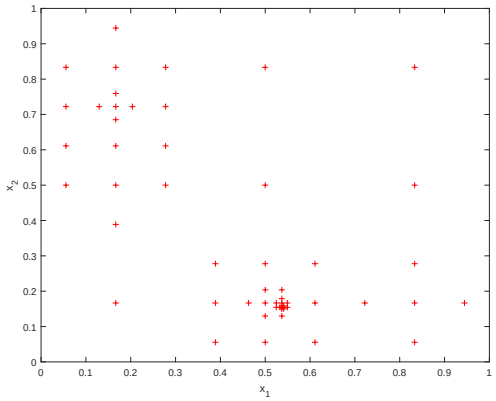

Figure 7: Generated hyperrectangles at iteration 9 (55 function evaluations) – variant V_1.

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 preliminary numerical experiments show that a cycle of a global search phase of ten iterations and a local search

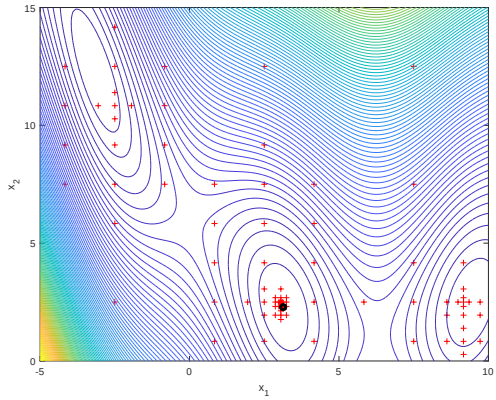

Figure 8: Generated hyperrectangles at final iteration (131 function evaluations) – variant V_1.

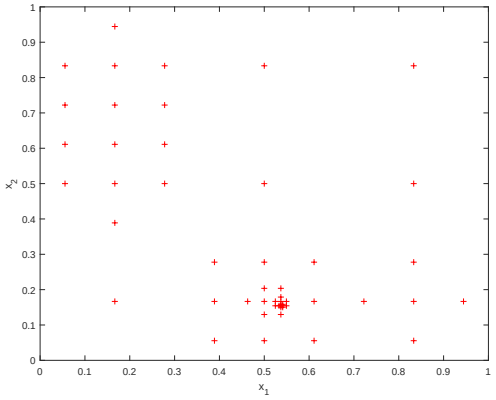

Figure 9: Generated hyperrectangles at iteration 9 (51 function evaluations) – variant V_2.

phase of five iterations provides in general a more efficient process.

Acknowledgments

This work has been supported by FCT – Fundação para a Ciência e Tecnologia within the Projects Scope: UID/CEC/00319/2019 and UID/MAT/ 00013/2013.

References

[1] Jones, D.R., Perttunen, C.D., and Stuckman, B.E. Lipschitzian optimization without the Lipschitz constant. *Journal of Optimization Theory and Applications* **79**(1), 157–181 (1993).

[2] Jones, D.R. Direct global optimization algorithm. In Encyclopedia of Optimization, Floudas, C. and Pardalos, P., editors, pp. 431–440 (Springer, Boston MA, 2008).

[3] Liuzzi, G., Lucidi, S., and Piccialli, V. A DIRECT-based approach exploiting local

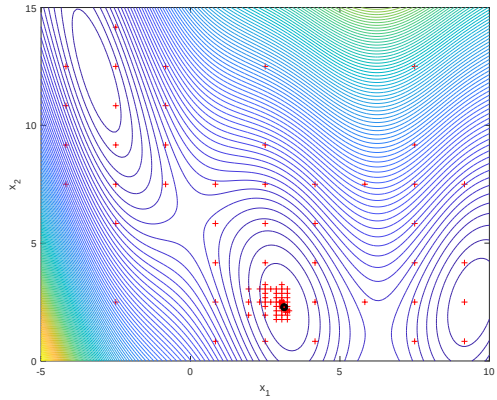

Figure 10: Generated hyperrectangles at final iteration (137 function evaluations) – variant V_2.

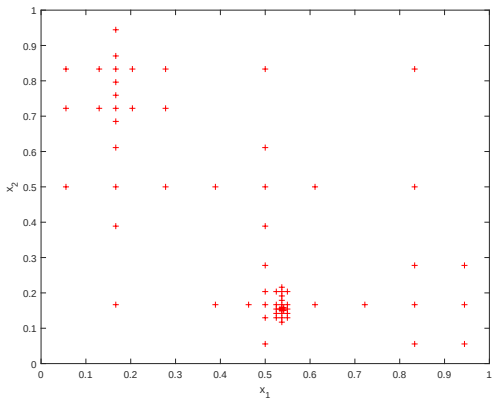

Figure 11: Generated hyperrectangles at iteration 9 (69 function evaluations) – variant V_3.

minimizations for the solution of large-scale global optimization problems. *Computational Optimization and Applications* **45**(1), 353–375 (2010).

[4] Wu, Y., Ozdamar, L., and Kumar, A. TRIOPT: a triangular-based partitioning algorithm for global optimization. *Journal of Computational and Applied Mathematics* **177**(1), 35–53 (2005).

[5] Gablonsky, J.M. and Kelley, C.T. A locally-biased form of the DIRECT algorithm. *Journal of Global Optimization* **21**(1), 27—37 (2001).

[6] Liu, Q. and Zeng, J. Global optimization by multilevel partition. *Journal of Global Optimization* **61**(1), 47–69 (2015).

[7] Sergeyev, Y.D. and Kvasov, D.E. Global search based on efficient diagonal partitions and a set of Lipschitz constants. *SIAM Journal on Optimization* **16**(3), 910–937 (2006).

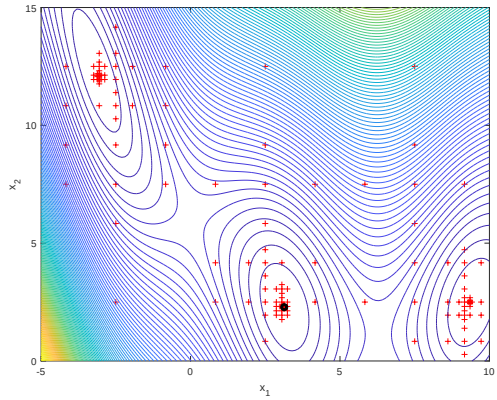

Figure 12: Generated hyperrectangles at final iteration (163 function evaluations) – variant V_3.

[8] Paulavičius, R., Sergeyev, Y.D., Kvasov, D.E., and Žilinskas, J. Globally-biased DISIMPL algorithm for expensive global optimization. *Journal of Global Optimization* **59**(2–3), 545–567 (2014).

[9] Stripinis, L., Paulavičius, R., and Žilinskas, J. Improved scheme for selection of potentially optimal hyper-rectangles in DIRECT. *Optimization Letters* **12**(7), 1699–1712 (2018).

[10] Liu, Q. and Cheng, W. A modified DIRECT algorithm with bilevel partition. *Journal of Global Optimization* **60**(3), 483–499 (2014).

[11] Paulavičius, R. and Žilinskas, J. Simplicial Lipschitz optimization without the Lipschitz constant. *Journal of Global Optimization* **59**(1), 23–40 (2014).

[12] Liu, Q. Linear scaling and the DIRECT algorithm. *Journal of Global Optimization* **56**(3), 1233–1245 (2013).

[13] Liu, Q., Zeng, J., and Yang, G. MrDIRECT: a multilevel robust DIRECT algorithm for global optimization problems. *Journal of Global Optimization* **62**(2), 205–227 (2015).

[14] Liu, H., Xu, S., Chen, X., Wang, X., and Ma, Q. Constrained global optimization via a DIRECT-type constraint-handling technique and an adaptive metamodeling strategy. *Structural and Multidisciplinary Optimization* **55**(1), 155–177 (2017).

[15] Finkel, D.E. and Kelley, C.T. Additive scaling and the DIRECT algorithm. *Journal of Global Optimization* **36**(4), 597-–608 (2006).