Vilnius University
Institute of Mathematics and Informatics
LITHUANIA

INFORMATICS (09 P)

IMPROVEMENT, DEVELOPMENT AND IMPLEMENTATION OF DERIVATIVE-FREE GLOBAL OPTIMIZATION ALGORITHMS

Linas Stripinis

October 2017

Technical Report MII-DS-09P-17-1 October 2016 - 30 September 2020

VU Institute of Mathematics and Informatics, Akademijos str. 4, Vilnius LT-08663, Lithuania
www.mii.lt

Abstract

We consider a box-constrained global optimization problem with a Lipschitz-continuous objective function and an unknown Lipschitz constant. The well known derivative-free global-search `DIRECT` (`DI`vide a hyper-`RECT`angle) algorithm performs well solving such problems. However, the efficiency of the `DIRECT` algorithm deteriorates on problems with many local optima and when the solution with high accuracy is required. To overcome these difficulties different regimes of global and local search are introduced or the algorithm is combined with local optimization. In first part we investigate a different direction of improvement of the `DIRECT` algorithm and propose a new strategy for the selection of potentially optimal rectangles, what does not require any additional parameters or local search subroutines. An extensive experimental investigation reveals the effectiveness of the proposed enhancements.

Keywords: Global optimization, `DIRECT`-type algorithms, Derivative-free optimization

Contents

1 Introduction

We consider a box-constrained global optimization problem of the form

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

where $f : \mathbb{R}^n \to \mathbb{R}$ denotes the objective function and the feasible region is an n-dimensional hyper-rectangle $D = [\mathbf{a}, \mathbf{b}] = \{\mathbf{x} \in \mathbb{R}^n : a_j \leq x_j \leq b_j, j = 1, \dots, n\}$. We also assume, that the objective function $f(\mathbf{x})$ is Lipschitz-continuous, but can be non-linear, non-differentiable, non-convex, and multi-modal. `DIRECT` is a popular partitioning-based Lipschitz optimization [HPT95,PŽ07,PŽ09,PŽ14,PŽG10,Pin96,SS00] algorithm extending ideas of Piyavskii [Piy67] (independently rediscovered also by Shubert [Shu72]) algorithm to multidimensional derivative-free optimization. The `DIRECT` algorithm [JPS93] seeks a global optimum by partitioning potentially optimal (the most promising) hyper-rectangles and evaluating the objective function at the centers of these hyper-rectangles. Simplicity and efficiency of the `DIRECT` algorithm attracted considerable research interest. Although most of `DIRECT`-type algorithms use hyper-rectangular partitions [GK01,LC14,LZY15,LLP10a,LLP10b], simplicial partitions (`DISIMPL` algorithm) [PSKŽ14, PŽ13, PŽ14] have several advantages [PŽ16]. Central sampling of the objective function can be changed to diagonal approach sampling at the endpoints of diagonal [KPS03, SK06, SK08, SK17]. A trisection of hyper-rectangles is usually used to reuse the objective function values at the center or endpoints of diagonals in descendant subregions. However, a bisection can ensure better shapes of hyper-rectangles with a smaller variety of sizes in different dimensions than trisection which produces sizes differing by three times, but a specific sampling strategy is necessary to enable the reuse of sample points [PCŽ16].

2 Improved scheme for selection of potentially optimal hyper-rectangles in `DIRECT`

The original `DIRECT` algorithm has two main weaknesses [LYZZ17,LLP16,PSKŽ14,SK06]. First, on problems with many local minima, `DIRECT` sometimes spends an excessive number of function evaluations exploring suboptimal local minima, thereby delaying the discovery of the global minimum. To address this issue, a two-phase globally-biased technique was proposed [PSKŽ14, SK06]. Second, `DIRECT` usually gets close to the global optimum quickly, but it can be slow to converge with a high accuracy. To overcome the latter issue, a two-phase locally-biased technique [LZY15] or hybrid versions of `DIRECT`-type algorithms enriched with the use of local searches [LLP10a,LLP16] can be employed. In this section, we propose an alternative strategy to overcome both drawbacks without the need to use local solvers or use two-phase scheme which requires the introduction of new parameters.

2.1 The selection of the most promising hyper-rectangles

The essential step in `DIRECT`-type algorithms is identification of potentially optimal (the most promising) hyper-rectangles of the current partition, which at the iteration k is defined as

$$\mathcal{P}_k = \{D_k^i : i \in \mathbb{I}_k\},$$

where $D_k^i = [\mathbf{a}^i, \mathbf{b}^i] = \{\mathbf{x} \in \mathbb{R}^n : 0 \leq a_j^i \leq x_j \leq b_j^i \leq 1, j = 1, \dots, n, \forall i \in \mathbb{I}_k\}$ and $\mathbb{I}_k$ is the index set identifying the current partition $\mathcal{P}_k$. The next partition $\mathcal{P}_{k+1}$ is obtained after the subdivision of the selected potentially optimal hyper-rectangles from the current partition $\mathcal{P}_k$.

2.1.1 Potentially optimal hyper-rectangles in the original `DIRECT` algorithm

To make the selection of potentially optimal hyper-rectangles in the future iterations, `DIRECT` assesses the goodness based on the lower bound estimates for the objective function $f(\mathbf{x})$ over each hyper-rectangle D_k^i. The requirement of potential optimality is stated formally in Definition 1.

Definition 1 (Potentially optimal hyper-rectangle) *Let $\mathbf{c}^i$ denote the center sampling point and δ_i be a measure (distance, size) of the hyper-rectangle D_k^i. Let $\varepsilon > 0$ be a positive constant and $f_{\min}$ be the best currently known value of the objective function. A hyper-rectangle $D_k^j, j \in \mathbb{I}_k$ is said to be potentially optimal if there exists some rate-of-change (Lipschitz) constant $\tilde{L} > 0$ such that*

$$f(\mathbf{c}^j) - \tilde{L}\delta_j \leq f(\mathbf{c}^i) - \tilde{L}\delta_i, \quad \forall i \in \mathbb{I}_k, \tag{2}$$

$$f(\mathbf{c}^j) - \tilde{L}\delta_j \leq f_{\min} - \varepsilon |f_{\min}|, \tag{3}$$

where the measure of the hyper-rectangle is

$$\delta_i = \frac{1}{2} \|\mathbf{b}^i - \mathbf{a}^i\|_2. \tag{4}$$

The hyper-rectangle D_k^j is potentially optimal if the lower Lipschitz bound for the objective function computed by the left-hand side of (2) is the smallest one with some positive constant $\tilde{L}$ among the hyper-rectangles of the current partition $\mathcal{P}_k$. In (3) the parameter ε is used to protect from an excessive refinement of the local minima [JPS93, PSKŽ14].

2.1.2 Selection of the most promising hyper-rectangles in other DIRECT-type algorithms

In the original DIRECT algorithm, the size of a hyper-rectangle is measured by the Euclidean distance from its center to a corner or equivalently by a half length of a diagonal (see (4)). In DIRECT-l [GK01], the measure of a hyper-rectangle is instead evaluated by the length of its longest side. Such a measure corresponds to the L^∞-norm and allows the DIRECT-l algorithm to group more hyper-rectangles with the same measure. Thus, there are fewer distinct measures and therefore, less potentially optimal hyper-rectangles are selected. Moreover, in DIRECT-l at most one hyper-rectangle from each group is selected, even if there are more than one potentially optimal hyper-rectangle in the same group. This allows reduction of the number of divisions within a group. The results presented in [GK01] and extended in [PSKŽ14] suggest that DIRECT-l performs well for lower dimensional problems, which do not have too many local and global minima.

The main principle of an aggressive version of DIRECT [BWG+00] is to select and divide a hyper-rectangle of every measure (δ_i) in each iteration. The aggressive version requires many more function evaluations than the other versions of DIRECT since the criteria for choosing hyper-rectangles to be divided have been relaxed. Although this approach does not appear to be favorable for simple test problems, more difficult problems may be easier solved by this strategy on a large parallel supercomputer [BWG+00].

In the PLOR algorithm [MPR+17], the set of all Lipschitz constants (herewith the set of potentially optimal hyper-rectangles) is reduced to just two: the maximal and the minimal ones. In such a way the PLOR approach is independent of any user-defined parameters and balances equally local and global search during the optimization process.

A two-phase globally [PSKŽ14, SK06] and locally-biased [LZY15] algorithms at one of the phases work in the same as the original DIRECT algorithm, i.e., during the selection procedure considers all hyper-rectangles from the current partition. However, in the second phase, they limit the selection of potentially optimal hyper-rectangles based on their measures. The globally-biased versions constrain themselves to the larger subregions (primary addressing the first weakness), while the locally-biased version constrains itself to the smaller ones and in such a way addresses the second weakness of DIRECT-type algorithms.

2.2 Extended set of potentially optimal hyper-rectangles

In this section, we present a new way to identify the extended set of potentially optimal hyper-rectangles. Using a new two-step based strategy, we enlarge the set of the best hyper-rectangles by adding more medium-measured hyper-rectangles with the smallest function value at their centers and additionally, closest to the current minimum point. The first extension forces the algorithm to work more globally (compared to the selection procedure used in DIRECT), while the second part assures faster and broader examination around the current minimum point. In such way, we address both weaknesses of `DIRECT` staying in the same algorithmic framework. Let's state it formally.

Let $\mathbb{L}_k$ be the set of all different indices at the current partition $\mathcal{P}_k$, corresponding to the groups of hyper-rectangles having the same measure (δ_k). The minimum value $l_k^{\min} \in \mathbb{L}_k$ corresponds to the group of hyper-rectangles having the smallest measure $\delta_k^{\min}$. The maximum value $l_k^{\max}$ of $\mathbb{L}_k$ corresponds to the group of hyper-rectangles having the largest measures $\delta_k^{\max}$, i. e., $l_k^{\max} = \max\{\mathbb{L}_k\} < \infty$. Finally, let $l_k^i \in \mathbb{L}_k$ be the index of the group the hyper-rectangle D_k^i belongs to. Having this, in Definitions 2 and 3 we formalize new strategies for identification of an extended set of potentially optimal hyper-rectangles from the current partition $\mathcal{P}_k$.

Definition 2 (Enhancing the global search)

- ***Step 1*** *Find an index $j \in \mathbb{I}_k$ and a corresponding hyper-rectangle D_k^j, such that*

$$D_k^j = \arg\max_j \{l_k^j : j = \arg\min_{i \in \mathbb{I}_k:\, l_k^{\min} \le l_k^i \le l_k^{\max}} \{f(\mathbf{c}^i)\}\}. \tag{5}$$

- ***Step 2*** *Set $l_k^{\min} = l_k^j + 1$. If $l_k^j \le l_k^{\max}$* ***repeat*** *from Step 1; otherwise* ***terminate****.*

At Step 1, the hyper-rectangle containing the minimum point $(\mathbf{x}^{\min})$ is selected. If there are several hyper-rectangles with the same lowest objective value $f(\mathbf{c}^i)$, the preference is given to hyper-rectangles with the largest l_k^j value, i.e., a bigger size measure. After this, in Step 2, the minimum value $l_k^{\min} = l_k^j + 1$ is increased; thus all hyper-rectangles from the groups with indices lower than the updated $l_k^{\min}$ (measures of these hyper-rectangles belonging to these groups are smaller than the measure of the $l_k^{\min}$ group) are not considered in the recurrent Step 1. A geometrical interpretation and comparison of the original `DIRECT` and the globally enhanced (let us call `DIRECT-G`) versions are shown in the left-hand side and middle graphs in Figure 1. By this strategy, we extend the number of medium-measured potentially optimal hyper-rectangles and force `DIRECT-G` to work more globally. Let us stress, that opposed to the aggressive `DIRECT` version, by Definition 2 `DIRECT-G` will not consider hyper-rectangles from the groups where the minimum function value is larger compared to the minimum value from the larger groups.

Definition 3 (Enhancing the local search)

- ***Step 1*** *At each iteration k, evaluate the Euclidean distance from the current minimum point $(\mathbf{x}^{\min})$ to other sampled points:*

$$d(\mathbf{x}^{\min}, \mathbf{c}^i) = \sqrt{\sum_{j=1}^{n} (x_j^{\min} - c_j^i)^2} \tag{6}$$

- ***Step 2*** *Apply the procedure described in Definition 2 in (5) using distances $d(\mathbf{x}^{\min}, \mathbf{c}^i)$ instead of objective function values.*

A geometrical interpretation of the selection of potentially optimal hyper-rectangles using the locally enhanced strategy is shown on the right-hand side of Figure 1. By this strategy, we extend the number of potentially optimal hyper-rectangles locating close to the current minimum point ($\mathbf{x}^{\min}$). Moreover, by this strategy, we select the closest hyper-rectangles from various measures.

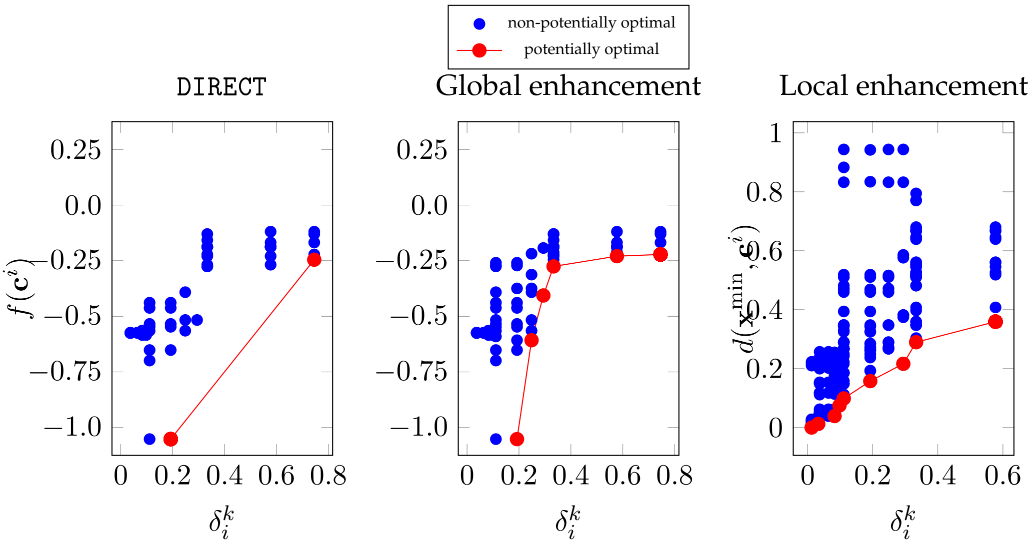

Figure 1: Geometric interpretation of the selection of potentially optimal hyper-rectangles by using `DIRECT` (on the left-hand side), `DIRECT-G` (middle), and the locally enhanced strategy (on the right-hand side) on the Shekel 5 test problem in the fifth iteration of corresponding algorithms/strategies

2.2.1 `DIRECT-GL` algorithm

In this subsection, we introduce a new `DIRECT`-type algorithm (let us call `DIRECT-GL`). The key feature of `DIRECT-GL` is that `DIRECT-GL` performs the identification of potentially-optimal hyper-rectangles twice in every iteration. First, by using Definition 2 the globally enhanced set of potentially optimal candidates is determined and fully processed (sampled and partitioned). Second, by using Definition 3 the locally enhanced set is identified and fully processed (sampled and partitioned) again. Thus, our new approach is based on "Divide the best" strategy [Ser98] and it has the everywhere-dense type of convergence (like other `DIRECT`-type algorithms [FK06, JPS93, PCŽ16, PSKŽ14, SK06]). This follows from the fact that, that using Definitions 2 and 3, `DIRECT-GL` always selects for partitioning hyper-rectangles from the group ($l_k^{\max}$) with the largest measure $\delta_k^{\max}$. Since each group contains only a finite number of hyper-rectangles, after a sufficient number of iterations, all hyper-rectangles will be partitioned. Such a procedure will be repeated with a new group of the largest hyper-rectangles and so on until the largest hyper-rectangles will have the measure smaller than the required tolerance ε.

The complete description of the `DIRECT-GL` algorithm is shown in Algorithm 1. The input for the algorithm is one (or few) stopping criteria: required tolerance (ε_{pe}), the maximal number of function evaluations ($\mathtt{M}_{\text{max}}$) and the maximal number of `DIRECT-GL` iterations ($\mathtt{K}_{\text{max}}$). After termination, `DIRECT-GL` returns the found objective value f_{min} and the solution point $\mathbf{x}^{\text{min}}$ together with algorithmic performance measures: final tolerance – percent error (pe), the number of function evaluations (m), and the number of iterations (k).

input : ε_{pe}, $\mathtt{M}_{\text{max}}$, $\mathtt{K}_{\text{max}}$;
output: f_{min}, $\mathbf{x}^{\text{min}}$;

1 Initialize $k = 1, m = 1, \mathbb{I}_{\text{k}} = \{1\}, f_{\text{min}} = f(\mathbf{c}^1), \mathbf{x}^{\text{min}} = \mathbf{c}^1$;
2 **while** $pe > \varepsilon_{\text{pe}}$ **and** $m < \text{M}_{\text{max}}$ **and** $k < \text{K}_{\text{max}}$ **do**   `// pe defined in Eq. (7)`
3   Identify the index set $\mathbb{J}_k^1 \subseteq \mathbb{I}_k$ of potentially optimal hyper-rectangles using Definition 2;
4   Set $\mathbf{x}^{\text{min}}_{\text{old}} = \mathbf{x}^{\text{min}}$;
5   **foreach** $i \in \mathbb{J}_k^1$ **do**
6   Subdivide (trisect) hyper-rectangle D_k^i and update $\mathbb{I}_{\text{k}}$;
7   Evaluate f at the centers of the new hyper-rectangles;
8   Update f_{min}, $\mathbf{x}^{\text{min}}$, pe and m;
9   **end**
10   **if** $\mathbf{x}^{\text{min}} \neq \mathbf{x}^{\text{min}}_{\text{old}}$ **then**
11   Calculate distances $\mathbf{d}(\mathbf{x}^{\text{min}}, \mathbf{c}^{\text{i}}),\ i \in \mathbb{I}_{\text{k}}$ to all sampled points;   `// using Eq. (6)`
12   Set $\mathbf{x}^{\text{min}}_{\text{old}} = \mathbf{x}^{\text{min}}$;
13   **else**
14   Calculate distances $\mathbf{d}(\mathbf{x}^{\text{min}}, \mathbf{c}^{\text{i}})$ to newly sampled points;
15   **end**
16   Identify the index set $\mathbb{J}_{\text{k}}^2 \subseteq \mathbb{I}_k$ of potentially optimal hyper-rectangles using Definition 3;
17   **foreach** $i \in \mathbb{J}_k^2$ **do**
18   Subdivide (trisect) hyper-rectangle D_k^i and update $\mathbb{I}_{\text{k}}$;
19   Evaluate f at the centers of the new hyper-rectangles;
20   Update f_{min}, $\mathbf{x}^{\text{min}}$, pe and m;
21   **end**
22   Increase $k = k + 1$ and check **if** condition described in lines 10-15;
23 **end**
24 **return** f_{min}, $\mathbf{x}^{\text{min}}$, pe, k, m;

Algorithm 1: Pseudo code of the `DIRECT-GL` algorithm

2.2.2 Numerical investigation

The introduced `DIRECT-G` and `DIRECT-GL` as well as the original `DIRECT` algorithm (Finkel's implementation [Fin04]) were implemented in the `MATLAB` programming language. Note, that for the `DIRECT` algorithm potentially optimal hyper-rectangles can be identified in at least two different ways: using modified Graham's scan algorithm [BH99] or the rule described by Lemma 2.3 in [Gab01]. Usually this does not impose significant differences, but occasionally it can have, e.g., when a higher precision is required. The selection procedure of potentially optimal hyper-rectangles in `DIRECT-GL` differs significantly, however, this does not have a notable difference to the overall performance, compared with the procedure used in DIRECT. This means, that for the identification of the same quantity of potentially optimal hyper-rectangles DIRECT and DIRECT-GL spent a similar amount of time.

We compare the efficiency of the algorithms on the Hedar test set [Hed05], which consist of 27 global optimization test functions. Some of test problems have several variants, e.g., Bohachevsky, Hartman, Shekel, and some of them can be tested for different dimensionality. In Table 1 we report main features of these problems: problem number (No.), name, dimensionality (n), feasible region (D), the number of local minima (if known), and the known minimum (f^*). Whenever the global minimum point lies at the initial sampling point for any tested algorithm the feasible region was modified (increased). These modified problems are marked with the star sign *.

Note, that the most of test problems from the Hedar test set are multimodal, therefore suitable to investigate how introduced modifications help to overcome the first weakness. Since all the global minima f^* are known for all Hedar test problems in advance, investigated algorithms were stopped either when the point $\bar{\mathbf{x}}$ was generated such that the percent error

$$pe = 100\% \times \begin{cases} \frac{f(\bar{\mathbf{x}})-f^*}{|f^*|}, & f^* \neq 0, \\ f(\bar{\mathbf{x}}), & f^* = 0, \end{cases} \tag{7}$$

is smaller than the tolerance value $\varepsilon_{\rm pe}$, or when the number of function evaluations exceeds the prescribed limit of 10^6. In our investigation, four different values for $\varepsilon_{\rm pe}$ were considered: 10^{-2}, 10^{-4}, 10^{-6}, 10^{-8}. By doing this, we investigate algorithm's ability to avoid the second weakness. The comparison is based on the number of function evaluations and the best (smallest) number for each problem is shown in bold font.

The results of experiments are given in Table 2. First, observe that DIRECT-G and DIRECT-GL perform on average much better (see **Aver. (overall)**) compared to DIRECT. Especially this is evident when a lower percentage error (pe) (higher accuracy) is sought. Observe, that original DIRECT on average performs better only for simpler (unimodal) test problems (see **Aver. (unimodal)**). That is mainly because the set of potentially optimal hyper-rectangles in DIRECT-G, DIRECT-L and DIRECT-GL is larger per iteration. Consequently, a greater number of function evaluations is needed.

For small dimensional problems (see **Aver.** $(n \leq 3)$), DIRECT requires on average from 4.5 times (when $\varepsilon_{\rm pe} = 10^{-2}$) to 175 times more function evaluations (when $\varepsilon_{\rm pe} = 10^{-8}$) compared to DIRECT-GL. Also DIRECT-L showed an advantage comparing with DIRECT. Observe, that DIRECT-G performed worst with $\varepsilon_{\rm pe} = 10^{-2}$ and $\varepsilon_{\rm pe} = 10^{-4}$. Again, for most of these problems DIRECT was able to converge after a small number of iterations. Therefore, by extending the set of potentially optimal hyper-rectangles only globally enhanced (DIRECT-G) is not very efficient for low-dimensional problems. However, when $\varepsilon_{\rm pe} = 10^{-6}$ and $\varepsilon_{\rm pe} = 10^{-8}$ was used, DIRECT-G performed significantly better compared to DIRECT.

For higher dimensional (see **Aver.** $(n \geq 4)$) and multimodal problems (see **Aver.**

Table 1: Key characteristics of the Hedar test problems

Problem No.	Problem name	Dimension n	Feasible region D	No. of local minima	Optimum f^*
$1, 2, 3$	Ackley*	2, 5, 10	$[-15, 35]^n$	multimodal	0.0
4	Beale	2	$[-4.5, 4.5]^2$	multimodal	0.0
5	Bohachevsky 1*	2	$[-100, 110]^2$	multimodal	0.0
6	Bohachevsky 2*	2	$[-100, 110]^2$	multimodal	0.0
7	Bohachevsky 3*	2	$[-100, 110]^2$	multimodal	0.0
8	Booth	2	$[-10, 10]^2$	unimodal	0.0
9	Branin	2	$[-5, 10] \times [10, 15]$	3	0.39789
10	Colville	4	$[-10, 10]^4$	multimodal	0.0
$11, 12, 13$	Dixon & Price	2, 5, 10	$[-10, 10]^n$	unimodal	0.0
14	Easom	2	$[-100, 100]^2$	multimodal	-1.0
15	Goldstein & Price	2	$[-2, 2]^2$	4	3.0
16	Griewank*	2	$[-600, 700]^2$	multimodal	0.0
17	Hartman	3	$[0, 1]^3$	4	-3.86278
18	Hartman	6	$[0, 1]^6$	4	-3.32237
19	Hump	2	$[-5, 5]^2$	6	-1.03163
$20, 21, 22$	Levy	2, 5, 10	$[-10, 10]^n$	multimodal	0.0
23	Matyas*	2	$[-10, 15]^2$	unimodal	0.0
24	Michalewicz	2	$[0, \pi]^2$	2!	-1.80130
25	Michalewicz	5	$[0, \pi]^5$	5!	-4.68765
26	Michalewicz	10	$[0, \pi]^{10}$	10!	-9.66015
27	Perm	4	$[-4, 4]^4$	multimodal	0.0
$28, 29$	Powell	4, 8	$[-4, 5]^n$	multimodal	0.0
30	Power Sum	4	$[0, 4]^4$	multimodal	0.0
$31, 32, 33$	Rastrigin*	2, 5, 10	$[-5.12, 6.12]^n$	multimodal	0.0
$34, 35, 36$	Rosenbrock	2, 5, 10	$[-5, 10]^n$	unimodal	0.0
$37, 38, 39$	Schwefel	2, 5, 10	$[-500, 500]^n$	unimodal	0.0
40	Shekel, $m = 5$	4	$[0, 10]^4$	5	-10.15320
41	Shekel, $m = 7$	4	$[0, 10]^4$	7	-10.40294
42	Shekel, $m = 10$	4	$[0, 10]^4$	10	-10.53641
43	Shubert	2	$[-10, 10]^2$	760	-186.73091
$44, 45, 46$	Sphere*	2, 5, 10	$[-5.12, 6.12]^n$	multimodal	0.0
$47, 48, 49$	Sum squares*	2, 5, 10	$[-10, 15]^n$	unimodal	0.0
50	Trid	6	$[-36, 36]^6$	multimodal	-50.0
51	Trid	10	$[-100, 100]^{10}$	multimodal	-210.0
$52, 53, 54$	Zakharov*	2, 5, 10	$[-5, 11]^n$	multimodal	0.0

(multimodal)) both introduced versions performed significantly better compared to `DIRECT`, and the best results were obtained using `DIRECT-GL`. Finally, in total `DIRECT` failed for 30.1% (65/216) cases, most of which when a lower percent error tolerance was required (10^{-6} and 10^{-8}) and optimization problems were more challenging. Meanwhile, `DIRECT-G`, `DIRECT-L` and `DIRECT-GL` in total failed on 18.1% (39/216),24% (52/216) and 9.2% (20/216) cases, accordingly.

Table 2: Number of function evaluations using `DIRECT`, `DIRECT-G` and `DIRECT-GL` algorithms solving Hedar test problems

Problem	`DIRECT`				`DIRECT-G`				`DIRECT-L`				`DIRECT-GL`			
No./$\varepsilon_{\rm pe}$	10^{-2}	10^{-4}	10^{-6}	10^{-8}	10^{-2}	10^{-4}	10^{-6}	10^{-8}	10^{-2}	10^{-4}	10^{-6}	10^{-8}	10^{-2}	10^{-4}	10^{-6}	10^{-8}
1	**225**	**443**	**655**	**909**	773	1,385	2,301	3,463	751	1,343	2,239	3,377	1,197	2,123	3,571	5,415
2	**8,845**	**11,289**	**14,619**	**17,757**	10,611	19,137	31,459	47,065	138,165	146,359	158,897	174,231	19,403	35,175	55,843	84,979
3	**80,927**	$> 10^6$	$> 10^6$	$> 10^6$	90,089	**151,575**	**240,677**	**350,075**	$> 10^6$	$> 10^6$	$> 10^6$	$> 10^6$	180,707	306,089	486,459	702,121
4	655	1,143	1,823	2,835	283	591	891	1,347	357	721	1,119	1,615	**183**	**395**	**591**	**833**
5	**327**	**457**	**551**	**845**	435	607	739	1,129	435	611	743	1,133	729	847	1,115	1,767
6	**345**	**489**	**589**	**897**	441	617	749	1,139	855	1,025	1,155	1,545	727	845	1,113	1,765
7	693	1,073	1,645	2,099	623	935	1,407	2,057	**459**	**787**	**1,119**	**1,595**	685	1,113	1,665	2,139
8	295	511	917	1,295	301	489	901	1,221	**283**	**395**	**699**	**1,015**	345	509	831	1,087
9	**195**	377	38,455	$> 10^6$	255	**365**	**603**	**841**	333	457	755	1,079	333	579	859	1,239
10	6,585	18,261	24,485	67,695	104,315	120,077	128,847	162,751	9,465	18,915	21,405	23,197	**1,623**	**2,809**	**3,539**	**5,371**
11	481	597	1,143	1,969	403	477	973	1,489	373	537	971	1,349	**235**	**393**	**823**	**1,297**
12	18,237	19,407	23,065	32,229	14,531	17,135	23,955	**29,471**	213,759	215,109	221,133	230,409	**13,109**	**16,501**	**22,951**	31,213
13	**365,221**	**458,743**	$> 10^6$	$> 10^6$	990,493	$> 10^6$	$> 10^6$	$> 10^6$	$> 10^6$	$> 10^6$	$> 10^6$	$> 10^6$	$> 10^6$	$> 10^6$	$> 10^6$	$> 10^6$
14	32,859	59,347	297,571	$> 10^6$	336,879	337,069	337,169	337,477	**377**	**623**	**741**	**1,097**	495	817	1,085	1,679
15	**191**	**305**	10,437	$> 10^6$	209	357	**553**	789	269	415	603	839	223	367	555	**789**
16	9,215	9,341	9,341	9,505	12,519	12,711	12,711	12,965	**1,753**	**1,965**	**1,965**	**2,249**	2,067	2,375	2,375	2,799
17	**199**	4,165	88,883	$> 10^6$	369	**669**	819	**1,493**	325	**621**	931	1,623	379	1,049	1,199	2,431
18	**571**	182,623	$> 10^6$	$> 10^6$	1,529	**4,063**	**6,903**	**12,163**	1,557	4,249	7,027	12,237	4,793	8,793	13,207	19,879
19	293	997	54,487	$> 10^6$	**211**	**355**	593	965	**211**	359	**555**	**927**	279	485	657	1,143
20	**127**	**155**	**267**	**401**	189	225	407	585	149	221	399	577	189	263	459	581
21	**705**	**1,021**	**1,921**	**2,845**	1,587	2,563	4,325	6,253	1,533	2,485	4,193	6,101	2,349	4,361	6,329	10,149
22	**5,589**	**10,431**	**18,475**	**28,461**	11,149	18,801	30,673	44,013	10,303	17,555	28,761	41,505	16,179	29,945	48,049	74,815
23	107	209	391	935	111	225	379	825	**65**	**179**	**281**	**477**	101	211	357	557
24	**67**	**109**	**109**	**109**	97	179	179	179	97	179	179	179	129	235	235	235
25	14,077	215,127	$> 10^6$	$> 10^6$	5,491	7,105	7,819	7,819	$> 10^6$	$> 10^6$	$> 10^6$	$> 10^6$	**2,445**	**4,619**	**5,575**	**5,575**
26	$> 10^6$	$> 10^6$	$> 10^6$	$> 10^6$	$> 10^6$	$> 10^6$	$> 10^6$	$> 10^6$	$> 10^6$	$> 10^6$	$> 10^6$	$> 10^6$	**601,433**	**608,113**	**611,077**	**611,077**
27	$> 10^6$	$> 10^6$	$> 10^6$	$> 10^6$	$> 10^6$	$> 10^6$	$> 10^6$	$> 10^6$	$> 10^6$	$> 10^6$	$> 10^6$	$> 10^6$	$> 10^6$	$> 10^6$	$> 10^6$	$> 10^6$
28	13,675	67,515	309,427	$> 10^6$	11,589	50,149	320,073	$> 10^6$	**5,135**	34,179	321,343	$> 10^6$	7,045	**24,591**	**85,235**	**202,795**
29	$> 10^6$	$> 10^6$	$> 10^6$	$> 10^6$	$> 10^6$	$> 10^6$	$> 10^6$	$> 10^6$	364,693	$> 10^6$	$> 10^6$	$> 10^6$	**147,105**	**905,027**	$> 10^6$	$> 10^6$
30	$> 10^6$	$> 10^6$	$> 10^6$	$> 10^6$	$> 10^6$	$> 10^6$	$> 10^6$	$> 10^6$	**13,243**	$> 10^6$	$> 10^6$	$> 10^6$	101,181	**763,635**	$> 10^6$	$> 10^6$
31	987	1,181	1,565	1,833	2,897	3,087	3,333	3,631	24,883	25,053	25,327	25,533	**811**	**1,109**	**1,507**	**1,803**
32	$> 10^6$	$> 10^6$	$> 10^6$	$> 10^6$	$> 10^6$	$> 10^6$	$> 10^6$	$> 10^6$	$> 10^6$	$> 10^6$	$> 10^6$	$> 10^6$	**180,429**	**184,247**	**192,151**	**196,343**
33	$> 10^6$	$> 10^6$	$> 10^6$	$> 10^6$	$> 10^6$	$> 10^6$	$> 10^6$	$> 10^6$	$> 10^6$	$> 10^6$	$> 10^6$	$> 10^6$	$> 10^6$	$> 10^6$	$> 10^6$	$> 10^6$

Continued on next page

MII-DS-09P-17-1 October 2016 - 30 September 2020

Table 2 Continued from previous page

Problem	DIRECT				DIRECT-G				DIRECT-L				DIRECT-GL			
No./$\varepsilon_{\mathrm{pe}}$	10^{-2}	10^{-4}	10^{-6}	10^{-8}	10^{-2}	10^{-4}	10^{-6}	10^{-8}	10^{-2}	10^{-4}	10^{-6}	10^{-8}	10^{-2}	10^{-4}	10^{-6}	10^{-8}
34	1,621	1,913	3,005	4,019	389	619	2,285	3,883	**313**	**471**	**679**	**1,471**	579	727	1,143	1,657
35	**19,693**	**24,681**	**35,575**	**41,687**	20,363	28,293	46,005	68,065	$> 10^6$	$> 10^6$	$> 10^6$	$> 10^6$	25,395	38,633	72,735	86,043
36	169,191	215,435	267,741	308,715	**53,193**	**83,559**	**146,087**	**273,021**	$> 10^6$	$> 10^6$	$> 10^6$	$> 10^6$	95,405	167,319	268,591	403,207
37	**255**	**447**	**597**	**1,195**	371	567	691	1,153	807	989	1,105	1,555	659	971	1,235	1,709
38	**27,543**	**30,307**	**31,199**	39,487	637,379	640,081	640,743	645,519	$> 10^6$	$> 10^6$	$> 10^6$	$> 10^6$	556,495	561,599	562,903	568,483
39	$> 10^6$	$> 10^6$	$> 10^6$	$> 10^6$	$> 10^6$	$> 10^6$	$> 10^6$	$> 10^6$	$> 10^6$	$> 10^6$	$> 10^6$	$> 10^6$	$> 10^6$	$> 10^6$	$> 10^6$	$> 10^6$
40	**155**	**255**	$> 10^6$	$> 10^6$	781	1,419	2,477	3,803	731	1,365	**2,389**	**3,697**	1,227	2,025	3,433	5,209
41	**145**	4,875	$> 10^6$	$> 10^6$	755	2,017	3,737	5,377	697	**1,953**	**3,645**	**5,273**	1,141	2,845	4,741	6,623
42	**145**	4,939	$> 10^6$	$> 10^6$	715	1,977	3,493	5,111	709	**1,949**	**3,443**	**5,047**	1,151	2,871	4,789	7,137
43	2,967	3,867	68,667	$> 10^6$	4,089	4,219	4,393	4,603	**369**	**535**	**807**	**1,079**	425	735	951	1,341
44	209	417	633	1,211	191	337	481	785	**173**	**309**	**449**	**743**	391	549	737	1,103
45	4,653	10,583	20,123	44,099	**2,287**	4,113	6,335	10,933	2,573	**3,963**	**6,103**	**10,175**	4,357	8,249	11,011	18,225
46	99,123	205,013	614,749	$> 10^6$	**16,857**	**28,243**	47,529	76,723	20,115	28,727	**46,803**	**75,211**	35,721	63,399	94,991	155,511
47	**107**	**195**	**321**	**623**	143	251	391	705	143	251	391	567	191	337	525	759
48	**833**	**1,489**	**2,463**	**3,827**	1,951	3,271	5,267	7,745	1,857	3,165	5,153	7,237	2,919	4,701	7,523	11,031
49	**7,795**	**14,691**	**22,651**	**34,735**	16,523	24,489	37,645	53,647	13,563	22,427	34,919	48,637	24,763	41,781	63,413	89,543
50	**4,897**	207,399	$> 10^6$	$> 10^6$	5,077	**10,069**	**17,411**	**26,079**	12,149	23,015	42,051	60,457	7,795	15,735	26,059	38,929
51	66,615	$> 10^6$	$> 10^6$	$> 10^6$	**22,201**	251,255	$> 10^6$	$> 10^6$	261,301	608,797	742,935	$> 10^6$	36,525	**119,093**	**174,059**	**299,163**
52	**237**	**303**	**653**	**949**	295	329	709	1,023	249	281	605	779	345	413	889	1,123
53	$> 10^6$	$> 10^6$	$> 10^6$	$> 10^6$	377,737	602,319	613,251	$> 10^6$	**5,465**	**9,725**	**15,591**	**22,243**	6,429	9,967	17,665	23,891
54	$> 10^6$	$> 10^6$	$> 10^6$	$> 10^6$	$> 10^6$	$> 10^6$	$> 10^6$	$> 10^6$	**94,175**	**151,287**	**268,999**	**317,611**	115,073	184,033	320,267	394,467
Aver. (overall)	184,591	236,891	369,800	493,577	199,253	211,822	235,896	263,322	226,023	265,436	277,382	298,068	**114,887**	**150,622**	**170,131**	**186,799**
Aver. (unimodal)	**115,099**	**126,330**	**170,648**	**176,480**	195,439	199,961	207,523	220,482	373,655	374,537	376,095	378,051	194,300	202,406	214,502	228,328
Aver. (multimodal)	208,913	275,588	439,503	604,561	200,588	215,973	245,826	278,316	174,351	227,251	242,832	270,074	**87,092**	**132,498**	**154,601**	**172,263**
Aver. ($n \leq 3$)	2,290	3,828	25,335	262,245	15,760	15,942	16,246	16,685	1,480	1,666	1,905	2,278	**509**	**759**	**1,064**	**1,533**
Aver. ($n \geq 4$)	319,846	409,809	625,371	665,211	335,394	357,192	398,862	446,311	392,619	461,136	481,767	517,525	**199,748**	**261,812**	**295,568**	**324,254**
Failed	9	11	18	26	8	9	10	12	11	13	13	15	**4**	**4**	**6**	**6**

Concluded

3 Conclusions

We introduced a new strategy for the selection of the extended set of potentially optimal hyper-rectangles in the DIRECT-type algorithmic framework. Using the proposed approach two well-known weaknesses of DIRECT-type algorithms were addressed. The experimental results confirmed the well-known fact that while for simpler problems DIRECT performs well, for more challenging (higher dimensional) and multimodal problems the proposed modified DIRECT-GL performs significantly faster. Moreover, since the set of potentially optimal hyper-rectangles is larger (compared to DIRECT), DIRECT-GL scheme looks promising for more efficient parallelization too.

References

[BH99] Mattias Björkman and Kenneth Holmström. Global optimization using the DIRECT algorithm in Matlab. *Advanced Modeling and Optimization*, 1(2):17–37, 1999.

[BWG+00] C. A. Baker, L. T. Watson, B. Grossman, W. H. Mason, and R. T. Haftka. Parallel global aircraft configuration design space exploration. In A. Tentner, editor, *High Performance Computing Symposium 2000*, pages 54–66. Soc. for Computer Simulation Internat, 2000.

[Fin04] D. E. Finkel. MATLAB source code for DIRECT. http://www4.ncsu.edu/~ctk/Finkel_Direct/, 2004. Online; accessed: 2017-03-22.

[FK06] D. E. Finkel and C. T. Kelley. Additive scaling and the DIRECT algorithm. *Journal of Global Optimization*, 36(4):597–608, 2006.

[Gab01] J. M. Gablonsky. *Modifications of the* DIRECT *Algorithm*. PhD thesis, North Carolina State University, 2001.

[GK01] J. M. Gablonsky and C. T. Kelley. A locally-biased form of the DIRECT algorithm. *Journal of Global Optimization*, 21(1):27–37, 2001.

[Hed05] A. Hedar. Test functions for unconstrained global optimization. http://www-optima.amp.i.kyoto-u.ac.jp/member/student/hedar/Hedar_files/TestGO.htm, 2005. Online; accessed: 2017-03-22.

[HPT95] R. Horst, P. M. Pardalos, and N. V. Thoai. *Introduction to Global Optimization*. Nonconvex Optimization and Its Application. Kluwer Academic Publishers, 1995.

[JPS93] D. R. Jones, C. D. Perttunen, and B. E. Stuckman. Lipschitzian optimization without the Lipschitz constant. *Journal of Optimization Theory and Application*, 79(1):157–181, 1993.

[KPS03] D. E. Kvasov, C. Pizzuti, and Ya. D. Sergeyev. Local tuning and partition strategies for diagonal GO methods. *Numerische Mathematik*, 94(1):93–106, 2003.

[LC14] Qunfeng Liu and Wanyou Cheng. A modified DIRECT algorithm with bilevel partition. *Journal of Global Optimization*, 60(3):483–499, 2014.

[LLP10a] Giampaolo Liuzzi, Stefano Lucidi, and Veronica Piccialli. A DIRECT-based approach exploiting local minimizations for the solution for large-scale global optimization problems. *Computational Optimization and Applications*, 45(2):353–375, 2010.

[LLP10b] Giampaolo Liuzzi, Stefano Lucidi, and Veronica Piccialli. A partition-based global optimization algorithm. *Journal of Global Optimization*, 48(1):113–128, 2010.

[LLP16] Giampaolo Liuzzi, Stefano Lucidi, and Veronica Piccialli. Exploiting derivative-free local searches in DIRECT-type algorithms for global optimization. *Computational Optimization and Applications*, 65:449–475, 2016.

[LYZZ17] Qunfeng Liu, Guang Yang, Zhongzhi Zhang, and Jinping Zeng. Improving the convergence rate of the DIRECT global optimization algorithm. *Journal of Global Optimization*, 67(4):851–872, 2017.

[LZY15] Qunfeng Liu, Jinping Zeng, and Gang Yang. MrDIRECT: a multilevel robust DIRECT algorithm for global optimization problems. *Journal of Global Optimization*, 62(2):205–227, 2015.

[MPR+17] Jonas Mockus, Remigijus Paulavičius, Dainius Rusakevičius, Dmitrij Šešok, and Julius Žilinskas. Application of Reduced-set Pareto-Lipschitzian Optimization to truss optimization. *Journal of Global Optimization*, 67(1-2):425–450, 2017.

[PCŽ16] Remigijus Paulavičius, Lakhdar Chiter, and Julius Žilinskas. Global optimization based on bisection of rectangles, function values at diagonals, and a set of Lipschitz constants. *Journal of Global Optimization*, (1):1–17, 2016.

[Pin96] J. D. Pintér. *Global Optimization in Action (Continuous and Lipschitz Optimization: Algorithms, Implementations and Applications)*. Kluwer Academic Publishers, Dordrecht, 1996.

[Piy67] S. A. Piyavskii. An algorithm for finding the absolute minimum of a function. *Theory of Optimal Solutions*, 2:13–24, 1967. in Russian.

[PSKŽ14] Remigijus Paulavičius, Yaroslav D. Sergeyev, Dmitri E. Kvasov, and Julius Žilinskas. Globally-biased DISIMPL algorithm for expensive global optimization. *Journal of Global Optimization*, 59(2-3):545–567, 2014.

[PŽ07] R. Paulavičius and J. Žilinskas. Analysis of different norms and corresponding Lipschitz constants for global optimization in multidimensional case. *Information Technology and Control*, 36(4):383–387, 2007.

[PŽ09] Remigijus Paulavičius and Julius Žilinskas. Global optimization using the branch-and-bound algorithm with a combination of Lipschitz bounds over simplices. *Technological and Economic Development of Economy*, 15(2):310–325, 2009.

[PŽ13] Remigijus Paulavičius and Julius Žilinskas. Simplicial Lipschitz optimization without the Lipschitz constant. *Journal of Global Optimization*, 59(1):23–40, 2013.

[PŽ14] Remigijus Paulavičius and Julius Žilinskas. *Simplicial Global Optimization*. SpringerBriefs in Optimization. Springer New York, New York, NY, 2014.

[PŽ16] Remigijus Paulavičius and Julius Žilinskas. Advantages of simplicial partitioning for Lipschitz optimization problems with linear constraints. *Optimization Letters*, 10(2):237–246, 2016.

[PŽG10] Remigijus Paulavičius, Julius Žilinskas, and Andreas Grothey. Investigation of selection strategies in branch and bound algorithm with simplicial partitions and combination of Lipschitz bounds. *Optimization Letters*, 4(2):173–183, 2010.

[Ser98] Yaroslav D Sergeyev. On convergence of "divide the best" global optimization algorithms. *Optimization*, 44(3):303–325, 1998.

[Shu72] B. O. Shubert. A sequential method seeking the global maximum of a function. *SIAM Journal on Numerical Analysis*, 9:379–388, 1972.

[SK06] Yaroslav D. Sergeyev and Dmitri E. Kvasov. Global search based on diagonal partitions and a set of Lipschitz constants. *SIAM Journal on Optimization*, 16(3):910–937, 2006.

[SK08] Ya. D. Sergeyev and D. E. Kvasov. *Diagonal Global Optimization Methods*. FizMatLit, Moscow, 2008. In Russian.

[SK17] Yaroslav D Sergeyev and Dmitri E Kvasov. *Deterministic Global Optimization: An Introduction to the Diagonal Approach*. SpringerBriefs in Optimization. Springer, 2017.

[SS00] R. G. Strongin and Ya. D. Sergeyev. *Global Optimization with Non-Convex Constraints: Sequential and Parallel Algorithms*. Kluwer Academic Publishers, Dordrecht, 2000.