

BBOB-Benchmarking the DIRECT Global Optimization Algorithm

Petr Pošík

Czech Technical University in Prague, Faculty of Electrical Engineering, Dept. of Cybernetics
Technická 2, 166 27 Prague 6
posik@labe.felk.cvut.cz

ABSTRACT

The DIRECT global optimization algorithm is tested on the BBOB 2009 testbed. The algorithm is rather time and space consuming since it does not forget any point it samples during the optimization. Furthermore, all the sampled points are considered when deciding where to sample next. The results suggest that the algorithm is a viable alternative only for low-dimensional search spaces (5D at most).

Categories and Subject Descriptors

G.1.6 [Numerical Analysis]: Optimization—*Global Optimization, Unconstrained Optimization*; F.2.1 [Analysis of Algorithms and Problem Complexity]: Numerical Algorithms and Problems

General Terms

Algorithms, Experimentation, Performance, Reliability

Keywords

Benchmarking, Black-box optimization, Evolutionary computation, Global optimization, DIRECT

1. INTRODUCTION

The DIRECT algorithm was introduced in [5]. The name of the algorithm not only expresses that it belongs to the class of direct search algorithms, it also describes the basic principle of the algorithm: the DIRECT acronym stands for *Dividing RECTangles*.

2. ALGORITHM DESCRIPTION

In this paper, a MATLAB implementation [2, 6] of the DIRECT algorithm is used. Only the basic algorithm design principles are described here; for the detailed description see [5] or [2].

The algorithm divides the search space to non-overlapping hyperrectangles; in each time instant, the whole search space

is completely covered by all the hyperrectangles. The point in the middle of each hyperrectangle (the base point) is evaluated. Each hyperrectangle thus has two (among others) important characteristics:

1. the fitness of its base point, and
2. the size of the hyperrectangle.

There are many possible definitions of the hyperrectangle size, here the distance from the basepoint to the hyperrectangle corner is used.

In each iteration the algorithm decides which of the existing hyperrectangles should be split, i.e. where the next points should be sampled. The decision is based on the following 2 assumptions:

- the better fitness of the base point, the higher the chance of finding an improvement, and
- the larger the rectangle, the higher the chance of finding an improvement.

The algorithm thus selects for division¹ large rectangles and rectangles with highly fit base points, while it ignores small rectangles, or rectangles with non-fit base points.

The division process then proceeds by splitting the rectangle into thirds by evaluating $2D$ points (D is the search space dimensionality) lying in the distance $\pm \frac{1}{3}$ of the respective side length of the hyperrectangle from the base point, and deciding the order in which the individual dimensions should be divided into thirds. The demonstration of the DIRECT sampling process can be seen in Fig. 1.

2.1 Implementation Modifications

The MATLAB implementation [2] was modified to better suit the needs of the BBOB-benchmarking.

1. The original implementation allowed the user to use it in the test mode when the value of global optimum is known. However, in that case all other termination criteria were ignored, i.e. the algorithm runs until it finds the global optimum. The modification takes into account also other termination criteria, the maximum number of evaluations in particular.

¹The selection of the next rectangles to divide is basically a multi-objective problem. The algorithm does not use the usual definition of non-dominated solutions; instead it selects (the right subset of) the rectangles that lie on the boundary of the convex hull of all rectangles described by their size and the base point fitness.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

GECCO '09, July 8–12, 2009, Montréal Québec, Canada.
Copyright 2009 ACM 978-1-60558-505-5/09/07 ...\$5.00.

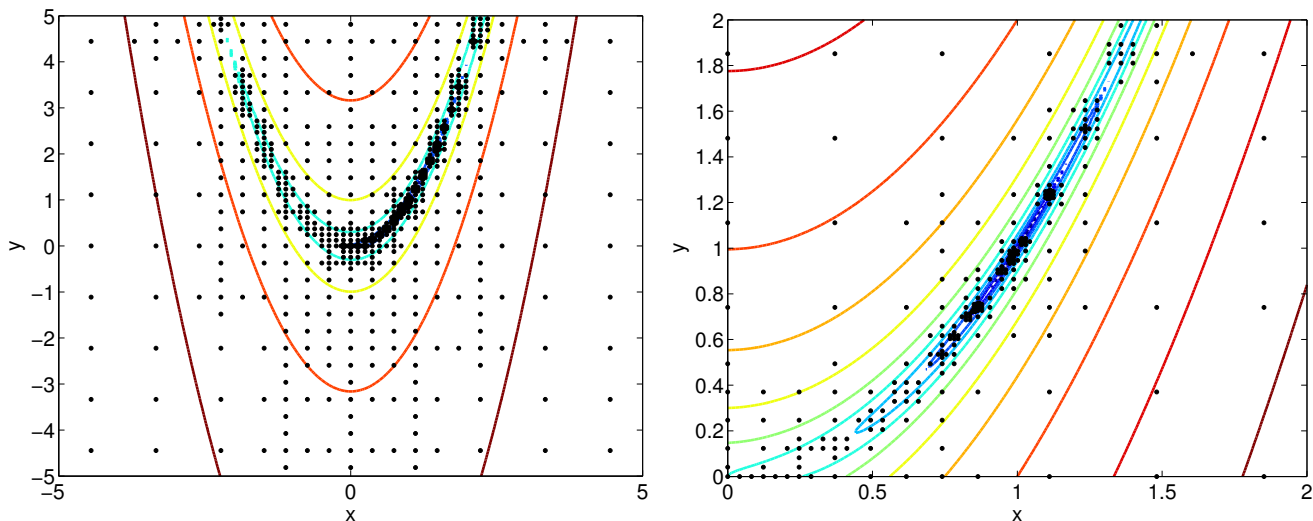


Figure 1: DIRECT algorithm optimizing 2D Rosenbrock’s function using 1000 samples. Left: the overall picture, interval $(-5, 5)^2$, right: the detail of interval $(0, 2)^2$

2. The original MATLAB implementation used function calls that work on shared data structures. Since MATLAB does not allow passing arguments to functions by reference and always passes them by value, this resulted in creating unnecessary temporary copies of the working data structures. The modified implementation now uses a pseudo-object-oriented approach, so that the data structures are really shared and no copies of data structures are created, which speeds up the algorithm.

2.2 Parameter Settings

The DIRECT algorithm uses several parameters which must be set:

- The Jones factor ϵ is the minimal amount of improvement which is considered to be significant by the algorithm. The value is $\epsilon = 10^{-10}$.
- The `maxdeep` argument determines the possible depth of the division tree; the parameter is roughly equivalent to setting the minimal allowed distance between two neighboring sampled points. It was set to 21, so that the minimal distance is of order 10^{-10} .

2.3 Box Constraints? Yes

The algorithm is not suitable for unconstrained optimization, it needs the bounding hypercube. In the BBOB experimentation documentation [3], it is stated that all the functions have the global optimum in $(-5, 5)^D$ hypercube (D is the search space dimensionality), thus it may seem that this hypercube can be a good choice for the box constraints for DIRECT. However, several of the benchmark functions have the global optimum near (or directly on) the search space boundary and DIRECT is pretty bad in approaching such solutions. Consequently, the DIRECT box constraints were chosen as $(-6, 6)^D$ hypercube. It was observed that this choice improved the results e.g. for the Linear slope function, while it did not worsen the results on the rest of the functions.

2.4 The Crafting Effort

The same parameter settings were used for all experiments on all functions, the crafting effort $\text{CrE} = 0$.

2.5 Invariance Properties

The algorithm searches a form of rectangular grid, it is thus *not invariant with respect to translation and rotation*. When deciding where to sample the next points, the DIRECT algorithm makes use of the fitness values in such a way that it is *not invariant with respect to order-preserving transformations of the fitness function*.

3. EXPERIMENTAL PROCEDURE

The standard experimental procedure of BBOB was adopted, with one exception: the algorithm was run on 24 test functions, 5 instances each, but only 1 run on each instance (as opposed to 3 runs in the standard procedure)—the DIRECT algorithm is deterministic so making more than one run would give the same results. Each run was finished

- after finding a solution with fitness difference $\Delta f \leq 10^{-8}$, or
- after performing more than 10^5 function evaluations.

The maximal allowed number of evaluations was set to be independent of the dimensionality of the search space. The decision to use such setting is based on the fact that for the DIRECT algorithm, the time needed to sample new points in later phases of optimization is more time consuming than in the beginning—the algorithm does not forget anything and before selecting the next rectangles to divide, it must go through all the rectangles created before. The available time allowed for 10^5 function evaluations.

4. RESULTS

Results from experiments according to [3] on the benchmark functions given in [1, 4] are presented in Figures 2 and

3 and in Table 1. The algorithm was able to find the global optimum (at least once out of 5 runs) in 24, 19, 9, 4, and 2 cases (out of 24) for search space dimensions 2, 3, 5, 10, and 20, respectively.

The restarted version of the DIRECT algorithm was tested as well. Each multistart run was allowed to use up to 10^4 function evaluations while the individual DIRECT launch was allowed only up to 10^3 function evaluations, i.e. about 10 restarts were carried out on average. The first launch was always run on the whole search space, for the other launches it was decided randomly, if the search should be started only in a small hypercube containing the best solution found so far in the middle, or if it should search almost the whole search space².

Surprisingly, the results did not differ very much from the results of the non-restarted version, that is why they are not presented in this paper.

5. CPU TIMING EXPERIMENT

The algorithm was run with the maximal number of evaluations set to 10^5 , it was restarted for at least 30 seconds. The results show the algorithm takes on approximately $1.7 \cdot 10^{-5}$, $1.6 \cdot 10^{-5}$, $1.1 \cdot 10^{-4}$, $3.4 \cdot 10^{-3}$, $2.7 \cdot 10^{-3}$, and $2.0 \cdot 10^{-3}$ seconds per function evaluation for 2-, 3-, 5-, 10-, 20-, and 40-dimensional search space, respectively. The experiment was conducted on Intel Core 2 CPU, T5600, 1.83 GHz, 1 GB RAM with Windows XP SP3 in MATLAB R2007b.

6. CONCLUSION

Although the DIRECT algorithm is a global search one and comes with a proof of finding the global optimum, it needs quickly increasing number of function evaluations which renders the algorithm practically unusable in high-dimensional search spaces. With 10^5 allowed function evaluations, the algorithm is usable in 2-, or 3-dimensional search space, for 5- and more dimensional spaces the algorithm converges too slowly.

Acknowledgements

The project was supported by the Ministry of Education, Youth and Sport of the Czech Republic with the grant No. MSM6840770012 entitled “Transdisciplinary Research in Biomedical Engineering II”.

7. REFERENCES

- [1] S. Finck, N. Hansen, R. Ros, and A. Auger. Real-parameter black-box optimization benchmarking 2009: Presentation of the noiseless functions. Technical Report 2009/20, Research Center PPE, 2009.
- [2] D. E. Finkel. *DIRECT Optimization Algorithm User Guide*, 2003.
- [3] N. Hansen, A. Auger, S. Finck, and R. Ros. Real-parameter black-box optimization benchmarking 2009: Experimental setup. Technical Report RR-6828, INRIA, 2009.
- [4] N. Hansen, S. Finck, R. Ros, and A. Auger. Real-parameter black-box optimization benchmarking 2009: Noiseless functions definitions. Technical Report RR-6829, INRIA, 2009.
- [5] D. R. Jones, C. D. Perttunen, and B. E. Stuckman. Lipschitzian optimization without the lipschitz constant. *Journal of Optimization Theory and Applications*, 79(1):157–181, October 1993.
- [6] C. Kelley. Research and codes: DIRECT. Published online http://www4.ncsu.edu/~ctk/Finkel_Direct/, 2004. Version from June 22, 2004 was used.

²A random point was chosen in the hypercube. This point and the most distant corner of the $\langle -5, 5 \rangle^D$ hypercube formed the two defining corners of the reduced search space.

