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-overlaping 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 hyperectangle corner is used.

In each iteartion 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 division[1] 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.

[1]The 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.

GECCO'09, July 8–12, 2009, Montréal Québec, Canada.

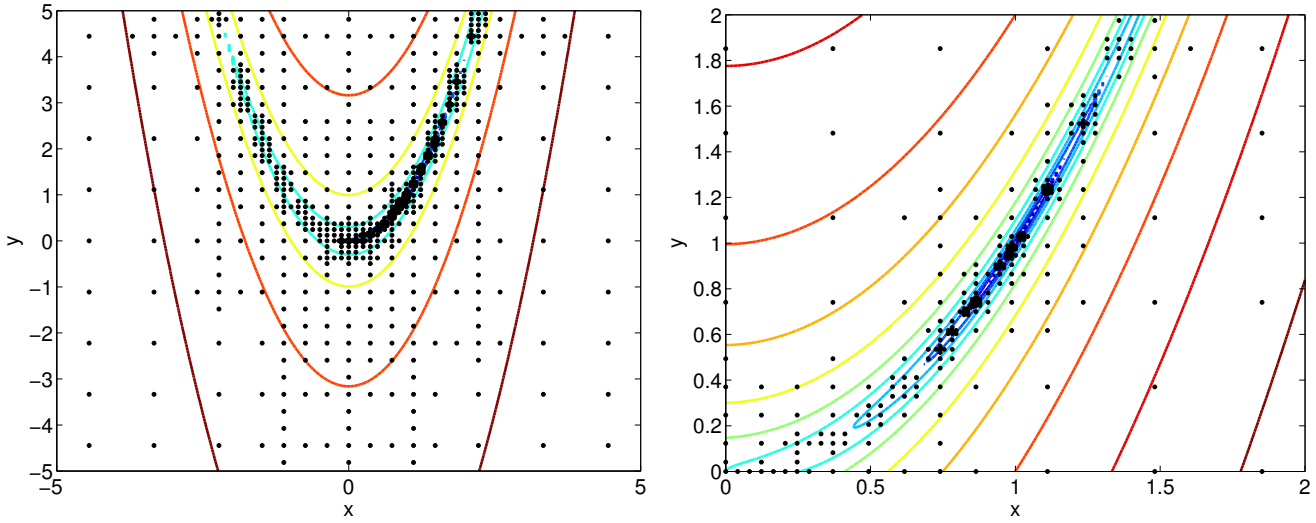

Figure 1: DIRECT algorithm optimizing 2D Rosenbrock's function using 1000 samples. Left: the overall picture, interval $\langle -5, 5\rangle^2$, right: the detail of interval $\langle 0, 2\rangle^2$

.

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 signficant 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 $\langle -5, 5\rangle^D$ hypercube (D is the search space dimensionality), thus it may seem that this hypercube can be a good choice for the box constraits 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 $\langle -6, 6\rangle^D$ 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 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 space[2].

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 moredimensional 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.

[2] A 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.

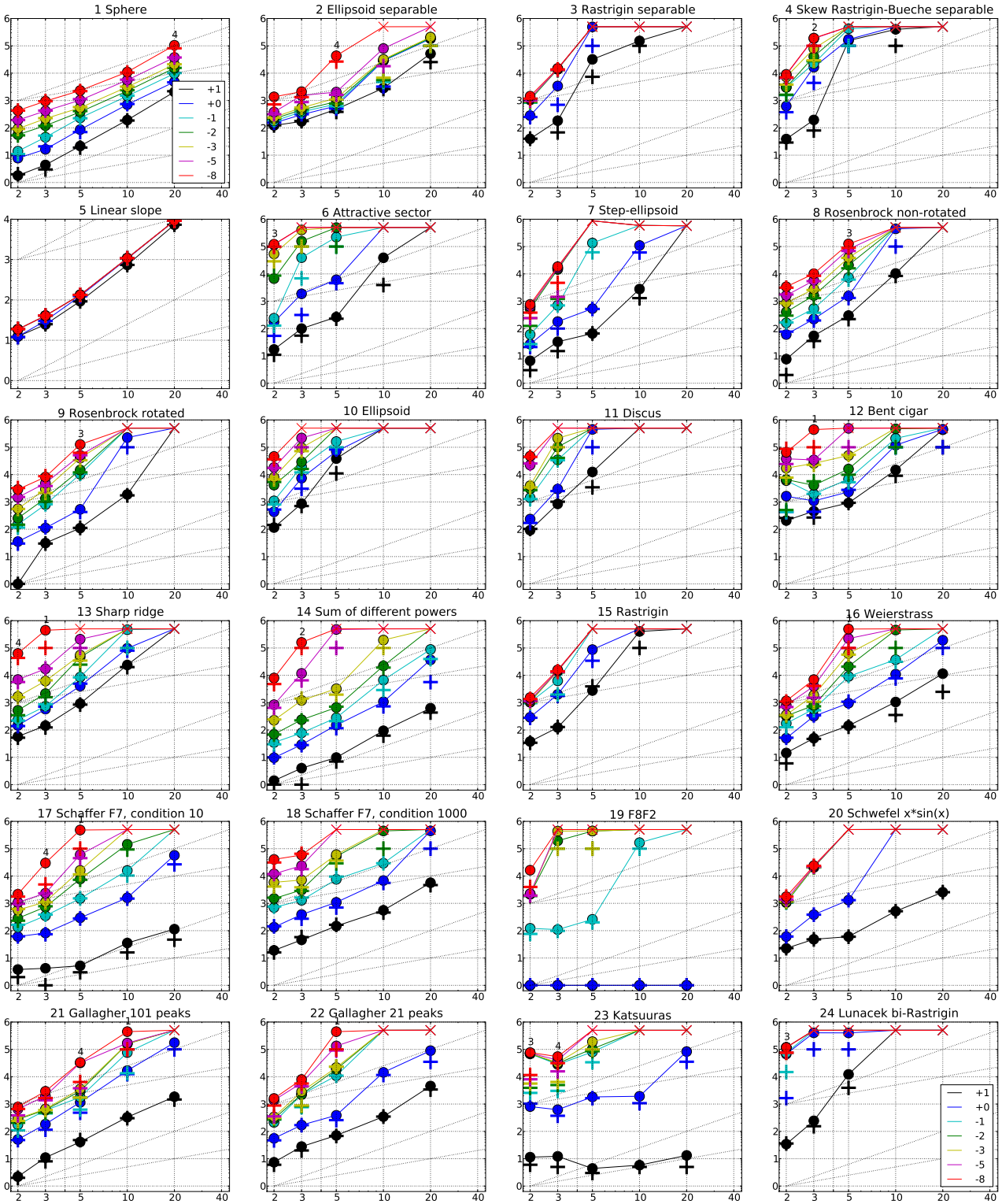

Figure 2: Expected Running Time (ERT, ●) **to reach** $f_{\mathrm{opt}} + \Delta f$ **and median number of function evaluations of successful trials** (+), **shown for** $\Delta f = 10, 1, 10^{-1}, 10^{-2}, 10^{-3}, 10^{-5}, 10^{-8}$ **(the exponent is given in the legend of** f_1 **and** f_{24}**) versus dimension in log-log presentation. The** $\mathrm{ERT}(\Delta f)$ **equals to** $\#\mathrm{FEs}(\Delta f)$ **divided by the number of successful trials, where a trial is successful if** $f_{\mathrm{opt}} + \Delta f$ **was surpassed during the trial. The** $\#\mathrm{FEs}(\Delta f)$ **are the total number of function evaluations while** $f_{\mathrm{opt}} + \Delta f$ **was not surpassed during the trial from all respective trials (successful and unsuccessful), and** f_{opt} **denotes the optimal function value. Crosses** (×) **indicate the total number of function evaluations** $\#\mathrm{FEs}(-\infty)$**. Numbers above ERT-symbols indicate the number of successful trials. Annotated numbers on the ordinate are decimal logarithms. Additional grid lines show linear and quadratic scaling.**

Δf	#	ERT	10%	90%	RT_{succ}
f_1 in 5-D, N=5, mFE=2723					
10	5	2.2e1	1.7e1	2.6e1	2.2e1
1	5	8.6e1	6.9e1	1.0e2	8.6e1
1e−1	5	2.3e2	2.2e2	2.5e2	2.3e2
1e−3	5	5.4e2	4.9e2	6.0e2	5.4e2
1e−5	5	1.0e3	9.1e2	1.1e3	1.0e3
1e−8	5	2.3e3	2.1e3	2.5e3	2.3e3
f_1 in 20-D, N=5, mFE=100651					
10	5	2.1e3	1.9e3	2.3e3	2.1e3
1	5	4.8e3	4.6e3	5.1e3	4.8e3
1e−1	5	9.7e3	8.9e3	1.0e4	9.7e3
1e−3	5	2.1e4	2.0e4	2.2e4	2.1e4
1e−5	5	3.8e4	3.6e4	3.9e4	3.8e4
1e−8	4	1.0e5	9.5e4	1.1e5	8.1e4
f_3 in 5-D, N=5, mFE=100137					
10	4	3.2e4	7.1e3	5.7e4	7.0e3
1	1	4.9e5	4.8e5	5.0e5	1.0e5
1e−1	0	*30e−1*	*99e−2*	*13e+0*	8.9e4
1e−3	.	.	.	.	.
1e−5	.	.	.	.	.
1e−8	.	.	.	.	.
f_3 in 20-D, N=5, mFE=100787					
10	0	*43e+0*	*24e+0*	*53e+0*	7.1e4
1	.	.	.	.	.
1e−1	.	.	.	.	.
1e−3	.	.	.	.	.
1e−5	.	.	.	.	.
1e−8	.	.	.	.	.
f_5 in 5-D, N=5, mFE=133					
10	5	9.2e1	9.2e1	9.3e1	9.2e1
1	5	1.2e2	1.2e2	1.2e2	1.2e2
1e−1	5	1.3e2	1.3e2	1.3e2	1.3e2
1e−3	5	1.3e2	1.3e2	1.3e2	1.3e2
1e−5	5	1.3e2	1.3e2	1.3e2	1.3e2
1e−8	5	1.3e2	1.3e2	1.3e2	1.3e2
f_5 in 20-D, N=5, mFE=9167					
10	5	7.3e3	7.3e3	7.3e3	7.3e3
1	5	9.1e3	9.0e3	9.2e3	9.1e3
1e−1	5	9.2e3	9.2e3	9.2e3	9.2e3
1e−3	5	9.2e3	9.2e3	9.2e3	9.2e3
1e−5	5	9.2e3	9.2e3	9.2e3	9.2e3
1e−8	5	9.2e3	9.2e3	9.2e3	9.2e3
f_7 in 5-D, N=5, mFE=317725					
10	5	6.6e1	5.1e1	8.1e1	6.6e1
1	5	5.4e2	3.9e2	6.8e2	5.4e2
1e−1	3	1.3e5	6.1e4	2.2e5	1.1e5
1e−3	0	*41e−3*	*13e−3*	*46e−2*	2.0e3
1e−5	.	.	.	.	.
1e−8	.	.	.	.	.
f_7 in 20-D, N=5, mFE=133265					
10	0	*15e+0*	*10e+0*	*17e+0*	3.2e4
1	.	.	.	.	.
1e−1	.	.	.	.	.
1e−3	.	.	.	.	.
1e−5	.	.	.	.	.
1e−8	.	.	.	.	.
f_9 in 5-D, N=5, mFE=100121					
10	5	1.1e2	8.0e1	1.4e2	1.1e2
1	5	5.3e2	4.0e2	6.7e2	5.3e2
1e−1	5	1.0e4	6.4e3	1.4e4	1.0e4
1e−3	5	4.0e4	3.0e4	5.2e4	4.0e4
1e−5	5	5.1e4	4.0e4	6.1e4	5.1e4
1e−8	3	1.3e5	1.1e5	1.5e5	7.2e4
f_9 in 20-D, N=5, mFE=100785					
10	0	*22e+0*	*22e+0*	*29e+0*	1.0e5
1	.	.	.	.	.
1e−1	.	.	.	.	.
1e−3	.	.	.	.	.
1e−5	.	.	.	.	.
1e−8	.	.	.	.	.
f_{11} in 5-D, N=5, mFE=100066					
10	5	1.2e4	3.1e3	2.2e4	1.2e4
1	1	4.5e5	4.0e5	5.0e5	5.0e4
1e−1	0	*19e−1*	*42e−2*	*24e−1*	8.9e4
1e−3	.	.	.	.	.
1e−5	.	.	.	.	.
1e−8	.	.	.	.	.
f_{11} in 20-D, N=5, mFE=100325					
10	0	*76e+0*	*62e+0*	*98e+0*	1.0e5
1	.	.	.	.	.
1e−1	.	.	.	.	.
1e−3	.	.	.	.	.
1e−5	.	.	.	.	.
1e−8	.	.	.	.	.
f_{13} in 5-D, N=5, mFE=100058					
10	5	9.2e2	6.8e2	1.2e3	9.2e2
1	5	4.0e3	2.6e3	5.3e3	4.0e3
1e−1	5	8.6e3	7.1e3	1.0e4	8.6e3
1e−3	4	5.5e4	3.5e4	7.6e4	5.1e4
1e−5	2	2.1e5	1.9e5	2.3e5	8.0e4
1e−8	0	*12e−6*	*92e−8*	*15e−3*	8.9e4
f_{13} in 20-D, N=5, mFE=100597					
10	0	*13e+1*	*45e+0*	*35e+1*	8.9e4
1	.	.	.	.	.
1e−1	.	.	.	.	.
1e−3	.	.	.	.	.
1e−5	.	.	.	.	.
1e−8	.	.	.	.	.
f_{15} in 5-D, N=5, mFE=100183					
10	5	2.8e3	1.8e3	3.7e3	2.8e3
1	3	8.7e4	4.8e4	1.2e5	6.9e4
1e−1	0	*99e−2*	*27e−2*	*30e−1*	6.3e4
1e−3	.	.	.	.	.
1e−5	.	.	.	.	.
1e−8	.	.	.	.	.
f_{15} in 20-D, N=5, mFE=100399					
10	0	*10e+1*	*52e+0*	*12e+1*	7.9e4
1	.	.	.	.	.
1e−1	.	.	.	.	.
1e−3	.	.	.	.	.
1e−5	.	.	.	.	.
1e−8	.	.	.	.	.
f_{17} in 5-D, N=5, mFE=100060					
10	5	5.2e0	2.2e0	7.8e0	5.2e0
1	5	3.0e2	2.1e2	3.9e2	3.0e2
1e−1	5	1.5e3	1.2e3	1.8e3	1.5e3
1e−3	5	1.6e4	8.7e3	2.4e4	1.6e4
1e−5	4	6.0e4	3.9e4	8.4e4	4.9e4
1e−8	1	4.8e5	4.6e5	5.0e5	8.2e4
f_{17} in 20-D, N=5, mFE=100493					
10	5	1.1e2	5.6e1	1.8e2	1.1e2
1	4	5.7e4	3.3e4	8.0e4	3.9e4
1e−1	0	*55e−2*	*21e−2*	*10e−1*	4.5e4
1e−3	.	.	.	.	.
1e−5	.	.	.	.	.
1e−8	.	.	.	.	.
f_{19} in 5-D, N=5, mFE=100056					
10	5	1.0e0	1.0e0	1.0e0	1.0e0
1	5	1.0e0	1.0e0	1.0e0	1.0e0
1e−1	5	2.6e2	1.9e2	3.3e2	2.6e2
1e−3	1	4.4e5	3.9e5	5.0e5	1.0e5
1e−5	0	*16e−3*	*31e−5*	*51e−3*	4.0e4
1e−8	.	.	.	.	.
f_{19} in 20-D, N=5, mFE=100159					
10	5	1.0e0	1.0e0	1.0e0	1.0e0
1	5	1.0e0	1.0e0	1.0e0	1.0e0
1e−1	0	*21e−2*	*18e−2*	*21e−2*	5.6e4
1e−3	.	.	.	.	.
1e−5	.	.	.	.	.
1e−8	.	.	.	.	.
f_{21} in 5-D, N=5, mFE=100091					
10	5	4.1e1	3.2e1	5.0e1	4.1e1
1	5	1.2e3	4.4e2	1.9e3	1.2e3
1e−1	5	1.8e3	6.8e2	2.9e3	1.8e3
1e−3	5	3.6e3	2.2e3	5.2e3	3.6e3
1e−5	4	3.2e4	7.6e3	6.0e4	3.1e4
1e−8	4	3.3e4	9.1e3	5.7e4	3.2e4
f_{21} in 20-D, N=5, mFE=100673					
10	5	1.8e3	1.4e3	2.4e3	1.8e3
1	2	1.8e5	1.2e5	2.2e5	5.5e4
1e−1	0	*19e−1*	*11e−2*	*54e−1*	8.9e4
1e−3	.	.	.	.	.
1e−5	.	.	.	.	.
1e−8	.	.	.	.	.
f_{23} in 5-D, N=5, mFE=100044					
10	5	4.4e0	2.4e0	6.2e0	4.4e0
1	5	1.8e3	9.5e2	2.7e3	1.8e3
1e−1	3	8.1e4	3.8e4	1.1e5	4.6e4
1e−3	2	1.9e5	1.5e5	2.3e5	6.6e4
1e−5	0	*93e−4*	*11e−5*	*44e−2*	8.9e4
1e−8	.	.	.	.	.
f_{23} in 20-D, N=5, mFE=100073					
10	5	1.3e1	4.2e0	2.3e1	1.3e1
1	3	8.3e4	4.6e4	1.2e5	1.6e4
1e−1	0	*38e−2*	*34e−2*	*13e−1*	7.1e4
1e−3	.	.	.	.	.
1e−5	.	.	.	.	.
1e−8	.	.	.	.	.

Δf	#	ERT	10%	90%	RT_{succ}
f_2 in 5-D, N=5, mFE=100113					
10	5	4.8e2	3.8e2	5.7e2	4.8e2
1	5	6.3e2	5.0e2	7.6e2	6.3e2
1e−1	5	7.4e2	5.6e2	9.3e2	7.4e2
1e−3	5	1.2e3	1.0e3	1.5e3	1.2e3
1e−5	5	2.0e3	1.5e3	2.5e3	2.0e3
1e−8	4	4.3e4	1.9e4	6.7e4	4.0e4
f_2 in 20-D, N=5, mFE=100777					
10	4	5.2e4	3.2e4	7.1e4	4.3e4
1	2	1.8e5	1.5e5	2.2e5	6.6e4
1e−1	2	1.9e5	1.6e5	2.2e5	6.9e4
1e−3	2	2.1e5	1.8e5	2.3e5	7.7e4
1e−5	0	*14e−1*	*43e−6*	*55e+0*	7.9e4
1e−8	.	.	.	.	.
f_4 in 5-D, N=5, mFE=100183					
10	2	1.6e5	1.0e5	2.0e5	5.4e4
1	2	1.7e5	1.3e5	2.1e5	6.2e4
1e−1	1	4.2e5	3.4e5	5.0e5	1.0e5
1e−3	0	*11e+0*	*51e−3*	*13e+0*	2.8e4
1e−5	.	.	.	.	.
1e−8	.	.	.	.	.
f_4 in 20-D, N=5, mFE=100863					
10	0	*88e+0*	*31e+0*	*10e+1*	7.9e4
1	.	.	.	.	.
1e−1	.	.	.	.	.
1e−3	.	.	.	.	.
1e−5	.	.	.	.	.
1e−8	.	.	.	.	.
f_6 in 5-D, N=5, mFE=100125					
10	5	2.6e2	1.9e2	3.5e2	2.6e2
1	5	6.1e3	3.9e3	8.0e3	6.1e3
1e−1	2	2.2e5	2.0e5	2.5e5	7.2e4
1e−3	0	*22e−2*	*26e−4*	*46e−2*	7.1e4
1e−5	.	.	.	.	.
1e−8	.	.	.	.	.
f_6 in 20-D, N=5, mFE=100531					
10	0	*40e+0*	*19e+0*	*58e+0*	1.0e5
1	.	.	.	.	.
1e−1	.	.	.	.	.
1e−3	.	.	.	.	.
1e−5	.	.	.	.	.
1e−8	.	.	.	.	.
f_8 in 5-D, N=5, mFE=100051					
10	5	3.0e2	2.1e2	3.8e2	3.0e2
1	5	1.6e3	9.2e2	2.2e3	1.6e3
1e−1	5	7.4e3	5.3e3	9.7e3	7.4e3
1e−3	5	3.9e4	2.5e4	5.4e4	3.9e4
1e−5	4	8.0e4	5.6e4	1.0e5	6.9e4
1e−8	3	1.3e5	9.1e4	1.5e5	9.7e4
f_8 in 20-D, N=5, mFE=100925					
10	0	*64e+0*	*17e+0*	*11e+1*	1.0e5
1	.	.	.	.	.
1e−1	.	.	.	.	.
1e−3	.	.	.	.	.
1e−5	.	.	.	.	.
1e−8	.	.	.	.	.
f_{10} in 5-D, N=5, mFE=100068					
10	4	3.8e4	1.1e4	6.3e4	3.7e4
1	4	7.1e4	4.4e4	9.9e4	6.8e4
1e−1	2	1.6e5	1.1e5	2.1e5	1.0e5
1e−3	0	*15e−2*	*76e−3*	*11e+0*	8.9e4
1e−5	.	.	.	.	.
1e−8	.	.	.	.	.
f_{10} in 20-D, N=5, mFE=100261					
10	0	*94e+2*	*45e+2*	*11e+3*	1.0e5
1	.	.	.	.	.
1e−1	.	.	.	.	.
1e−3	.	.	.	.	.
1e−5	.	.	.	.	.
1e−8	.	.	.	.	.
f_{12} in 5-D, N=5, mFE=100135					
10	5	9.2e2	8.7e2	9.6e2	9.2e2
1	5	2.3e3	1.7e3	3.0e3	2.3e3
1e−1	5	6.9e3	4.4e3	9.7e3	6.9e3
1e−3	5	5.0e4	3.8e4	6.1e4	5.0e4
1e−5	1	4.9e5	4.8e5	5.0e5	1.0e5
1e−8	0	*21e−6*	*30e−7*	*46e−5*	8.9e4
f_{12} in 20-D, N=5, mFE=100847					
10	1	4.4e5	3.7e5	5.0e5	1.0e5
1	1	4.6e5	4.1e5	5.0e5	1.0e5
1e−1	0	*25e+3*	*47e−2*	*67e+4*	1.0e5
1e−3	.	.	.	.	.
1e−5	.	.	.	.	.
1e−8	.	.	.	.	.
f_{14} in 5-D, N=5, mFE=100109					
10	5	9.8e0	4.2e0	1.6e1	9.8e0
1	5	1.5e2	9.6e1	2.0e2	1.5e2
1e−1	5	2.8e2	1.9e2	3.6e2	2.8e2
1e−3	5	3.2e3	1.8e3	4.6e3	3.2e3
1e−5	1	4.8e5	4.5e5	5.0e5	1.0e5
1e−8	0	*62e−6*	*19e−7*	*10e−5*	8.9e4
f_{14} in 20-D, N=5, mFE=100461					
10	5	6.3e2	4.2e2	8.3e2	6.3e2
1	4	3.7e4	1.2e4	6.1e4	1.2e4
1e−1	3	8.8e4	5.0e4	1.3e5	4.2e4
1e−3	0	*90e−3*	*17e−3*	*12e−1*	1.0e5
1e−5	.	.	.	.	.
1e−8	.	.	.	.	.
f_{16} in 5-D, N=5, mFE=100092					
10	5	1.4e2	1.1e2	1.7e2	1.4e2
1	5	9.6e2	8.0e2	1.1e3	9.6e2
1e−1	5	9.1e3	5.4e3	1.3e4	9.1e3
1e−3	4	6.2e4	3.8e4	8.6e4	3.7e4
1e−5	2	2.2e5	2.1e5	2.4e5	1.0e5
1e−8	1	4.9e5	4.9e5	5.0e5	1.0e5
f_{16} in 20-D, N=5, mFE=100254					
10	5	1.1e4	2.2e3	2.1e4	1.1e4
1	2	1.9e5	1.5e5	2.2e5	6.8e4
1e−1	0	*12e−1*	*31e−2*	*85e−1*	7.1e4
1e−3	.	.	.	.	.
1e−5	.	.	.	.	.
1e−8	.	.	.	.	.
f_{18} in 5-D, N=5, mFE=100018					
10	5	1.5e2	1.1e2	1.8e2	1.5e2
1	5	1.1e3	6.5e2	1.5e3	1.1e3
1e−1	5	7.6e3	4.8e3	1.0e4	7.6e3
1e−3	4	6.0e4	4.0e4	8.2e4	5.6e4
1e−5	0	*44e−6*	*17e−6*	*25e−3*	7.9e4
1e−8	.	.	.	.	.
f_{18} in 20-D, N=5, mFE=100168					
10	5	5.7e3	3.6e3	7.8e3	5.7e3
1	1	4.5e5	4.0e5	5.0e5	1.0e5
1e−1	0	*12e−1*	*51e−2*	*30e−1*	1.0e5
1e−3	.	.	.	.	.
1e−5	.	.	.	.	.
1e−8	.	.	.	.	.
f_{20} in 5-D, N=5, mFE=100167					
10	5	6.0e1	6.0e1	6.1e1	6.0e1
1	5	1.3e3	1.3e3	1.3e3	1.3e3
1e−1	0	*47e−2*	*47e−2*	*47e−2*	4.0e4
1e−3	.	.	.	.	.
1e−5	.	.	.	.	.
1e−8	.	.	.	.	.
f_{20} in 20-D, N=5, mFE=100925					
10	5	2.6e3	2.6e3	2.6e3	2.6e3
1	0	*18e−1*	*18e−1*	*18e−1*	6.3e4
1e−1	.	.	.	.	.
1e−3	.	.	.	.	.
1e−5	.	.	.	.	.
1e−8	.	.	.	.	.
f_{22} in 5-D, N=5, mFE=100105					
10	5	7.1e1	5.3e1	8.9e1	7.1e1
1	5	3.9e2	3.0e2	4.9e2	3.9e2
1e−1	5	1.1e4	6.5e3	1.5e4	1.1e4
1e−3	5	2.2e4	1.6e4	2.8e4	2.2e4
1e−5	3	1.3e5	1.1e5	1.6e5	7.0e4
1e−8	1	4.4e5	3.9e5	5.0e5	1.0e5
f_{22} in 20-D, N=5, mFE=100775					
10	5	4.6e3	2.6e3	6.7e3	4.6e3
1	3	9.0e4	5.6e4	1.4e5	7.9e4
1e−1	0	*71e−2*	*69e−2*	*20e−1*	8.9e4
1e−3	.	.	.	.	.
1e−5	.	.	.	.	.
1e−8	.	.	.	.	.
f_{24} in 5-D, N=5, mFE=100102					
10	5	1.2e4	3.0e3	2.2e4	1.2e4
1	1	4.0e5	3.0e5	5.0e5	1.0e5
1e−1	0	*72e−1*	*75e−2*	*91e−1*	7.1e4
1e−3	.	.	.	.	.
1e−5	.	.	.	.	.
1e−8	.	.	.	.	.
f_{24} in 20-D, N=5, mFE=100091					
10	0	*15e+1*	*14e+1*	*17e+1*	7.9e4
1	.	.	.	.	.
1e−1	.	.	.	.	.
1e−3	.	.	.	.	.
1e−5	.	.	.	.	.
1e−8	.	.	.	.	.

Table 1: Shown are, for a given target difference to the optimal function value Δf: the number of successful trials (#); the expected running time to surpass $f_{\text{opt}}+\Delta f$ (ERT, see Figure 2); the 10%-tile and 90%-tile of the bootstrap distribution of ERT; the average number of function evaluations in successful trials or, if none was successful, as last entry the median number of function evaluations to reach the best function value (RT_{succ}). If $f_{\text{opt}}+\Delta f$ was never reached, figures in *italics* denote the best achieved Δf-value of the median trial and the 10% and 90%-tile trial. Furthermore, N denotes the number of trials, and mFE denotes the maximum of number of function evaluations executed in one trial. See Figure 2 for the names of functions.

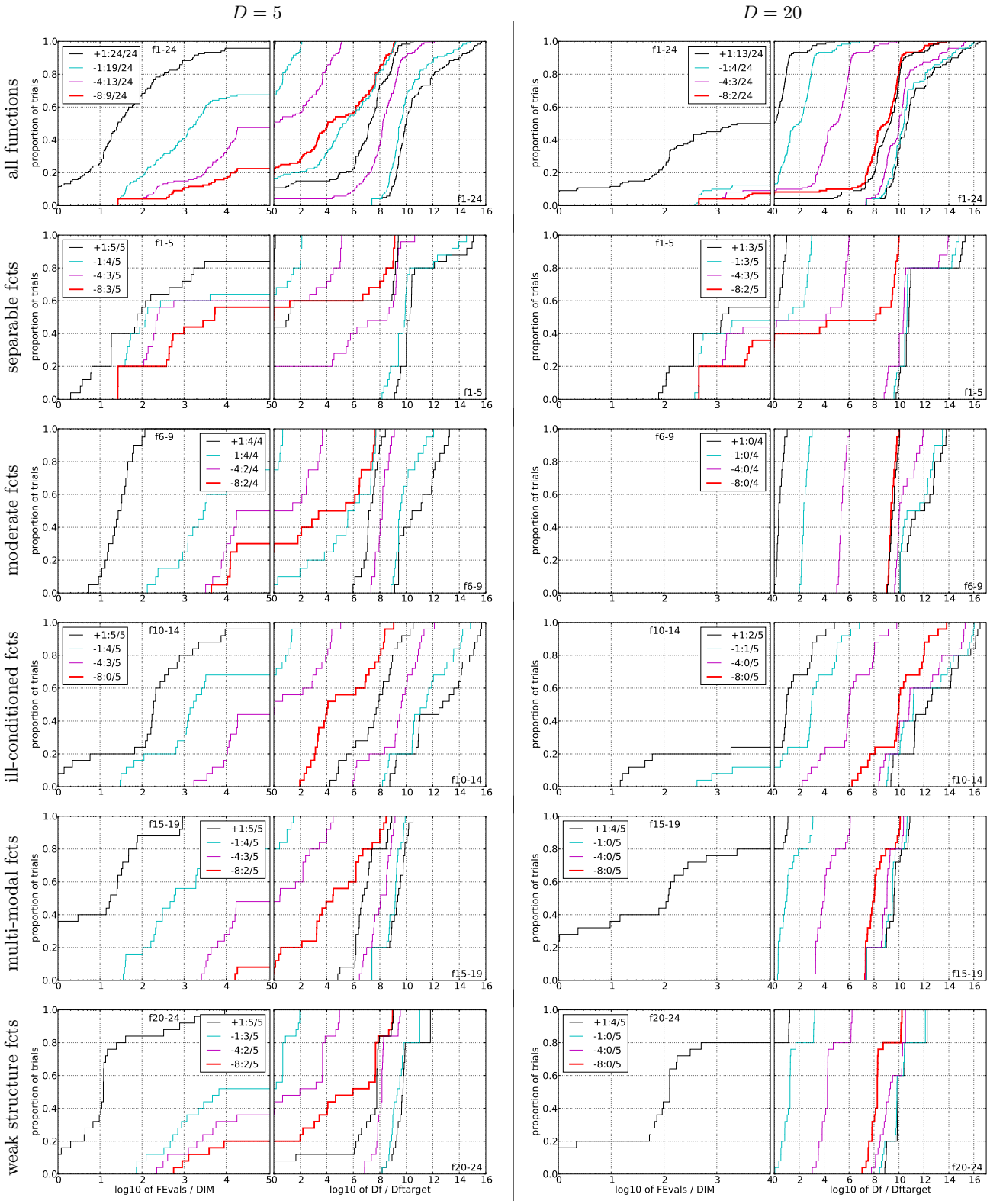

Figure 3: Empirical cumulative distribution functions (ECDFs), plotting the fraction of trials versus running time (left) or Δf. Left subplots: ECDF of the running time (number of function evaluations), divided by search space dimension D, to fall below $f_{\mathrm{opt}} + \Delta f$ with $\Delta f = 10^k$, where k is the first value in the legend. Right subplots: ECDF of the best achieved Δf divided by 10^k (upper left lines in continuation of the left subplot), and best achieved Δf divided by 10^{-8} for running times of $D, 10\,D, 100\,D \ldots$ function evaluations (from right to left cycling black-cyan-magenta). Top row: all results from all functions; second row: separable functions; third row: misc. moderate functions; fourth row: ill-conditioned functions; fifth row: multi-modal functions with adequate structure; last row: multi-modal functions with weak structure. The legends indicate the number of functions that were solved in at least one trial. FEvals denotes number of function evaluations, D and DIM denote search space dimension, and Δf and Df denote the difference to the optimal function value.