

Benchmarking the Pure Random Search on the BBOB-2009 Testbed

Anne Auger, Raymond Ros

► **To cite this version:**

Anne Auger, Raymond Ros. Benchmarking the Pure Random Search on the BBOB-2009 Testbed. ACM-GECCO Genetic and Evolutionary Computation Conference, Jul 2009, Montreal, Canada. 2009. <inria-00430532>

HAL Id: inria-00430532

<https://hal.inria.fr/inria-00430532>

Submitted on 8 Nov 2009

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Benchmarking the Pure Random Search on the BBOB-2009 Testbed

Anne Auger
TAO Team - INRIA Saclay
LRI, Bat 490, Univ. Paris-Sud
91405 Orsay Cedex, France
anne.auger@inria.fr

Raymond Ros
Univ. Paris-Sud, LRI
UMR 8623 / INRIA Saclay, projet TAO
F-91405 Orsay, France
raymond.ros@lri.fr

ABSTRACT

We benchmark the pure random search algorithm on the BBOB 2009 noise-free testbed. Each candidate solution is sampled uniformly in $[-5, 5]^D$, where D denotes the search space dimension. The maximum number of function evaluations chosen is 10^6 times the search space dimension. With this budget the algorithm is not able to solve any single function of the testbed.

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

Keywords

Benchmarking, Pure random search, Monte-Carlo, Black-box optimization, Evolutionary computation

1. INTRODUCTION

The pure random search, first proposed by Brooks in 1958 [1], is the most simple stochastic search algorithm that consists in sampling each search point independently in the search domain and keeping the best solution found.

2. METHODS

We have used a uniform sampling in $[-5, 5]^D$, where D denotes the dimension of the search space. The experiments according to [3] on the benchmark functions given in [2, 4] have been conducted using both a C-code and a Matlab code. The algorithm implementation in Matlab is given in Figure 1. A maximum of $10^6 \times D$ function evaluations has

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: Pure random search in Matlab. At each iteration (iter), 200 points are sampled and stored in a matrix of size $D \times 200$ so as to reduce loops and function calls within Matlab.

```
function MY_OPTIMIZER(FUN, DIM, ftarget, maxfunevals)
% MY_OPTIMIZER(FUN, DIM, ftarget, maxfunevals)
% samples new points uniformly randomly in [-5,5]^DIM
% and evaluates them on FUN until ftarget or maxfunevals
% is reached, or until 1e8 * DIM fevals are conducted.
% Relies on FUN to keep track of the best point.

maxfunevals = min(1e8 * DIM, maxfunevals);
popsize = min(maxfunevals, 200);
for iter = 1:ceil(maxfunevals/popsize)
    feval(FUN, 10 * rand(DIM, popsize) - 5);
    if feval(FUN, 'fbest') < ftarget % task achieved
        break;
    end
    % if useful, modify more options here for next start
end
```

been used. The Matlab code was used for previous experiments using $10^5 \times D$ function evaluations. The simulations for 2; 5; 10 and 20 D with $10^5 \times D$ function evaluations with the C-code took 21 hours. The 40 D experiments with $10^5 \times D$ function evaluations in Matlab took 17 hours. Experiments with $10^5 \times D$ function evaluations were done on the machine described in Section 4. The experiments with $10^6 \times D$ function evaluations were done with the C-code on an Intel Xeon 2.00GHz: the experiments till dimension 20 took 38 hours and the 40-D experiments took 116 hours.

No parameter tuning was done and the crafting effort CrE [3] is computed to zero.

3. RESULTS AND DISCUSSION

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

We see that with $10^6 \times D$ function evaluations, the pure random search algorithm is not able to solve any function. However, since we use a uniform sampling in the search domain, we obtain as a by-product of the results an estimate of the volume of the sublevel sets: the sublevel sets of a function $f : \mathbb{R}^D \rightarrow \mathbb{R}$ are defined as $S_c = \{x \in \mathbb{R}^D | f(x) \leq c\}$ for c spanning \mathbb{R} . If S_c is a subset of $[-5, 5]^D$, the hitting

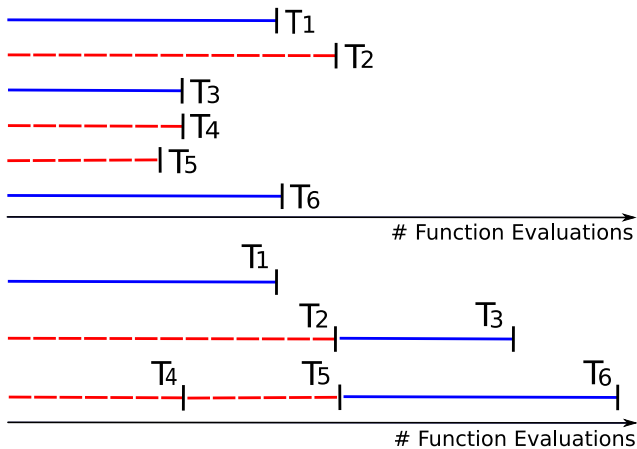


Figure 2: Illustration that $ERT(\Delta f)$ estimates the expected hitting time of an algorithm restarted until success (assuming infinite horizon): among 6 runs of the same algorithm A, the 1st, 3rd and 6th are successful while the 2nd, 4th and 5th are unsuccessful and thus T_1 , T_2+T_3 and $T_4+T_5+T_6$ are 3 instances of the algorithm restart-A (i.e., algorithm A restarted until success). Thus an estimate of the expected hitting time of restart-A is $(T_1+(T_2+T_3)+(T_4+T_5+T_6))/3$, i.e., total number of function evaluations divided by number of successes of algorithm A, i.e., $ERT(\Delta f)$. In the case where algorithm A is the pure random search, the picture is simpler because unsuccessful runs always reach the maximum number of evaluations and thus the 2nd, 4th and 5th runs have the same length. T_1 , T_2+T_3 and $T_4+T_5+T_6$ represent then 3 instances of the pure random search that would be run with infinite horizon until a success is reached and $ERT(\Delta f)$ estimates thus the expected hitting time of the pure random search with infinite horizon.

time T_c (assuming infinite horizon) of the sublevel set S_c is distributed according to a geometric random variable of parameter $p_c = Vol(S_c)/Vol([-5, 5]^D)$. The expected running time $ERT(\Delta f)$ estimates the expected value of $T_{\Delta f}$ (see Figure 2), that equals $1/p_c$ since $T_{\Delta f}$ is a geometric random variable. And thus $ERT(\Delta f)$ gives the ratio between $Vol([-5, 5]^D)$ and $Vol(S_c)$.

4. CPU TIMING EXPERIMENT

For the timing experiment the pure random search was run with a maximum of $10^5 \times D$ function evaluations and restarted until 30 seconds has passed (according to Figure 2 in [3]). The experiments have been conducted with an Intel Core 2 Duo 2.53 GHz under Mac OS X Version 10.5.6 using the C-code provided. The time per function evaluation was 2.0; 2.3; 2.8; 4.2; 6.9 times 10^{-7} seconds in dimensions 2; 3; 5; 10; 20; 40 respectively.

5. CONCLUSION

We have presented the results of the pure random search, a non-adaptive algorithm, that does not use information gathered during search for guiding its next steps. The performance is poor and expected to be outperformed by any reasonable algorithm. Furthermore, those results constitute reference results useful for the investigation of more advanced algorithms.

Acknowledgments

We would like to thank Nikolaus Hansen for the way he led the BOB project, Steffen Finck and Nikolaus Hansen for their great and hard work. We also would like to thank Marc Schoenauer for his kind support and essential help on the C-code.

6. REFERENCES

- [1] S. H. Brooks. A discussion of random methods for seeking maxima. *Operations Research*, 6:244– 251, 1958.
- [2] 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.
- [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.

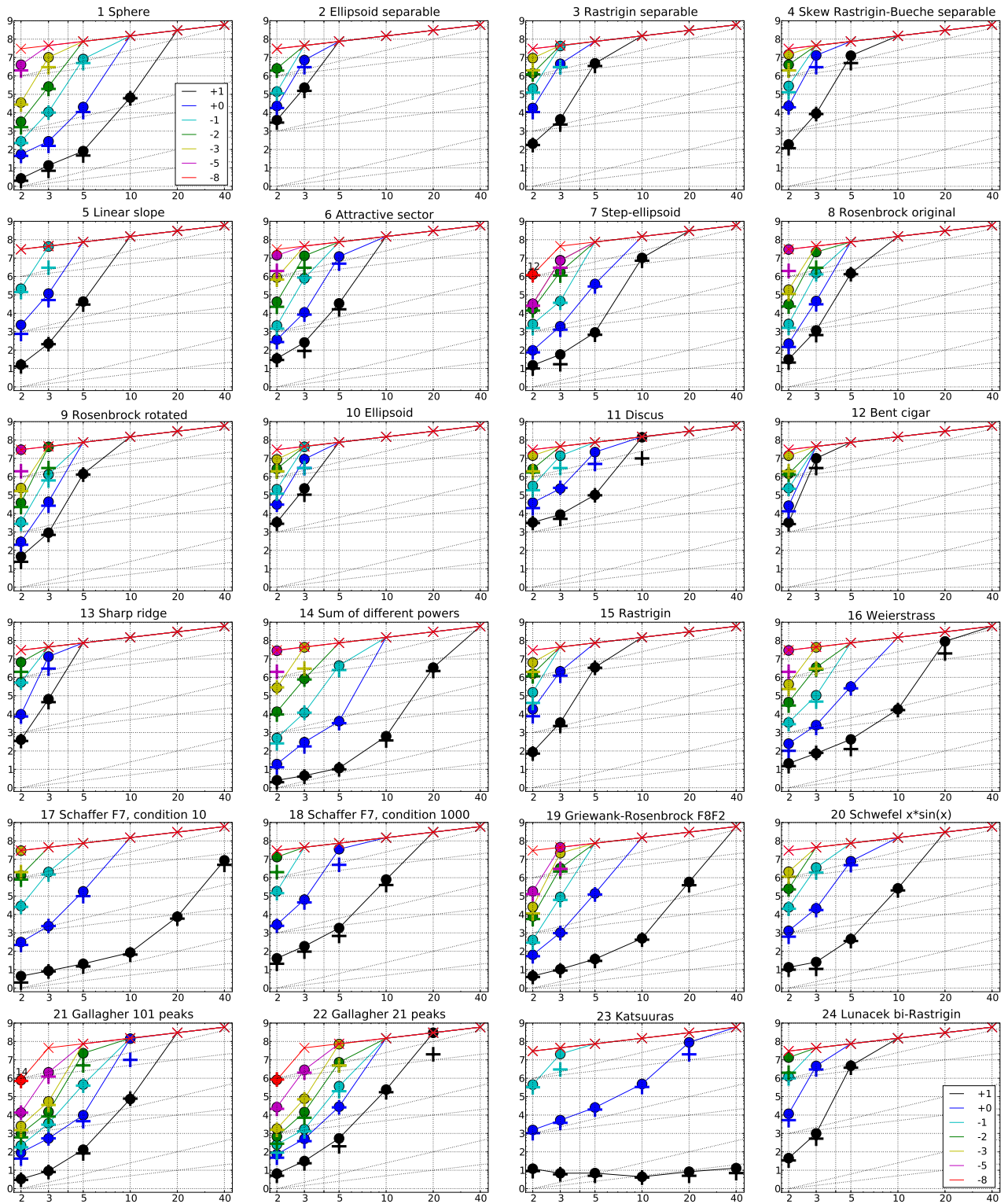


Figure 3: Expected Running Time (ERT, \bullet) to reach $f_{\text{opt}} + \Delta f$ and median number of function evaluations of successful trials (+), shown for $\Delta f = 10, 1, 10^{-1}, 10^{-2}, 10^{-3}, 10^{-5}, 10^{-8}$ (the exponent is given in the legend of f_1 and f_{24}) versus dimension in log-log presentation. The ERT(Δf) equals to $\#FEs(\Delta f)$ divided by the number of successful trials, where a trial is successful if $f_{\text{opt}} + \Delta f$ was surpassed during the trial. The $\#FEs(\Delta f)$ are the total number of function evaluations while $f_{\text{opt}} + \Delta f$ was not surpassed during the trial from all respective trials (successful and unsuccessful), and f_{opt} denotes the optimal function value. Crosses (\times) indicate the total number of function evaluations $\#FEs(-\infty)$. 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.

f1 in 5-D, N=15, mFE=5.00e6					f1 in 20-D, N=15, mFE=2.00e7					f2 in 5-D, N=15, mFE=5.00e6					f2 in 20-D, N=15, mFE=2.00e7							
Δf	#	ERT	10%	90%	RT _{succ}	#	ERT	10%	90%	RT _{succ}	#	ERT	10%	90%	RT _{succ}	#	ERT	10%	90%	RT _{succ}		
10	15	8.2e1	5.5e1	1.1e2	8.2e1	0	<i>29e+0</i>	<i>27e+0</i>	<i>33e+0</i>	8.3e6	10	0	<i>11e+1</i>	<i>57e+0</i>	<i>22e+1</i>	1.6e6	0	<i>12e+4</i>	<i>79e+3</i>	<i>15e+4</i>	1.0e7	
1	15	2.0e4	1.5e4	2.0e4	2.0e4	1	
1e-1	7	8.3e6	5.9e6	1.3e7	3.8e6	1e-1	
1e-3	0	<i>10e-2</i>	<i>55e-3</i>	<i>15e-2</i>	2.5e6	1e-3	
1e-5	1e-5
1e-8	1e-8
f3 in 5-D, N=15, mFE=5.00e6					f3 in 20-D, N=15, mFE=2.00e7					f4 in 5-D, N=15, mFE=5.00e6					f4 in 20-D, N=15, mFE=2.00e7							
Δf	#	ERT	10%	90%	RT _{succ}	#	ERT	10%	90%	RT _{succ}	#	ERT	10%	90%	RT _{succ}	#	ERT	10%	90%	RT _{succ}		
10	10	4.8e6	3.8e6	6.3e6	3.6e6	0	<i>26e+1</i>	<i>23e+1</i>	<i>29e+1</i>	6.3e6	10	5	1.3e7	8.6e6	2.3e7	4.6e6	0	<i>33e+1</i>	<i>30e+1</i>	<i>35e+1</i>	1.0e7	
1	0	<i>83e-1</i>	<i>56e-1</i>	<i>11e+0</i>	2.8e6	1	0	<i>12e+0</i>	<i>47e-1</i>	<i>16e+0</i>	2.2e6	
1e-1	1e-1
1e-3	1e-3
1e-5	1e-5
1e-8	1e-8
f5 in 5-D, N=15, mFE=5.00e6					f5 in 20-D, N=15, mFE=2.00e7					f6 in 5-D, N=15, mFE=5.00e6					f6 in 20-D, N=15, mFE=2.00e7							
Δf	#	ERT	10%	90%	RT _{succ}	#	ERT	10%	90%	RT _{succ}	#	ERT	10%	90%	RT _{succ}	#	ERT	10%	90%	RT _{succ}		
10	15	4.3e4	3.0e4	5.8e4	4.3e4	0	<i>11e+1</i>	<i>97e+0</i>	<i>12e+1</i>	8.9e6	10	15	3.5e4	2.2e4	4.7e4	3.5e4	0	<i>48e+1</i>	<i>21e+1</i>	<i>46e+3</i>	1.1e7	
1	0	<i>37e-1</i>	<i>23e-1</i>	<i>42e-1</i>	2.0e6	1	5	1.2e7	7.8e6	2.3e7	3.4e6	
1e-1	1e-1	0	<i>14e-1</i>	<i>73e-2</i>	<i>17e-1</i>	2.2e6	
1e-3	1e-3
1e-5	1e-5
1e-8	1e-8
f7 in 5-D, N=15, mFE=5.00e6					f7 in 20-D, N=15, mFE=2.00e7					f8 in 5-D, N=15, mFE=5.00e6					f8 in 20-D, N=15, mFE=2.00e7							
Δf	#	ERT	10%	90%	RT _{succ}	#	ERT	10%	90%	RT _{succ}	#	ERT	10%	90%	RT _{succ}	#	ERT	10%	90%	RT _{succ}		
10	15	9.1e2	6.6e2	1.2e3	9.1e2	0	<i>10e+1</i>	<i>68e+0</i>	<i>11e+1</i>	6.3e6	10	15	1.5e6	1.0e6	2.0e6	1.5e6	0	<i>80e+2</i>	<i>56e+2</i>	<i>10e+3</i>	1.0e7	
1	15	3.9e5	2.9e5	4.9e5	3.9e5	1	0	<i>64e-1</i>	<i>41e-1</i>	<i>90e-1</i>	1.8e6	
1e-1	0	<i>38e-2</i>	<i>20e-2</i>	<i>66e-2</i>	2.0e6	1e-1
1e-3	1e-3
1e-5	1e-5
1e-8	1e-8
f9 in 5-D, N=15, mFE=5.00e6					f9 in 20-D, N=15, mFE=2.00e7					f10 in 5-D, N=15, mFE=5.00e6					f10 in 20-D, N=15, mFE=2.00e7							
Δf	#	ERT	10%	90%	RT _{succ}	#	ERT	10%	90%	RT _{succ}	#	ERT	10%	90%	RT _{succ}	#	ERT	10%	90%	RT _{succ}		
10	15	1.4e6	1.0e6	1.8e6	1.4e6	0	<i>68e+2</i>	<i>54e+2</i>	<i>90e+2</i>	1.0e7	10	0	<i>10e+1</i>	<i>41e+0</i>	<i>20e+1</i>	2.0e6	0	<i>11e+4</i>	<i>74e+3</i>	<i>17e+4</i>	8.9e6	
1	0	<i>58e-1</i>	<i>42e-1</i>	<i>89e-1</i>	2.8e6	1	
1e-1	1e-1
1e-3	1e-3
1e-5	1e-5
1e-8	1e-8
f11 in 5-D, N=15, mFE=5.00e6					f11 in 20-D, N=15, mFE=2.00e7					f12 in 5-D, N=15, mFE=5.00e6					f12 in 20-D, N=15, mFE=2.00e7							
Δf	#	ERT	10%	90%	RT _{succ}	#	ERT	10%	90%	RT _{succ}	#	ERT	10%	90%	RT _{succ}	#	ERT	10%	90%	RT _{succ}		
10	15	1.0e5	8.7e4	1.2e5	1.0e5	0	<i>67e+0</i>	<i>48e+0</i>	<i>70e+0</i>	1.0e7	10	0	<i>19e+3</i>	<i>12e+3</i>	<i>29e+3</i>	2.5e6	0	<i>28e+6</i>	<i>22e+6</i>	<i>35e+6</i>	6.3e6	
1	3	2.2e7	1.3e7	6.5e7	5.0e6	1	
1e-1	0	<i>11e-1</i>	<i>57e-2</i>	<i>24e-1</i>	1.8e6	1e-1	
1e-3	1e-3
1e-5	1e-5
1e-8	1e-8
f13 in 5-D, N=15, mFE=5.00e6					f13 in 20-D, N=15, mFE=2.00e7					f14 in 5-D, N=15, mFE=5.00e6					f14 in 20-D, N=15, mFE=2.00e7							
Δf	#	ERT	10%	90%	RT _{succ}	#	ERT	10%	90%	RT _{succ}	#	ERT	10%	90%	RT _{succ}	#	ERT	10%	90%	RT _{succ}		
10	0	<i>30e+0</i>	<i>17e+0</i>	<i>45e+0</i>	2.8e6	0	<i>92e+1</i>	<i>83e+1</i>	<i>10e+2</i>	7.9e6	10	15	1.2e1	8.1e0	1.5e1	1.2e1	15	3.4e6	2.5e6	4.3e6	3.4e6	
1	1	15	4.1e3	3.1e3	5.1e3	4.1e3	0	<i>80e-1</i>	<i>62e-1</i>	<i>91e-1</i>	7.9e6	
1e-1	1e-1	10	4.4e6	3.0e6	6.5e6	2.4e6	
1e-3	1e-3	0	<i>93e-3</i>	<i>49e-3</i>	<i>14e-2</i>	2.0e6	
1e-5	1e-5
1e-8	1e-8
f15 in 5-D, N=15, mFE=5.00e6					f15 in 20-D, N=15, mFE=2.00e7					f16 in 5-D, N=15, mFE=5.00e6					f16 in 20-D, N=15, mFE=2.00e7							
Δf	#	ERT	10%	90%	RT _{succ}	#	ERT	10%	90%	RT _{succ}	#	ERT	10%	90%	RT _{succ}	#	ERT	10%	90%	RT _{succ}		
10	13	3.5e6	2.8e6	4.4e6	3.0e6	0	<i>26e+1</i>	<i>23e+1</i>	<i>28e+1</i>	7.9e6	10	15	4.3e2	2.6e2	6.0e2	4.3e2	3	9.0e7	5.5e7	2.7e8	2.0e7	
1	0	<i>83e-1</i>	<i>64e-1</i>	<i>11e+0</i>	3.2e6	1	15	3.1e5	2.3e5	4.0e5	3.1e5	0	<i>11e+0</i>	<i>95e-1</i>	<i>13e+0</i>	1.0e7	
1e-1	1e-1	0	<i>30e-2</i>	<i>17e-2</i>	<i>45e-2</i>	2.5e6	
1e-3	1e-3
1e-5	1e-5
1e-8	1e-8
f17 in 5-D, N=15, mFE=5.00e6					f17 in 20-D, N=15, mFE=2.00e7					f18 in 5-D, N=15, mFE=5.00e6					f18 in 20-D, N=15, mFE=2.00e7							
Δf	#	ERT	10%	90%	RT _{succ}	#	ERT	10%	90%	RT _{succ}	#	ERT	10%	90%	RT _{succ}	#	ERT	10%	90%	RT _{succ}		
10	15	2.1e1	1.3e1	3.0e1	2.1e1	15	7.5e3	5.3e3	9.8e3	7.5e3	10	15	1.8e3	9.8e2	2.7e3	1.8e3	0	<i>18e+0</i>	<i>17e+0</i>	<i>20e+0</i>	1.0e7	
1	15	1.8e5	1.2e5	2.4e5	1.8e5	0	<i>50e-1</i>	<i>43e-1</i>	<i>55e-1</i>	6.3e6	1	2	3.6e7	1.8e7	>7e7	5.0e6	
1e-1	0	<i>48e-2</i>	<i>39e-2</i>	<i>57e-2</i>	2.2e6	1e-1	0	<i>15e-1</i>	<i>95e-2</i>	<i>18e-1</i>	2.2e6	
1e-3	1e-3
1e-5	1e-5
1e-8	1e-8
f19 in 5-D, N=15, mFE=5.00e6					f19 in 20-D, N=15, mFE=2.00e7					f20 in 5-D, N=15, mFE=5.00e6					f20 in 20-D, N=15, mFE=2.00e7							
Δf	#	ERT	10%	90%	RT _{succ}	#	ERT	10%	90%	RT _{succ}	#	ERT	10%	90%	RT _{succ}	#	ERT	10%	90%	RT _{succ}		
10	15	3.8e1	2.9e1	4.7e1	3.8e1	15	5.9e5	4.2e5	7.7e5	5.9e5	10	15	4.6e2	3.2e2	6.							

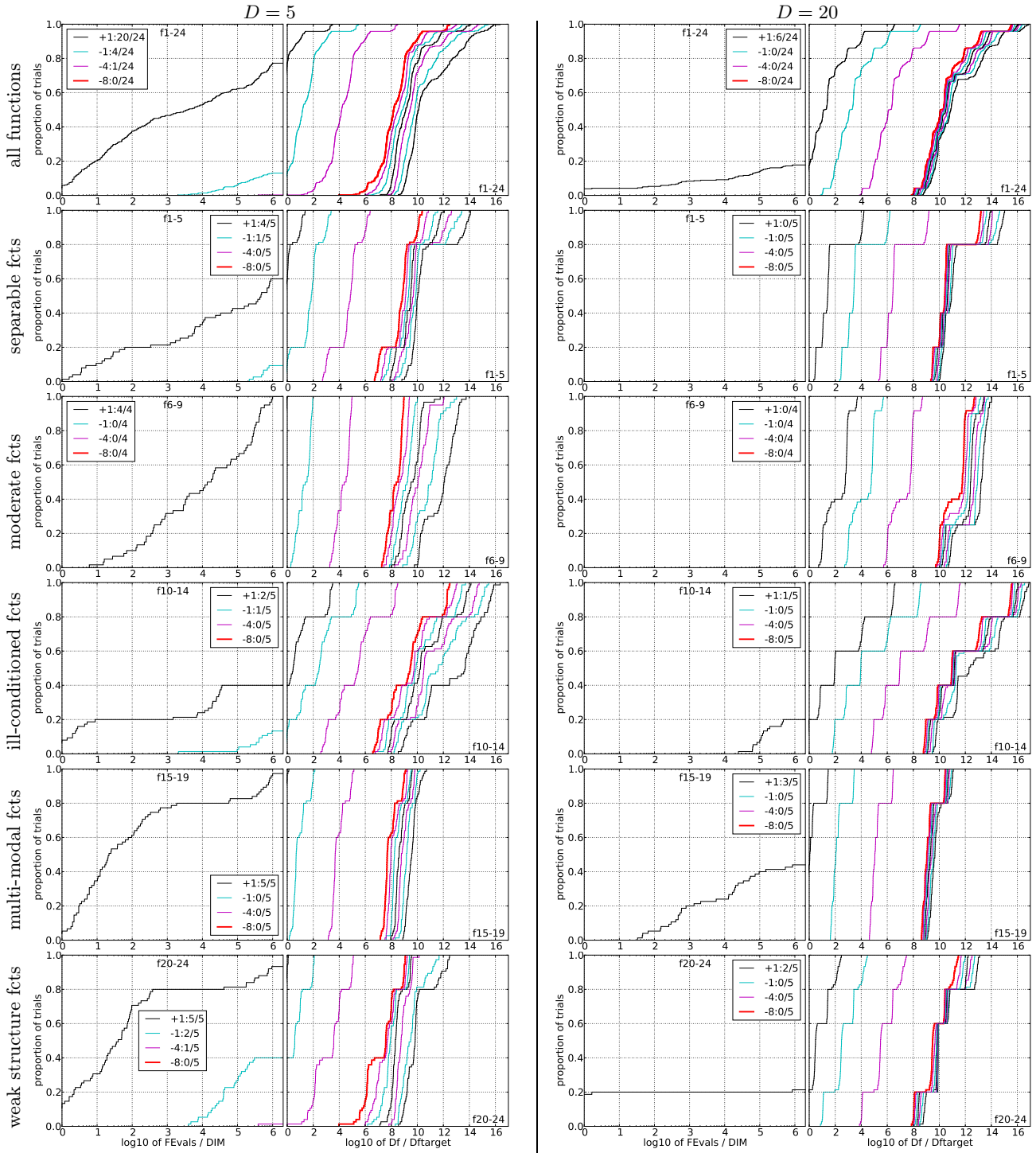


Figure 4: Empirical cumulative distribution functions (ECDFs), plotting the fraction of trials versus running time (left subplots) or versus Δf (right subplots). The thick red line represents the best achieved results. Left subplots: ECDF of the running time (number of function evaluations), divided by search space dimension D , to fall below $f_{\text{opt}} + \Delta f$ 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 $D, 10D, 100D \dots$ function evaluations (from right to left cycling black-cyan-magenta). Top row: all functions; second row: separable functions; third row: misc. moderate functions; fourth row: ill-conditioned functions; fifth row: multi-modal functions with adequate 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.