The Impact of Initial Designs on the Performance of MATSuMoTo on the Noiseless BBOB-2015 Testbed: A Preliminary Study

Dimo Brockhoff, Bernd Bischl, Tobias Wagner

The Impact of Initial Designs on the Performance of MATSuMoTo on the Noiseless BBOB-2015 Testbed: A Preliminary Study

Dimo Brockhoff
INRIA Lille - Nord Europe
DOLPHIN research team
Villeneuve d'Ascq, France
firstname.lastname@inria.fr

Bernd Bischl
LMU München
Institut für Statistik
Munich, Germany
firstname.lastname@stat.uni-muenchen.de

Tobias Wagner
TU Dortmund University
Institute of Machining Technology (ISF)
Dortmund, Germany
lastname@isf.de

ABSTRACT

Most surrogate-assisted algorithms for expensive optimization follow the same framework: After an initial design phase in which the true objective function is evaluated for a few search points, an iterative process builds a surrogate model of the expensive function and, based on the current model, a so-called infill criterion suggests one or more points to be evaluated on the true problem. The evaluations are used to successively update and refine the model. Implementing surrogate-assisted algorithms requires several design choices to be made. It is practically relevant to understand their impact on the algorithms' performance. Here, we start to look at the initial design phase and experimentally investigate the performance of the freely available MATLAB Surrogate Model Toolbox (MATSuMoTo) with regard to the initial design. The results are preliminary in the sense that not all possible choices are investigated, but we can make first well-founded statements about whether Latin Hypercube or uniform random sampling should be preferred and about the effect of the size of the initial design on the performance of MATSuMoTo on the 24 noiseless test functions of the BBOB-2015 test suite.

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

Keywords

Benchmarking, Black-box optimization, Expensive problems

1. INTRODUCTION

Expensive optimization problems in which the evaluation of the objective function might take minutes or hours of (parallelized) optimization time occur frequently in practical applications, e.g., in the case of simulation-based engineering design or in biological and medical applications. Algorithms, especially designed for this expensive optimization scenario where only a few hundreds to thousands of function evaluations are affordable, often follow a common framework. In an initial design phase, a few representative solutions are sampled in the search space and evaluated on the true, expensive objective function. Then, an iterative process builds a surrogate- or meta-model of the objective function, suggests a batch of new solutions to be evaluated on the expensive objective function based on the model and a user-defined infill criterion, and uses the new evaluated solutions to again update the model. The actual implementation of an algorithm for an expensive optimization problem, requires a choice among several building blocks for each part of this iterative process. Many surrogate models and several infill criteria have been proposed in the literature [16].

Also for the initial design, several approaches have been suggested. Most often, Latin Hypercube sampling (LHS, [12]) is employed, but also uniform random or quasi-random sampling methods such as Halton or Sobol sequences [6, 15] can be applied. Over the last 15 years, many studies have been conducted in order to find the optimal design for fitting surrogate models. Koehler and Owen [11] have provided an overview of theoretical results and visual examples. Comparative studies [12, 4] can only conclude that LHS is superior to random sampling in some cases.

With respect to size of the initial design, the recommendation $10 \cdot \mathrm{DIM}$ was once established [10] with DIM being the search space dimension. More recent studies have shown that also much smaller initial designs can produce good results given that the modeled response has not too many local optima or is not too irregular [2, 9].

To measure and visualize performance, several benchmarking suites have been proposed of which the Black-box Optimization Benchmarking suite (BBOB) is one of the most sophisticated ones [7]. The underlying Coco software allows to automatically collect data during the optimization runs and to produce comparison plots and result tables with a single python script. Due to the simplicity of algorithm benchmarking, we will employ the BBOB framework here.

With respect to the algorithm implementations, we rely on the MATLAB Surrogate Model Toolbox (MATSuMoTo) which has recently been provided for free to the research community [13] and which allows to easily plug in and test several initial designs, surrogate models, and infill criteria. Note that the default version of the MATSuMoTo library has been already benchmarked on the BBOB testbed [3] without investigating the impact of its many parameters and algorithm design choices.

Our Contribution.

In this paper, we provide a first preliminary experimental investigation of the impact of the initial design on the performance of MATSuMoTo on the BBOB testbed. In particular, we try to answer the following research questions:

Q1: What is the effect of having larger initial designs, such as two times or ten times the default value of $2\cdot(\text{DIM}+1)$ function evaluations?

Q2: What is the effect of replacing the LHS with simple (uniform pseudo-)random sampling?

Both questions will be answered empirically by comparing different MATSuMoTo variants on the 24 noiseless BBOB functions of [5].

2. ALGORITHM PRESENTATION

As baseline algorithm, we use the MATSuMoTo library in its default setting (January 2014 from https://ccse.lbl.gov/people/julianem/). Note that during our experiments, a small bug affecting the restarting mechanism of MATSuMoTo has been fixed such that the source code differs slightly from the original MATSuMoTo library.[1]. The default setting of the MATSuMoTo library—as already previously benchmarked on the BBOB testbed [3]—employs LHS as initial design for $2\cdot(\text{DIM}+1)$ function evaluations, a cubic Radial Basis Functions surrogate model, and an infill criterion which either chooses the next point either as a (small) random perturbation around the model's minimum (exploitation) or, with a certain probability, uniformly at random in the whole variable domain (exploration).

In order to investigate the impact of the initial design on the library's performance, we run four different versions of the algorithm that only differ in the type and size of the initial design as indicated in Table 1. The table also introduces the abbreviations for the algorithms used in the following. The default version, LHD-Default corresponds to the algorithm benchmarked in [3] and employs an LHS of size $2\cdot(\text{DIM}+1)$ for fitting the first surrogate model. The algorithms LHD-2xDefault and LHD-10xDefault differ from LHD-Default only in the size of the initial design which is two times and ten times larger than the default, respectively. Finally, RAND-2xDefault samples the initial points in the bounded search space uniformly at random for the same number of function evaluations as LHD-2xDefault. All algorithm variants are run in total for $50\cdot\text{DIM}$ function evaluations and otherwise use the default settings of the library. The upper and lower bounds for the initial sampling within MATSuMoTo are set to -5 and 5, respectively, according to the definition of the BBOB test functions which guarantees that the global optimum lies within this range.

[1]The source code for reproducing our results can be found on the Coco/BBOB web page under http://coco.gforge.inria.fr/doku.php?id=bbob-2015-results.

algorithm name	initial design	length of initial design phase
LHD-Default	LHS	default
LHD-2xDefault	LHS	2x default
RAND-2xDefault	random	2x default
LHD-10xDefault	LHS	10x default

Table 1: Overview of the algorithms investigated. LHS: Latin hypercube design, default length of initial design phase is $2\cdot(\textbf{DIM}+1)$.

	2-D	3-D	5-D	10-D	20-D
LHD-Default	0.0044	0.0040	0.16	0.59	1.9
LHD-2xDefault	0.0048	0.0013	0.20	0.54	1.8
RAND-2xDefault	0.0068	0.031	0.22	0.56	1.8
LHD-10xDefault	0.0021	0.013	0.14	0.41	1.4

Table 2: Running times per function evaluation on function f_8 (in seconds).

3. CPU TIMING

In order to evaluate the CPU time, we have run all four variants of MATSuMoTo on BBOB function f_8 with restarts for at least 30 seconds *and* for at least $50\cdot\text{DIM}$ function evaluations. The code was run on a Mac Intel(R) Core(TM) i7-2635QM CPU @ 2GHz with 1 processor, 4 cores, and 8GB of RAM. The time per function evaluation in seconds for all algorithms and dimensions are given in Table 2. We observe that the runtime per function evaluation does not differ much between the algorithm variants, only the large design results in slightly shorter times. With increasing dimension, however, the computation time per function evaluation increases drastically. Nevertheless, with less than two seconds in 20-D, the computation time of the algorithm itself is neglectable in an expensive optimization scenario.

4. EXPERIMENTAL RESULTS

Results from experiments according to [7] on the benchmark functions given in [5, 8] are presented in Figures 1, 2 and 3 and in Tables 3 and 4. The **expected running time (ERT)**, used in the figures and tables, depends on a given target value, $f_t = f_{\text{opt}} + \Delta f$, and is computed as sum over all evaluations executed in each run until $f_t$has been obtained divided by the number of trials that actually reached f_t [7, 14]. **Statistical significance** is tested with the rank-sum test for a given target Δf_t using, for each trial, either the number of function evaluations to reach Δf_t (inverted and multiplied by -1), or, if the target was not reached, the best Δf-value and the smallest number of overall function evaluations for any unsuccessful trial under consideration.

Besides the mentioned LHD-Default, LHD-2xDefault, LHD-10xDefault, and RAND-2xDefault, the pure random search of [1] and SMAC [9], an improved version of the Efficient Global Optimization (EGO) algorithm, have been used as a baseline algorithms. In the following, we investigate the figures and tables, provided by the BBOB framework, with the two main research question introduced in Sec. 1 in mind.

4.1 Impact of the Size of the Initial Design

Comparing the algorithms, which only differ in the size of the initial design (LHD-Default, LHD-2xDefault, and LHD-10xDefault), we can state two main observations.

First, during the initial design phase, all three algorithms follow, as expected, roughly the same trend in the empirical cumulative distribution plots, which resembles the performance of the pure random search. The actual improvements start with the proposal of new search points according to the model. Increasing the initial design by a factor of ten results in a maximum performance loss at this time of about a factor of 3 in 5-D and of about a factor of 6 in 20-D.

Second, a smaller initial design seems to be beneficial—for sure in an any-time scenario, but also for the final budget of 50·DIM function evaluations. In accordance with [2], the exception is the class of multi-modal problems (f_{15}–f_{19}) where LHD-10xDefault takes the lead in the empirical cumulative distribution function plots of Fig. 2 and Fig. 3 from about 30–40·DIM function evaluations. The results are not significant, however, and the effect is mainly caused by a reduced variance in the expected running time for the more difficult targets. For the smaller budget of function evaluations, the trend of a small initial design being favorable is supported by the performance of SMAC which tries to dispense with the initial design [9]. It is outperforming all other compared algorithms in the beginning of the optimization.

Considering question Q1 from above, we can conclude that smaller initial designs seem beneficial on the BBOB testbed and that, among the chosen settings, the default initial design size works best. We expect, however, that even smaller initial designs might further improve the performance.

4.2 Impact of the Type of Initial Design

Regarding question Q2 from above, we can state that there is no significant difference between LHS and random sampling when the initial design size of MATSuMoTo is set to twice the default value. When looking at the Empirical Cumulative Distribution Functions, aggregating over all 24 test functions, the graphs for LHD-2xDefault and RAND-2xDefault are almost superimposed. Even if the interaction with the initial design size cannot be evaluated based on the reported experiments, initial experiments, not shown here, showed the same effect for other sizes of the initial design.

5. CONCLUSIONS

Preliminary experiments on the impact of MATSuMoTo's initial design on its performance on the noiseless BBOB testbed revealed that (i) a longer initial design phase seems detrimental for the any-time performance and that (ii) replacing the default LHS with uniform (pseudo-)random sampling does not change much for the algorithm's performance. The results are the first of this kind on the BBOB test suite and preliminary in the sense that the MATSuMoTo also offers *symmetric* LHS and an two-level factorial designs. Based on our results, however, these options will have less effect compared to the design size.

Acknowledgments: The authors acknowledge support by the French National Research Agency (ANR) under grant ANR-12-MONU-0009 (NumBBO).

6. REFERENCES

[1] A. Auger and R. Ros. Benchmarking the pure random search on the BBOB-2009 testbed. In F. Rothlauf, editor, *GECCO (Companion)*, pages 2479–2484. ACM, 2009.

[2] T. Bartz-Beielstein and M. Preuß. Considerations of budget allocation for sequential parameter optimization (SPO). In L. Paquete et al., editors, *Proc. Workshop on Empirical Methods for the Analysis of Algorithms (EMAA 2006)*, pages 35–40, 2006.

[3] D. Brockhoff. Comparison of the MATSuMoTo Library for Expensive Optimization on the Noiseless Black-Box Optimization Benchmarking Testbed. In *Congress on Evolutionary Computation (CEC 2015)*. IEEE Press, 2015. accepted.

[4] D. Bursztyn and D. Steinberg. Comparison of designs for computer experiments. *Statistical Planning and Inference*, 136(3):1103–1119, 2006.

[5] 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. Updated February 2010.

[6] J. H. Halton. Algorithm 247: Radical-inverse Quasi-random Point Sequence. *Communications of the ACM*, 7(12):701–702, Dec. 1964.

[7] N. Hansen, A. Auger, S. Finck, and R. Ros. Real-parameter black-box optimization benchmarking 2012: Experimental setup. Technical report, INRIA, 2012.

[8] 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. Updated February 2010.

[9] F. Hutter, H. Hoos, and K. Leyton-Brown. An Evaluation of Sequential Model-Based Optimization for Expensive Blackbox Functions. In *GECCO workshop on Black-Box Optimization Benchmarking (BBOB'2013)*, pages 1209–1216. ACM Press, 2013.

[10] D. R. Jones, M. Schonlau, and W.-J. Welch. Efficient Global Optimization of Expensive Black-Box Functions. *Journal of Global Optimization*, 13(4):455–492, 1998.

[11] J. R. Koehler and A. B. Owen. Computer experiments. In S. Ghosh and C. R. Rao, editors, *Handbook of Statistics*, volume 13, pages 261–308. Elsevier, New York, NY, 1996.

[12] M. D. McKay, R. J. Beckman, and W. J. Conover. A Comparison of Three Methods for Selecting Values of Input Variables in the Analysis of Output from a Computer Code. *Technometrics*, 21:239–245, 1979.

[13] J. Müller. MATSuMoTo: The MATLAB Surrogate Model Toolbox For Computationally Expensive Black-Box Global Optimization Problems. *ArXiv e-prints*, Apr. 2014.

[14] K. Price. Differential evolution vs. the functions of the second ICEO. In *Proceedings of the IEEE International Congress on Evolutionary Computation*, pages 153–157, 1997.

[15] I. M. Sobol. On the Distribution of Points in a Cube and the Approximate Evaluation of Integrals. *USSR Computational Mathematics and Mathematical Physics*, 7(4):86–112, 1967.

[16] F. A. C. Viana, T. W. Simpson, V. Balabanov, and V. Toropov. Metamodeling in multidisciplinary design optimization: How far have we really come? *AIAA Journal*, 52:670–690, 2014.

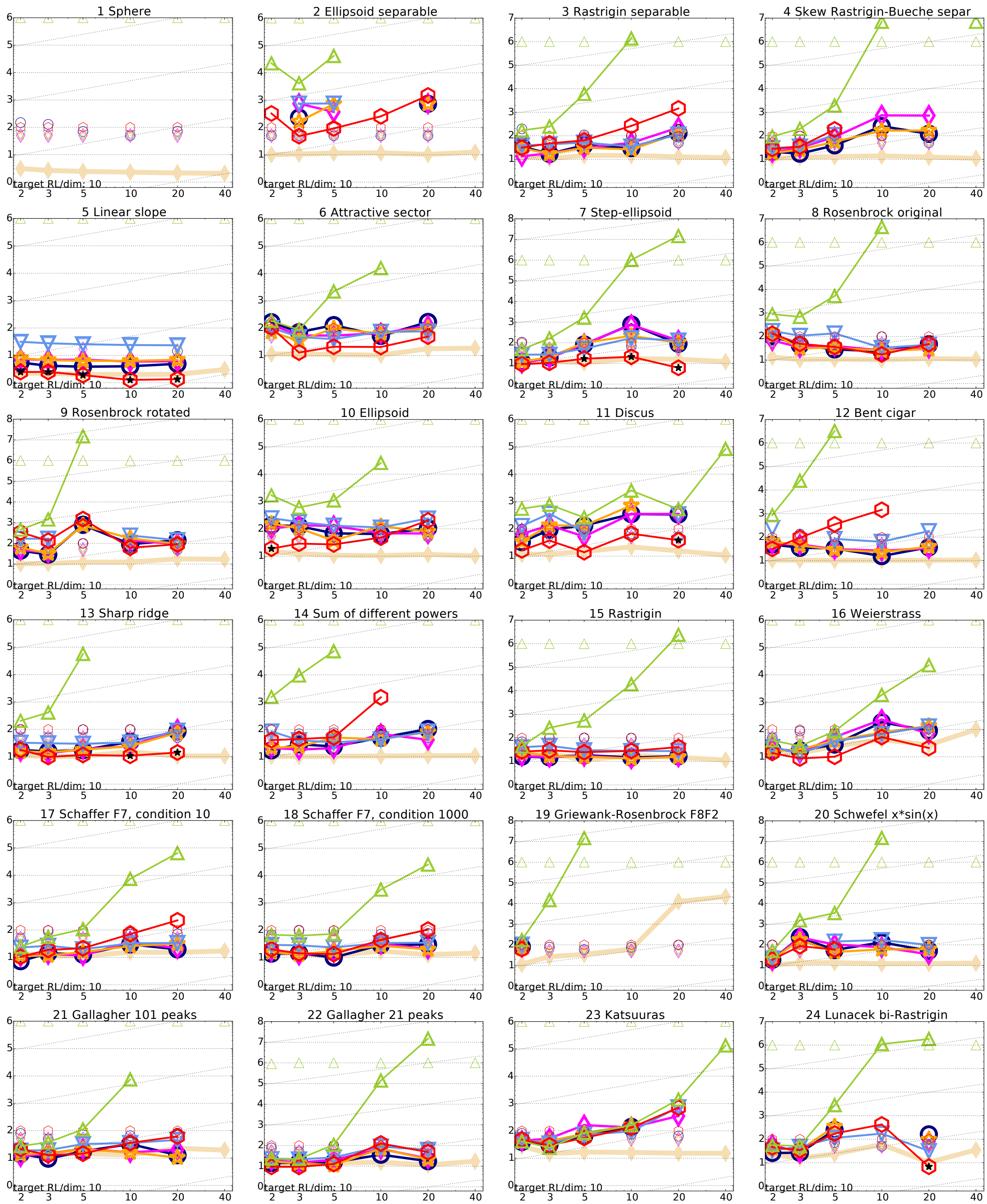

Figure 1: Expected running time (ERT in number of f-evaluations as $\log_{10}$ value) divided by dimension versus dimension. The target function value is chosen such that the bestGECCO2009 artificial algorithm just failed to achieve an ERT of $10 \times \mathrm{DIM}$. Different symbols correspond to different algorithms given in the legend of f_1 and f_{24}. Light symbols give the maximum number of function evaluations from the longest trial divided by dimension. Black stars indicate a statistically better result compared to all other algorithms with $p < 0.01$ and Bonferroni correction number of dimensions (six). Legend: ○:LHD-Default, ◇:LHD-2xDefault, ⋆:RAND-2xDefault, ▽:LHD-10xDefault, ⬡:SMAC, △:RANDOMSEARCH

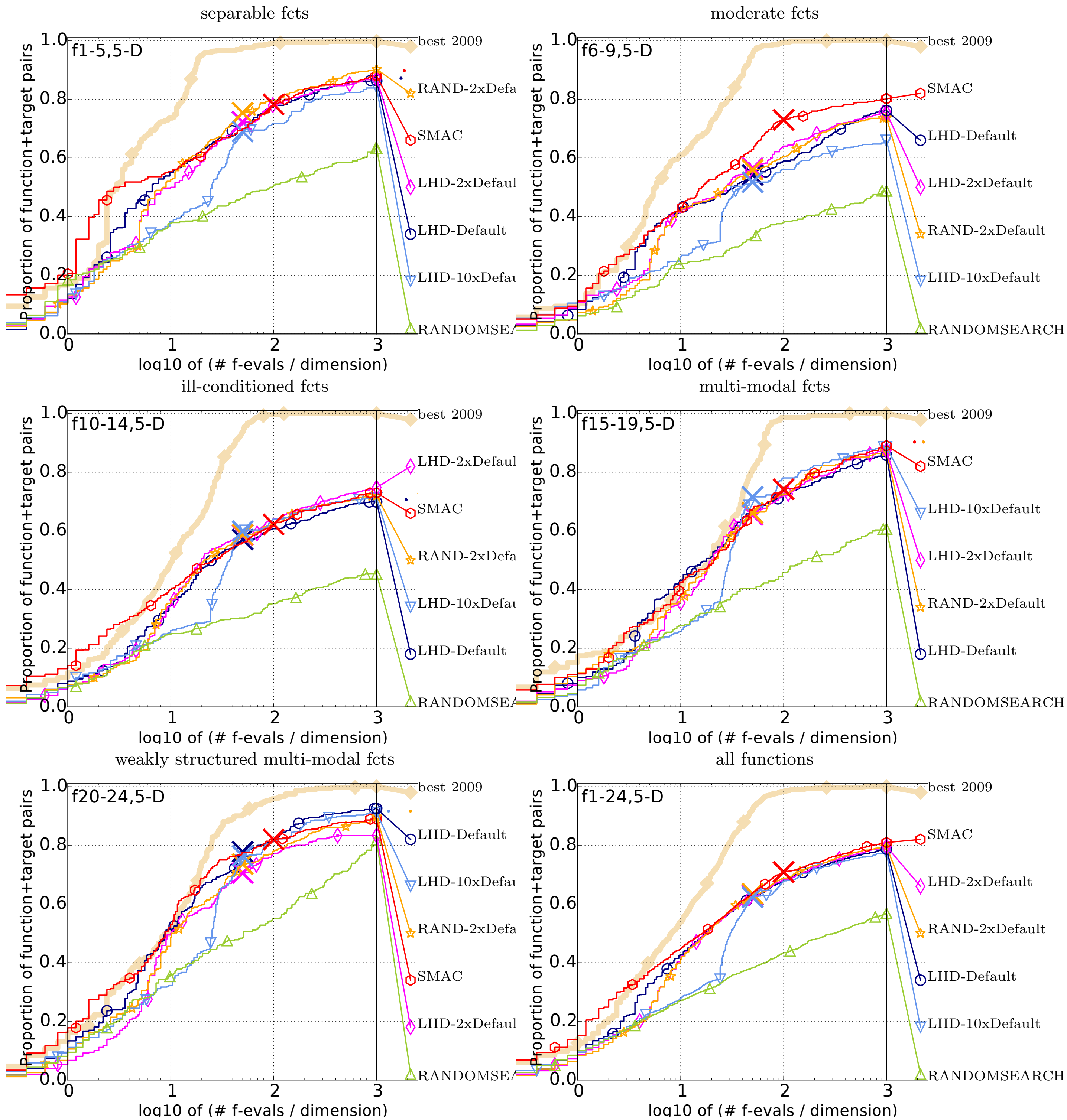

Figure 2: Bootstrapped empirical cumulative distribution of the number of objective function evaluations divided by dimension (FEvals/DIM) for all functions and subgroups in 5-D. The targets are chosen from $10^{[-8..2]}$ such that the bestGECCO2009 artificial algorithm just not reached them within a given budget of k × DIM, with $k \in \{0.5, 1.2, 3, 10, 50\}$. The "best 2009" line corresponds to the best ERT observed during BBOB 2009 for each selected target.

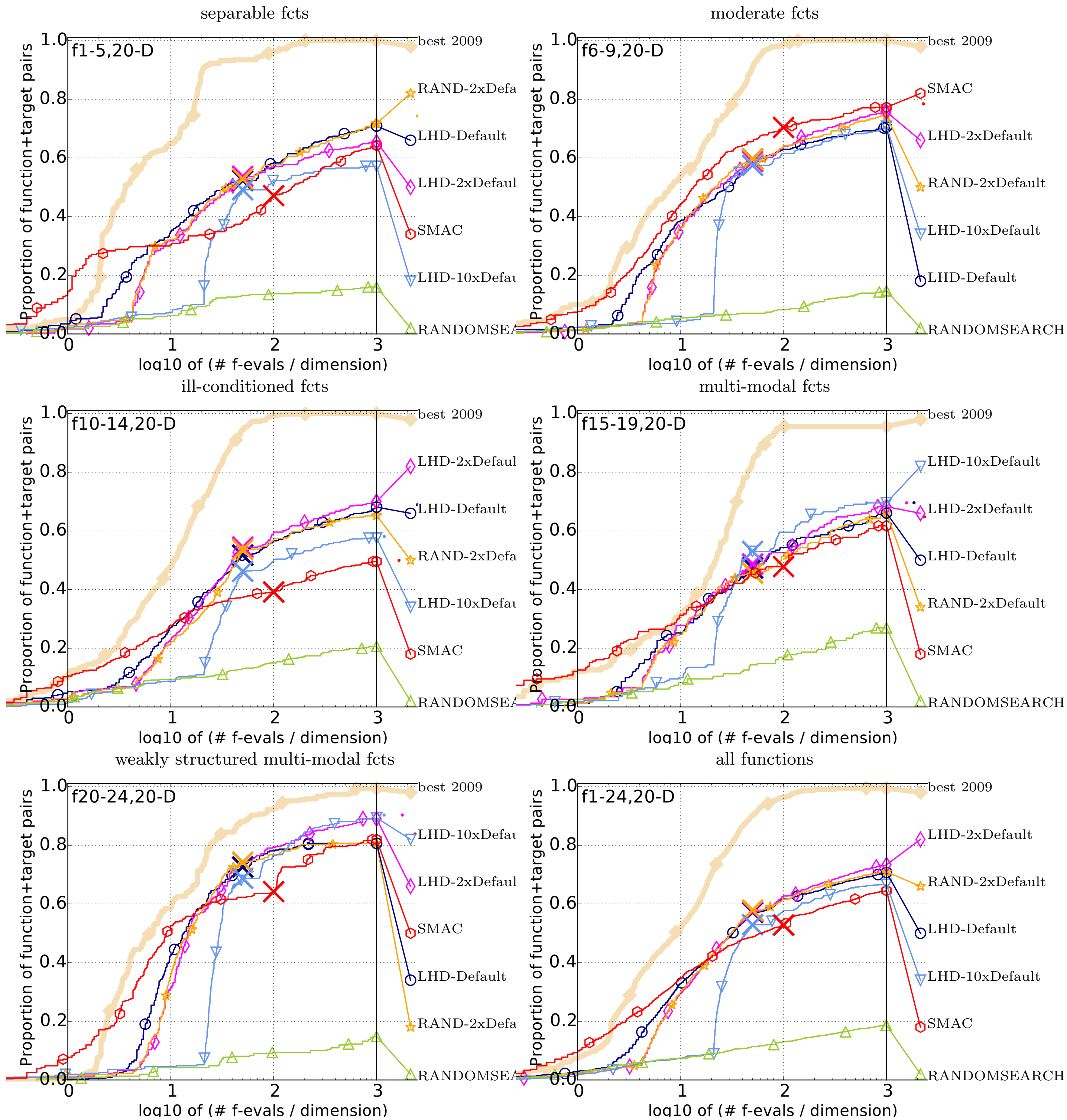

Figure 3: Bootstrapped empirical cumulative distribution of the number of objective function evaluations divided by dimension (FEvals/DIM) for all functions and subgroups in 20-D. The targets are chosen from $10^{[-8..2]}$ such that the bestGECCO2009 artificial algorithm just not reached them within a given budget of k × DIM, with $k \in \{0.5, 1.2, 3, 10, 50\}$. The "best 2009" line corresponds to the best ERT observed during BBOB 2009 for each selected target.

#FEs/D	0.5	1.2	3	10	50	#succ
f1	*2.5e+1*:4.8	*1.6e+1*:7.6	*1.0e-8*:12	*1.0e-8*:12	*1.0e-8*:12	15/15
LHD-Default	1.8(1)	1.7(0.8)	∞	∞	∞ *250*	0/15
LHD-2xDefault	2.6(2)	2.8(1)	∞	∞	∞ *250*	0/15
RAND-2xDefault	2.0(2)	2.2(1)	∞	∞	∞ *250*	0/15
LHD-10xDefault	2.8(8)	3.8(1)	∞	∞	∞ *250*	0/15
SMAC	**0.79**(0.7)	**0.84**(0.4)	∞	∞	∞ *500*	0/15
RANDOMSEARCH	1.8(1)	2.5(3)	∞	∞	∞ *5e6*	0/15

#FEs/D	0.5	1.2	3	10	50	#succ
f2	*1.6e+6*:2.9	*4.0e+5*:11	*4.0e+4*:15	*6.3e+2*:58	*1.0e-8*:95	15/15
LHD-Default	1.6(0.7)	**0.68**(0.4)	4.0(6)	∞	∞ *250*	0/15
LHD-2xDefault	1.3(1.0)	0.79(1)	4.3(3)	32(26)	∞ *250*	0/15
RAND-2xDefault	1.6(2)	1.0(0.5)	3.3(2)	62(50)	∞ *250*	0/15
LHD-10xDefault	1.1(0.8)	1.2(2)	8.4(2)	65(114)	∞ *250*	0/15
SMAC	**1.0**(1)	0.74(0.4)	**1.7**(2)	**8.0**(3)	∞ *500*	0/15
RANDOMSEARCH	**1.0**(0.3)	1(0.8)	13(24)	3591(2077)	∞ *5e6*	0/15

#FEs/D	0.5	1.2	3	10	50	#succ
f3	*1.6e+2*:4.1	*1.0e+2*:15	*6.3e+1*:23	*2.5e+1*:73	*1.0e+1*:716	15/15
LHD-Default	1.8(2)	1.2(1)	1.8(2)	2.7(2)	1.1(0.8)	4/15
LHD-2xDefault	2.1(2)	1.5(1)	1.9(0.9)	**2.1**(1)	2.5(2)	2/15
RAND-2xDefault	2.3(2)	1.4(0.7)	**1.6**(0.7)	2.2(1)	**0.58**(0.5)	7/15
LHD-10xDefault	2.1(2)	2.8(4)	3.7(3)	3.6(2)	1.0(0.6)	5/15
SMAC	**0.73**(0.8)	**0.74**(0.5)	2.6(4)	4.4(2)	5.1(5)	2/15
RANDOMSEARCH	1(0.4)	2.1(2)	17(16)	416(509)	6763(1e4)	10/15

#FEs/D	0.5	1.2	3	10	50	#succ
f4	*2.5e+2*:2.6	*1.6e+2*:10	*1.0e+2*:19	*4.0e+1*:65	*1.6e+1*:434	15/15
LHD-Default	2.6(2)	1.1(0.7)	2.9(3)	**3.0**(2)	9.0(17)	1/15
LHD-2xDefault	2.2(1)	1.5(2)	3.0(3)	6.8(4)	∞ *250*	0/15
RAND-2xDefault	2.8(3)	2.2(2)	3.1(2)	4.2(2)	**8.4**(9)	1/15
LHD-10xDefault	3.9(4)	3.2(4)	5.1(3)	8.3(11)	∞ *250*	0/15
SMAC	**0.56**(0.7)	**0.54**(1.0)	**1.8**(3)	14(11)	∞ *500*	0/15
RANDOMSEARCH	2.3(2)	1.8(2)	3.4(7)	148(115)	4682(4091)	13/15

#FEs/D	0.5	1.2	3	10	50	#succ
f5	*6.3e+1*:4.0	*4.0e+1*:10	*1.0e-8*:10	*1.0e-8*:10	*1.0e-8*:10	15/15
LHD-Default	1.6(0.9)	1.2(0.6)	1.9(0.5)	1.9(0.6)	1.9(0.6)	15/15
LHD-2xDefault	1.9(1)	1.8(1)	3.5(0.5)	3.5(3)	3.5(2)	15/15
RAND-2xDefault	2.0(2)	2.0(1)	3.1(0.1)	3.1(0.2)	3.1(0.2)	15/15
LHD-10xDefault	3.2(5)	4.6(2)	13(0.2)	13(0.2)	13(0.1)	15/15
SMAC	**1.3**(0.2)	**0.63**(0.2)*	**0.95**(0.1)$^{\star 4}$	**0.95**(0.1)$^{\star 4}$	**0.95**(0.2)$^{\star 4}$	15/15
RANDOMSEARCH	2.1(3)	3.8(6)	∞	∞	∞ *5e6*	0/15

#FEs/D	0.5	1.2	3	10	50	#succ
f6	*1.0e+5*:3.0	*2.5e+4*:8.4	*1.0e+2*:16	*2.5e+1*:54	*2.5e-1*:254	15/15
LHD-Default	**1.3**(2)	**0.90**(0.7)	1.7(1)	12(12)	∞ *250*	0/15
LHD-2xDefault	1.7(1)	1.3(1)	3.2(0.8)	4.9(4)	∞ *250*	0/15
RAND-2xDefault	2.3(2)	1.8(2)	2.8(4)	9.0(9)	∞ *250*	0/15
LHD-10xDefault	1.6(1)	2.2(4)	5.5(3)	3.6(5)	∞ *250*	0/15
SMAC	1.4(1)	1.1(0.7)	**1.5**(1)	**1.9**(0.9)	∞ *500*	0/15
RANDOMSEARCH	3.3(11)	5.1(10)	475(133)	203(924)	∞ *5e6*	0/15

#FEs/D	0.5	1.2	3	10	50	#succ
f7	*1.6e+2*:4.2	*1.0e+2*:6.2	*2.5e+1*:20	*4.0e+0*:54	*1.0e+0*:324	15/15
LHD-Default	1.3(1)	1.8(1)	1.5(1)	7.6(5)	5.4(6)	2/15
LHD-2xDefault	1.4(0.5)	2.0(3)	1.8(0.7)	8.0(5)	11(15)	1/15
RAND-2xDefault	**1.3**(2)	1.7(1)	2.5(2)	9.3(18)	11(11)	1/15
LHD-10xDefault	1.5(2)	2.2(3)	5.2(1)	4.8(2)	5.5(6)	2/15
SMAC	1.3(1.0)	**1.1**(2)	**1.5**(1)	**1.6**(1.0)*	**0.88**(0.3)	13/15
RANDOMSEARCH	2.0(2)	2.9(4)	8.8(3)	151(148)	1207(1052)	15/15

#FEs/D	0.5	1.2	3	10	50	#succ
f8	*1.0e+4*:4.6	*6.3e+3*:6.8	*1.0e+3*:18	*6.3e+1*:54	*1.6e+0*:258	15/15
LHD-Default	1.7(2)	1.9(1)	1.4(0.4)	**2.6**(4)	∞ *250*	0/15
LHD-2xDefault	2.4(2)	2.3(2)	1.5(0.5)	3.6(6)	∞ *250*	0/15
RAND-2xDefault	3.1(2)	2.3(2)	1.5(0.7)	3.1(3)	∞ *250*	0/15
LHD-10xDefault	2.7(4)	2.8(3)	5.2(3)	12(18)	∞ *250*	0/15
SMAC	**0.99**(2)	**0.91**(0.7)	**1.2**(1.0)	3.3(2)	∞ *500*	0/15
RANDOMSEARCH	3.0(4)	3.1(3)	10(10)	482(443)	∞ *5e6*	0/15

#FEs/D	0.5	1.2	3	10	50	#succ
f9	*2.5e+1*:20	*1.6e+1*:26	*1.0e+1*:35	*4.0e+0*:62	*1.6e-2*:256	15/15
LHD-Default	18(22)	34(27)	35(40)	64(84)	∞ *250*	0/15
LHD-2xDefault	12(8)	12(15)	25(35)	∞	∞ *250*	0/15
RAND-2xDefault	**11**(20)	**11**(14)	20(11)	**60**(38)	∞ *250*	0/15
LHD-10xDefault	185(211)	144(39)	∞	∞	∞ *250*	0/15
SMAC	14(6)	12(3)	**12**(6)	120(150)	∞ *500*	0/15
RANDOMSEARCH	7845(5326)	2.3e4(4e4)	4.1e4(3e4)	1.2e6(2e6)	∞ *5e6*	0/15

#FEs/D	0.5	1.2	3	10	50	#succ
f10	*2.5e+6*:2.9	*6.3e+5*:7.0	*2.5e+5*:17	*6.3e+3*:54	*2.5e+1*:297	15/15
LHD-Default	1.5(2)	1.9(1)	1.4(0.9)	6.4(11)	∞ *250*	0/15
LHD-2xDefault	1.9(2)	1.4(1)	1.6(2)	12(25)	∞ *250*	0/15
RAND-2xDefault	2.3(2)	2.1(2)	1.5(1)	3.5(2)	∞ *250*	0/15
LHD-10xDefault	1.5(3)	1.7(2)	1.4(1)	12(7)	∞ *250*	0/15
SMAC	**1.3**(0.7)	**0.80**(0.5)	**0.58**(0.5)	**2.5**(2)	∞ *500*	0/15
RANDOMSEARCH	1.6(2)	1.5(2)	1.6(0.9)	102(169)	**2.5e5**(2e5)	1/15

#FEs/D	0.5	1.2	3	10	50	#succ
f11	*1.0e+6*:3.0	*6.3e+4*:6.2	*6.3e+2*:16	*6.3e+1*:74	*6.3e-1*:298	15/15
LHD-Default	1.4(2)	2.5(3)	4.7(3)	8.9(15)	∞ *250*	0/15
LHD-2xDefault	1.5(1)	3.0(3)	4.1(3)	3.5(4)	∞ *250*	0/15
RAND-2xDefault	1.6(0.4)	3.6(3)	5.8(3)	8.7(11)	∞ *250*	0/15
LHD-10xDefault	1.7(2)	3.1(2)	6.1(6)	5.2(5)	∞ *250*	0/15
SMAC	**0.73**(0.3)	**0.94**(0.9)	**1.9**(2)	**0.94**(1)	∞ *500*	0/15
RANDOMSEARCH	1.8(0.3)	2.5(2)	6.8(9)	17(13)	**1.2e5**(1e5)	2/15

#FEs/D	0.5	1.2	3	10	50	#succ
f12	*4.0e+7*:3.6	*1.6e+7*:7.6	*4.0e+6*:19	*1.6e+4*:52	*1.0e+0*:268	15/15
LHD-Default	1.2(0.9)	1.4(0.7)	1.7(0.7)	3.2(1)	∞ *250*	0/15
LHD-2xDefault	1.5(0.8)	1.9(1)	**1.4**(0.9)	3.0(1)	∞ *250*	0/15
RAND-2xDefault	1.4(2)	2.0(2)	1.5(0.3)	**2.9**(1)	∞ *250*	0/15
LHD-10xDefault	1.1(0.6)	2.8(3)	4.5(2)	8.2(3)	∞ *250*	0/15
SMAC	**0.57**(0.8)	**1.3**(0.5)	3.6(8)	34(21)	∞ *500*	0/15
RANDOMSEARCH	2.2(2)	4.0(4)	16(18)	3.1e5(4e5)	∞ *5e6*	0/15

#FEs/D	0.5	1.2	3	10	50	#succ
f13	*1.0e+3*:2.8	*6.3e+2*:8.4	*4.0e+2*:17	*6.3e+1*:52	*6.3e-2*:264	15/15
LHD-Default	2.0(2)	1.8(0.8)	1.5(0.6)	1.7(0.2)	∞ *250*	0/15
LHD-2xDefault	2.1(3)	2.0(1)	1.9(0.4)	1.8(2)	∞ *250*	0/15
RAND-2xDefault	2.0(4)	1.8(1)	1.8(0.4)	1.7(0.6)	∞ *250*	0/15
LHD-10xDefault	2.6(5)	3.0(1)	4.8(3)	2.9(0.5)	∞ *250*	0/15
SMAC	**1**(1)	**1.1**(1)	**0.96**(0.4)	**1.1**(0.3)	∞ *500*	0/15
RANDOMSEARCH	1.9(3)	4.5(6)	11(10)	5483(7583)	∞ *5e6*	0/15

#FEs/D	0.5	1.2	3	10	50	#succ
f14	*1.6e+1*:3.0	*1.0e+1*:10	*6.3e+0*:15	*2.5e-1*:53	*1.0e-5*:251	15/15
LHD-Default	2.1(1)	1.4(1)	1.3(1)	2.3(2)	∞ *250*	0/15
LHD-2xDefault	2.2(2)	1.5(1)	1.4(0.7)	**2.0**(1)	∞ *250*	0/15
RAND-2xDefault	3.0(4)	1.4(2)	1.7(1)	4.8(6)	∞ *250*	0/15
LHD-10xDefault	2.0(2)	1.2(2)	2.2(2)	3.2(0.5)	∞ *250*	0/15
SMAC	**1.2**(1)	**0.62**(0.9)	**0.76**(0.6)	4.9(2)	∞ *500*	0/15
RANDOMSEARCH	2.2(2)	1.2(0.8)	1.6(0.9)	6945(7637)	∞ *5e6*	0/15

#FEs/D	0.5	1.2	3	10	50	#succ
f15	*1.6e+2*:3.0	*1.0e+2*:13	*6.3e+1*:24	*4.0e+1*:55	*1.6e+1*:289	5/5
LHD-Default	2.4(2)	1.3(0.4)	1.6(3)	1.7(2)	**0.97**(1)	10/15
LHD-2xDefault	2.1(2)	1.5(1)	1.5(0.3)	1.5(1)	1.0(1)	9/15
RAND-2xDefault	3.7(3)	1.6(0.8)	**1.3**(0.7)	**1.4**(0.8)	1.2(2)	8/15
LHD-10xDefault	4.4(6)	2.9(2)	3.8(0.5)	2.7(0.5)	1.3(0.9)	9/15
SMAC	**1.1**(2)	**0.83**(0.3)	1.6(1)	2.2(2)	8.1(9)	3/15
RANDOMSEARCH	3.4(7)	1.7(2)	10(8)	49(41)	1083(772)	15/15

#FEs/D	0.5	1.2	3	10	50	#succ
f16	*4.0e+1*:4.8	*2.5e+1*:16	*1.6e+1*:46	*1.0e+1*:120	*4.0e+0*:334	15/15
LHD-Default	1.4(2)	0.86(0.4)	1.2(1)	1.2(2)	3.9(1)	3/15
LHD-2xDefault	2.5(3)	1.4(1)	1.6(2)	2.2(3)	11(19)	1/15
RAND-2xDefault	**1.3**(0.7)	1.2(1)	1.8(1)	1.7(1)	1.3(1)	7/15
LHD-10xDefault	1.4(1)	1.6(2)	1.9(2)	1.5(2)	1.4(1)	7/15
SMAC	1.7(1)	**0.77**(0.4)	**0.53**(0.5)	**0.42**(0.2)	**0.45**(0.5)	15/15
RANDOMSEARCH	1.6(2)	1.5(2)	1.9(0.8)	3.5(7)	19(28)	15/15

#FEs/D	0.5	1.2	3	10	50	#succ
f17	*1.0e+1*:5.2	*6.3e+0*:26	*4.0e+0*:57	*2.5e+0*:110	*6.3e-1*:412	15/15
LHD-Default	3.1(5)	1.2(1)	1.1(0.7)	1.9(1)	9.2(12)	1/15
LHD-2xDefault	2.4(0.8)	1.4(2)	**1.0**(0.7)	**1.1**(1.0)	8.9(8)	1/15
RAND-2xDefault	2.3(2)	**1.2**(1)	1.4(1)	1.8(2)	4.3(5)	2/15
LHD-10xDefault	**2.1**(1)	2.4(2)	1.8(1)	1.4(0.4)	4.4(6)	2/15
SMAC	2.5(3)	1.6(2)	1.9(1.0)	2.1(3)	**2.5**(4)	6/15
RANDOMSEARCH	4.0(5)	3.3(4)	10(8)	32(45)	2314(1858)	15/15

#FEs/D	0.5	1.2	3	10	50	#succ
f18	*6.3e+1*:3.4	*4.0e+1*:7.2	*2.5e+1*:20	*1.6e+1*:58	*1.6e+0*:318	15/15
LHD-Default	1.9(2)	1.4(1)	1.1(0.6)	**0.85**(0.8)	∞ *250*	0/15
LHD-2xDefault	1.3(2)	1.6(2)	2.2(1)	1.3(1)	**5.5**(11)	2/15
RAND-2xDefault	1.4(0.7)	2.2(2)	1.7(0.8)	1.5(1)	12(14)	1/15
LHD-10xDefault	2.6(2)	3.4(4)	3.3(3)	2.0(0.8)	12(8)	1/15
SMAC	1.1(1)	**0.85**(0.4)	**0.97**(0.9)	1.1(2)	11(9)	2/15
RANDOMSEARCH	**1**(2)	3.3(4)	1.7(2)	6.3(8)	1.7e4(1e4)	9/15

#FEs/D	0.5	1.2	3	10	50	#succ
f19	*1.6e-1*:172	*1.0e-1*:242	*6.3e-2*:675	*4.0e-2*:3078	*2.5e-2*:4946	15/15
LHD-Default	∞	∞	∞	∞	∞ *250*	0/15
LHD-2xDefault	∞	∞	∞	∞	∞ *250*	0/15
RAND-2xDefault	∞	∞	∞	∞	∞ *250*	0/15
LHD-10xDefault	∞	∞	∞	∞	∞ *250*	0/15
SMAC	∞	∞	∞	∞	∞ *500*	0/15
RANDOMSEARCH	**4.2e5**(5e5)	∞	∞	∞	∞ *5e6*	0/15

#FEs/D	0.5	1.2	3	10	50	#succ
f20	*6.3e+3*:5.1	*4.0e+3*:8.4	*4.0e+1*:15	*2.5e+0*:69	*1.0e+0*:851	15/15
LHD-Default	1.7(2)	1.4(1.0)	1.9(0.4)	4.1(4)	∞ *250*	0/15
LHD-2xDefault	2.8(2)	2.2(1)	2.2(1)	7.6(6)	∞ *250*	0/15
RAND-2xDefault	2.0(2)	2.1(1)	2.6(1)	**3.3**(1)	∞ *250*	0/15
LHD-10xDefault	1.8(1)	1.4(1)	6.3(2)	11(9)	∞ *250*	0/15
SMAC	**0.57**(0.2)	**0.44**(0.3)	**0.73**(0.3)$^{\star 2}$	4.3(4)	∞ *500*	0/15
RANDOMSEARCH	1.6(1)	3.5(5)	24(24)	251(91)	**9234**(7768)	8/15

#FEs/D	0.5	1.2	3	10	50	#succ
f21	*4.0e+1*:3.9	*2.5e+1*:11	*1.6e+1*:31	*6.3e+0*:73	*1.6e+0*:347	5/5
LHD-Default	**1.3**(0.6)	1.3(1)	**0.73**(0.5)	1.2(1)	1.5(2)	7/15
LHD-2xDefault	1.7(1)	2.1(1)	1.3(0.7)	1.5(1)	1.7(1)	5/15
RAND-2xDefault	1.4(1)	1.1(0.8)	1.3(0.6)	1.4(0.8)	3.3(3)	3/15
LHD-10xDefault	1.5(1)	1.7(2)	1.5(2)	2.2(1)	2.0(3)	5/15
SMAC	1.8(1)	1.8(2)	1.0(0.8)	**1.0**(0.8)	**1.0**(1)	11/15
RANDOMSEARCH	1.4(2)	**1**(2)	1.3(2)	7.7(6)	15(17)	15/15

#FEs/D	0.5	1.2	3	10	50	#succ
f22	*6.3e+1*:3.6	*4.0e+1*:15	*2.5e+1*:32	*1.0e+1*:71	*1.6e+0*:341	5/5
LHD-Default	**1.6**(1)	**1.1**(0.8)	1.1(0.4)	1.1(0.7)	2.3(3)	5/15
LHD-2xDefault	3.3(4)	1.8(2)	1.3(0.6)	1.4(1)	1.8(2)	5/15
RAND-2xDefault	1.8(2)	1.5(1)	**0.98**(0.6)	**0.79**(0.4)	2.4(2)	4/15
LHD-10xDefault	2.2(2)	2.0(2)	2.3(2)	1.9(0.5)	2.5(2)	4/15
SMAC	1.8(2)	1.5(0.8)	1.0(0.9)	0.90(0.8)	**1.0**(1)	11/15
RANDOMSEARCH	2.4(0.8)	3.0(3)	2.7(1)	7.6(11)	43(57)	15/15

#FEs/D	0.5	1.2	3	10	50	#succ
f23	*1.0e+1*:3.0	*6.3e+0*:9.0	*4.0e+0*:33	*2.5e+0*:84	*1.0e+0*:518	15/15
LHD-Default	1.8(2)	**1.7**(1)	2.4(3)	3.9(3)	∞ *250*	0/15
LHD-2xDefault	3.1(3)	3.8(5)	3.8(5)	10(26)	∞ *250*	0/15
RAND-2xDefault	2.5(3)	2.7(3)	2.1(2)	4.6(11)	**7.1**(5)	1/15
LHD-10xDefault	3.9(6)	3.4(2)	3.3(3)	4.3(3)	**6.8**(10)	1/15
SMAC	**1.6**(1)	2.9(2)	2.6(4)	**3.3**(4)	∞ *500*	0/15
RANDOMSEARCH	2.3(1)	2.5(3)	**1.6**(2)	5.1(6)	49(45)	15/15

#FEs/D	0.5	1.2	3	10	50	#succ
f24	*6.3e+1*:15	*4.0e+1*:37	*4.0e+1*:37	*2.5e+1*:118	*1.6e+1*:692	15/15
LHD-Default	1.9(0.7)	3.1(6)	3.1(5)	10(13)	**5.3**(8)	1/15
LHD-2xDefault	1.9(2)	3.4(4)	3.4(2)	∞	∞ *250*	0/15
RAND-2xDefault	1.7(3)	7.3(17)	7.3(13)	15(12)	∞ *250*	0/15
LHD-10xDefault	2.7(3)	6.5(3)	6.5(7)	**4.8**(7)	∞ *250*	0/15
SMAC	**0.51**(0.7)	**2.6**(5)	**2.6**(4)	7.4(12)	∞ *500*	0/15
RANDOMSEARCH	3.6(4)	19(23)	19(10)	119(138)	418(416)	15/15

Table 3: Expected running time (ERT in number of function evaluations) divided by the respective best ERT measured during BBOB-2009 in dimension 5. The ERT and in braces, as dispersion measure, the half difference between 90 and 10%-tile of bootstrapped run lengths appear for each algorithm and run-length based target, the corresponding best ERT (preceded by the target Δf-value in *italics*) in the first row. #succ is the number of trials that reached the target value of the last column. The median number of conducted function evaluations is additionally given in *italics*, if the target in the last column was never reached. Entries, succeeded by a star, are statistically significantly better (according to the rank-sum test) when compared to all other algorithms of the table, with $p = 0.05$ or $p = 10^{-k}$ when the number k following the star is larger than 1, with Bonferroni correction by the number of instances.

#FEs/D	0.5	1.2	3	10	50	#succ
f1	*6.3e+1*:24	*4.0e+1*:42	*1.0e-8*:43	*1.0e-8*:43	*1.0e-8*:43	15/15
LHD-Default	2.5(0.5)	2.0(0.2)	∞	∞	∞ *1000*	0/15
LHD-2xDefault	3.9(0.2)	2.5(0.3)	∞	∞	∞ *1000*	0/15
RAND-2xDefault	4.0(0.6)	2.8(0.2)	∞	∞	∞ *1000*	0/15
LHD-10xDefault	17(0.1)	10(0.1)	∞	∞	∞ *1000*	0/15
SMAC	**0.80**(0.3)^{*4}	**0.67**(0.2)^{*4}	∞	∞	∞ *2000*	0/15
RANDOMSEARCH	892(1135)	3.7e4(4e4)	∞	∞	∞ *2e7*	0/15

#FEs/D	0.5	1.2	3	10	50	#succ
f2	*4.0e+6*:29	*2.5e+6*:42	*1.0e+5*:65	*1.0e+4*:207	*1.0e-8*:412	15/15
LHD-Default	0.79(0.8)	1.3(1.0)	9.3(6)	71(113)	∞ *1000*	0/15
LHD-2xDefault	1.1(0.1)	1.9(0.8)	8.5(4)	72(34)	∞ *1000*	0/15
RAND-2xDefault	1.3(2)	1.6(1)	**7.8**(4)	**70**(63)	∞ *1000*	0/15
LHD-10xDefault	1.6(2)	4.1(2)	30(31)	∞	∞ *1000*	0/15
SMAC	**0.54**(0.5)	**0.70**(0.5)	23(26)	143(252)	∞ *2000*	0/15
RANDOMSEARCH	2.8(4)	4.4(6)	1.0e6(9e5)	∞	∞ *2e7*	0/15

#FEs/D	0.5	1.2	3	10	50	#succ
f3	*6.3e+2*:33	*4.0e+2*:44	*1.6e+2*:109	*1.0e+2*:255	*2.5e+1*:3277	15/15
LHD-Default	1.9(0.8)	2.5(1)	8.0(12)	10(11)	∞ *1000*	0/15
LHD-2xDefault	2.5(0.8)	3.0(0.3)	8.6(5)	18(15)	∞ *1000*	0/15
RAND-2xDefault	2.7(0.2)	2.8(0.4)	**5.7**(5)	10(11)	∞ *1000*	0/15
LHD-10xDefault	9.1(5)	10(0.3)	8.7(6)	**9.0**(8)	∞ *1000*	0/15
SMAC	**0.49**(0.4)$^{\star2}_{\downarrow2}$	**2.1**(4)	124(143)	114(100)	∞ *2000*	0/15
RANDOMSEARCH	9.1(6)	1397(1258)	∞	∞	∞ *2e7*	0/15

#FEs/D	0.5	1.2	3	10	50	#succ
f4	*6.3e+2*:22	*4.0e+2*:91	*2.5e+2*:250	*1.6e+2*:332	*6.3e+1*:1927	15/15
LHD-Default	7.6(2)	**5.1**(4)	**10**(12)	∞	∞ *1000*	0/15
LHD-2xDefault	9.1(3)	6.8(8)	58(67)	∞	∞ *1000*	0/15
RAND-2xDefault	10(8)	8.4(9)	13(12)	∞	∞ *1000*	0/15
LHD-10xDefault	22(8)	18(7)	∞	∞	∞ *1000*	0/15
SMAC	**6.9**(8)	102(159)	∞	∞	∞ *2000*	0/15
RANDOMSEARCH	254(327)	3.2e4(5e4)	∞	∞	∞ *2e7*	0/15

#FEs/D	0.5	1.2	3	10	50	#succ
f5	*2.5e+2*:19	*1.6e+2*:34	*1.0e-8*:41	*1.0e-8*:41	*1.0e-8*:41	15/15
LHD-Default	1.8(0.9)	1.3(0.1)	2.4(1)	2.4(0.1)	2.4(0.3)	15/15
LHD-2xDefault	3.7(0.0)	2.5(0.1)	3.0(0.2)	3.0(0.1)	3.0(0.2)	15/15
RAND-2xDefault	3.4(2)	2.6(0.1)	3.4(2)	3.4(3)	3.4(0.2)	15/15
LHD-10xDefault	8.1(7)	12(0.0)	11(0.1)	11(0.1)	11(0.3)	15/15
SMAC	**0.46**(0.1)$^{\star}_{\downarrow2}$	**0.33**(0.1)^{*4}	**0.66**(0.2)^{*4}	**0.66**(0.2)^{*4}	**0.66**(0.3)^{*4}	15/15
RANDOMSEARCH	8.0(12)	1833(2538)	∞	∞	∞ *2e7*	0/15

#FEs/D	0.5	1.2	3	10	50	#succ
f6	*2.5e+5*:16	*6.3e+4*:43	*1.6e+4*:62	*1.6e+2*:353	*1.6e+1*:1078	15/15
LHD-Default	2.4(0.4)	1.4(0.5)	**1.4**(0.4)	9.5(14)	∞ *1000*	0/15
LHD-2xDefault	4.6(3)	2.3(0.4)	1.9(0.4)	6.8(8)	∞ *1000*	0/15
RAND-2xDefault	4.8(1)	2.3(0.6)	1.8(0.5)	5.7(8)	∞ *1000*	0/15
LHD-10xDefault	17(12)	10(0.1)	7.1(0.3)	4.3(3)	∞ *1000*	0/15
SMAC	**1.6**(1)	**1.2**(0.9)	1.6(0.8)	**2.8**(3)	∞ *2000*	0/15
RANDOMSEARCH	263(299)	3.2e4(9e4)	1.7e5(2e5)	∞	∞ *2e7*	0/15

#FEs/D	0.5	1.2	3	10	50	#succ
f7	*1.0e+3*:11	*4.0e+2*:39	*2.5e+2*:74	*6.3e+1*:319	*1.0e+1*:1351	15/15
LHD-Default	1.2(0.6)	2.2(3)	2.0(2)	5.6(6)	∞ *1000*	0/15
LHD-2xDefault	1.5(2)	2.6(0.4)	1.9(0.5)	8.1(6)	∞ *1000*	0/15
RAND-2xDefault	2.2(1)	2.8(1)	2.1(0.5)	6.4(5)	∞ *1000*	0/15
LHD-10xDefault	2.4(2)	10(2)	5.7(1)	8.1(6)	∞ *1000*	0/15
SMAC	**0.58**(0.7)	**0.61**(0.5)^{*3}	**0.49**(0.4)^{*4}	**0.39**(0.3)^{*3}	**0.57**(0.3)^{*4}	15/15
RANDOMSEARCH	3.2(7)	76(53)	1020(1458)	9.2e5(1e6)	∞ *2e7*	0/15

#FEs/D	0.5	1.2	3	10	50	#succ
f8	*4.0e+4*:19	*2.5e+4*:35	*4.0e+3*:67	*2.5e+2*:231	*1.6e+1*:1470	15/15
LHD-Default	4.5(1)	2.9(0.8)	3.3(2)	4.0(3)	∞ *1000*	0/15
LHD-2xDefault	5.5(0.7)	3.2(0.6)	3.1(0.8)	2.8(0.4)	∞ *1000*	0/15
RAND-2xDefault	5.4(0.5)	3.2(0.8)	2.8(0.6)	**2.5**(0.5)	∞ *1000*	0/15
LHD-10xDefault	23(0.2)	12(0.1)	6.7(0.1)	3.8(3)	∞ *1000*	0/15
SMAC	**1.4**(2)^{*3}	**1.5**(1)^{*2}	**2.5**(0.8)	4.1(2)	∞ *2000*	0/15
RANDOMSEARCH	809(670)	4532(4639)	4.2e6(7e6)	∞	∞ *2e7*	0/15

#FEs/D	0.5	1.2	3	10	50	#succ
f9	*1.0e+2*:357	*6.3e+1*:560	*4.0e+1*:684	*2.5e+1*:756	*1.0e+1*:1716	15/15
LHD-Default	7.9(4)	∞	∞	∞	∞ *1000*	0/15
LHD-2xDefault	5.3(3)	13(10)	22(28)	∞	∞ *1000*	0/15
RAND-2xDefault	5.3(4)	**13**(12)	**22**(32)	∞	∞ *1000*	0/15
LHD-10xDefault	8.0(6)	27(26)	∞	∞	∞ *1000*	0/15
SMAC	**5.0**(1)	26(29)	∞	∞	∞ *2000*	0/15
RANDOMSEARCH	∞	∞	∞	∞	∞ *2e7*	0/15

#FEs/D	0.5	1.2	3	10	50	#succ
f10	*1.6e+6*:15	*1.0e+6*:27	*4.0e+5*:70	*6.3e+4*:231	*4.0e+3*:1015	15/15
LHD-Default	4.4(3)	**3.1**(1)	3.4(4)	9.1(10)	∞ *1000*	0/15
LHD-2xDefault	6.7(4)	4.8(4)	**3.3**(2)	**5.8**(8)	∞ *1000*	0/15
RAND-2xDefault	6.4(3)	4.6(1)	3.6(2)	6.9(7)	∞ *1000*	0/15
LHD-10xDefault	15(14)	12(7)	7.5(1)	21(21)	∞ *1000*	0/15
SMAC	**3.7**(2)	5.5(4)	6.2(6)	18(13)	∞ *2000*	0/15
RANDOMSEARCH	28(29)	77(171)	894(1278)	∞	∞ *2e7*	0/15

#FEs/D	0.5	1.2	3	10	50	#succ
f11	*4.0e+4*:11	*2.5e+3*:27	*1.6e+2*:313	*1.0e+2*:481	*1.0e+1*:1002	15/15
LHD-Default	1.7(2)	2.5(2)	21(16)	∞	∞ *1000*	0/15
LHD-2xDefault	1.9(2)	2.6(3)	22(14)	∞	∞ *1000*	0/15
RAND-2xDefault	3.0(4)	2.5(2)	∞	∞	∞ *1000*	0/15
LHD-10xDefault	2.6(2)	4.0(6)	22(25)	∞	∞ *1000*	0/15
SMAC	**0.59**(0.4)	**0.68**(0.5)	**2.5**(2)*	**7.3**(8)	∞ *2000*	0/15
RANDOMSEARCH	2.0(2)	3.2(2)	34(32)	929(827)	∞ *2e7*	0/15

#FEs/D	0.5	1.2	3	10	50	#succ
f12	*1.0e+8*:23	*6.3e+7*:39	*2.5e+7*:76	*4.0e+6*:209	*1.0e+1*:1042	15/15
LHD-Default	3.1(1)	**2.9**(0.9)	**3.0**(1)	3.6(0.6)	∞ *1000*	0/15
LHD-2xDefault	4.6(2)	4.1(0.9)	3.8(0.8)	**3.2**(2)	∞ *1000*	0/15
RAND-2xDefault	5.0(1)	4.0(1)	3.7(2)	3.8(2)	∞ *1000*	0/15
LHD-10xDefault	17(0.8)	12(0.8)	8.9(4)	17(17)	∞ *1000*	0/15
SMAC	**2.8**(3)	38(51)	∞	∞	∞ *2000*	0/15
RANDOMSEARCH	294(301)	2589(4144)	8.3e5(6e5)	∞	∞ *2e7*	0/15

#FEs/D	0.5	1.2	3	10	50	#succ
f13	*1.6e+3*:28	*1.0e+3*:64	*6.3e+2*:79	*4.0e+1*:211	*2.5e+0*:1724	15/15
LHD-Default	2.6(0.6)	1.8(0.3)	2.6(1)	8.0(4)	∞ *1000*	0/15
LHD-2xDefault	3.3(1)	2.2(0.2)	2.8(1)	9.4(13)	∞ *1000*	0/15
RAND-2xDefault	3.5(0.4)	2.2(0.5)	2.9(1)	7.1(4)	∞ *1000*	0/15
LHD-10xDefault	15(0.2)	6.9(0.2)	6.3(0.3)	8.5(9)	∞ *1000*	0/15
SMAC	**0.81**(0.3)^{*2}	**0.66**(0.1)^{*4}	**0.84**(0.1)^{*4}	**1.4**(0.3)^{*4}	∞ *2000*	0/15
RANDOMSEARCH	358(81)	1.5e5(2e5)	∞	∞	∞ *2e7*	0/15

#FEs/D	0.5	1.2	3	10	50	#succ
f14	*2.5e+1*:15	*1.6e+1*:42	*1.0e+1*:75	*1.6e+0*:219	*6.3e-4*:1106	15/15
LHD-Default	5.7(2)	**3.1**(1)	**2.9**(1)	9.5(9)	∞ *1000*	0/15
LHD-2xDefault	8.1(2)	3.7(1)	3.1(1)	**3.8**(3)	∞ *1000*	0/15
RAND-2xDefault	8.8(2)	4.1(1)	3.3(1)	7.3(5)	∞ *1000*	0/15
LHD-10xDefault	25(7)	11(0.5)	6.9(1)	7.7(12)	∞ *1000*	0/15
SMAC	**2.0**(0.8)^{*2}	3.3(4)	19(26)	∞	∞ *2000*	0/15
RANDOMSEARCH	155(127)	1839(1525)	4.5e4(5e4)	∞	∞ *2e7*	0/15

#FEs/D	0.5	1.2	3	10	50	#succ
f15	*6.3e+2*:15	*4.0e+2*:67	*2.5e+2*:292	*1.6e+2*:846	*1.0e+2*:1671	15/15
LHD-Default	3.5(1)	**1.7**(0.7)	1.1(0.5)	2.0(3)	8.8(12)	1/15
LHD-2xDefault	5.5(2)	2.0(0.3)	1.1(1)	1.5(2)	8.7(12)	1/15
RAND-2xDefault	5.1(3)	1.9(0.5)	**1.1**(0.4)	1.6(2)	8.8(8)	1/15
LHD-10xDefault	16(9)	6.4(0.1)	1.9(0.6)	**1.1**(0.8)	**1.2**(1)	7/15
SMAC	**1.1**(2)^{*2}	2.9(0.2)	2.8(4)	∞	∞ *2000*	0/15
RANDOMSEARCH	35(61)	1316(1150)	1.6e5(4e5)	∞	∞ *2e7*	0/15

#FEs/D	0.5	1.2	3	10	50	#succ
f16	*4.0e+1*:26	*2.5e+1*:127	*1.6e+1*:540	*1.6e+1*:540	*1.0e+1*:1384	15/15
LHD-Default	3.4(2)	4.9(2)	3.3(3)	3.3(2)	11(7)	1/15
LHD-2xDefault	3.2(2)	3.6(3)	2.6(2)	2.6(2)	5.1(6)	2/15
RAND-2xDefault	3.8(2)	6.4(6)	5.0(4)	5.0(7)	11(16)	1/15
LHD-10xDefault	6.5(4)	5.4(5)	4.8(5)	4.8(8)	3.4(5)	3/15
SMAC	**2.4**(3)	**1.5**(0.7)	**0.78**(0.6)	**0.78**(0.3)	**0.76**(0.8)	14/15
RANDOMSEARCH	3.9(5)	33(40)	830(1010)	830(312)	6.5e4(3e4)	3/15

#FEs/D	0.5	1.2	3	10	50	#succ
f17	*1.6e+1*:11	*1.0e+1*:63	*6.3e+0*:305	*4.0e+0*:468	*1.0e+0*:1030	15/15
LHD-Default	3.3(3)	2.2(2)	**1.3**(0.7)	32(40)	∞ *1000*	0/15
LHD-2xDefault	3.3(4)	2.6(0.9)	1.4(1)	10(9)	∞ *1000*	0/15
RAND-2xDefault	5.0(4)	2.7(2)	2.1(3)	31(21)	∞ *1000*	0/15
LHD-10xDefault	12(6)	7.3(2)	2.2(1.0)	**5.7**(5)	∞ *1000*	0/15
SMAC	**0.52**(0.5)^{*2}	**0.92**(1.0)	15(36)	61(103)	∞ *2000*	0/15
RANDOMSEARCH	16(32)	120(73)	4216(4815)	∞	∞ *2e7*	0/15

#FEs/D	0.5	1.2	3	10	50	#succ
f18	*4.0e+1*:116	*2.5e+1*:252	*1.6e+1*:430	*1.0e+1*:621	*4.0e+0*:1090	15/15
LHD-Default	1.0(0.9)	2.4(2)	8.3(5)	∞	∞ *1000*	0/15
LHD-2xDefault	1.00(0.3)	1.6(2)	11(27)	∞	∞ *1000*	0/15
RAND-2xDefault	0.99(0.3)	**1.6**(0.8)	16(15)	∞	∞ *1000*	0/15
LHD-10xDefault	3.5(0.5)	2.7(1)	**5.2**(4)	∞	∞ *1000*	0/15
SMAC	**0.31**(0.2)$^{\star2}_{\downarrow2}$	8.3(9)	20(21)	**22**(50)	∞ *2000*	0/15
RANDOMSEARCH	23(13)	1995(1123)	∞	∞	∞ *2e7*	0/15

#FEs/D	0.5	1.2	3	10	50	#succ
f19	*1.6e-1*:2.5e5	*1.0e-1*:3.4e5	*6.3e-2*:3.4e5	*4.0e-2*:3.4e5	*2.5e-2*:3.4e5	3/15
LHD-Default	∞	∞	∞	∞	∞ *1000*	0/15
LHD-2xDefault	∞	∞	∞	∞	∞ *1000*	0/15
RAND-2xDefault	∞	∞	∞	∞	∞ *1000*	0/15
LHD-10xDefault	∞	∞	∞	∞	∞ *1000*	0/15
SMAC	∞	∞	∞	∞	∞ *2000*	0/15
RANDOMSEARCH	∞	∞	∞	∞	∞ *2e7*	0/15

#FEs/D	0.5	1.2	3	10	50	#succ
f20	*1.6e+4*:38	*1.0e+4*:42	*2.5e+2*:62	*2.5e+0*:250	*1.6e+0*:2536	15/15
LHD-Default	2.2(0.7)	2.4(1)	4.6(3)	4.1(6)	∞ *1000*	0/15
LHD-2xDefault	3.3(0.8)	3.3(0.6)	4.4(1)	**2.8**(1)	**5.7**(5)	1/15
RAND-2xDefault	3.1(0.7)	3.1(0.7)	4.8(2)	4.9(6)	∞ *1000*	0/15
LHD-10xDefault	11(0.5)	10(0.7)	8.7(0.6)	7.9(6)	∞ *1000*	0/15
SMAC	**0.25**(0.1)$^{\star4}_{\downarrow}$	**0.46**(0.2)$^{\star4}_{\downarrow4}$	**0.90**(0.2)^{*4}	∞	∞ *2000*	0/15
RANDOMSEARCH	216(190)	1380(524)	∞	∞	∞ *2e7*	0/15

#FEs/D	0.5	1.2	3	10	50	#succ
f21	*6.3e+1*:36	*4.0e+1*:77	*4.0e+1*:77	*1.6e+1*:456	*4.0e+0*:1094	15/15
LHD-Default	**2.7**(0.8)	**1.9**(0.4)	**1.9**(0.4)	0.56(0.2)	0.97(0.4)	9/15
LHD-2xDefault	3.6(0.5)	2.6(2)	2.6(1)	0.88(0.4)	1.2(1.0)	9/15
RAND-2xDefault	3.4(0.8)	1.9(0.3)	1.9(0.2)	**0.46**(0.1)	**0.57**(0.8)	12/15
LHD-10xDefault	13(0.6)	6.6(0.7)	6.6(0.4)	2.0(2)	2.4(5)	5/15
SMAC	7.5(3)	4.2(20)	4.2(8)	2.7(3)	5.2(9)	4/15
RANDOMSEARCH	943(222)	2.0e4(3e4)	2.0e4(1e4)	∞	∞ *2e7*	0/15

#FEs/D	0.5	1.2	3	10	50	#succ
f22	*6.3e+1*:45	*4.0e+1*:68	*4.0e+1*:68	*1.6e+1*:231	*6.3e+0*:1219	15/15
LHD-Default	**2.2**(0.5)	**2.1**(0.4)	**2.1**(2)	**1.5**(2)	**0.58**(0.3)	11/15
LHD-2xDefault	3.4(0.7)	3.3(3)	3.3(3)	2.0(2)	1.2(1)	8/15
RAND-2xDefault	3.3(0.3)	3.2(2)	3.2(4)	1.9(3)	1.1(1)	8/15
LHD-10xDefault	10(5)	7.8(2)	7.8(3)	5.4(4)	2.1(3)	5/15
SMAC	10(24)	7.3(3)	7.3(2)	4.2(7)	2.0(4)	7/15
RANDOMSEARCH	1986(1078)	9.2e4(1e5)	9.2e4(1e5)	1.3e6(2e6)	∞ *2e7*	0/15

#FEs/D	0.5	1.2	3	10	50	#succ
f23	*6.3e+0*:29	*4.0e+0*:118	*2.5e+0*:306	*2.5e+0*:306	*1.0e+0*:1614	15/15
LHD-Default	2.0(1)	10(11)	∞	∞	∞ *1000*	0/15
LHD-2xDefault	1.6(2)	5.6(4)	**23**(30)	**23**(40)	∞ *1000*	0/15
RAND-2xDefault	2.1(3)	6.2(3)	∞	∞	∞ *1000*	0/15
LHD-10xDefault	2.3(4)	8.2(9)	48(91)	48(78)	∞ *1000*	0/15
SMAC	**1.6**(2)	**5.0**(2)	46(112)	46(58)	∞ *2000*	0/15
RANDOMSEARCH	2.3(2)	7.0(7)	92(67)	92(95)	**5.5e4**(7e4)	3/15

#FEs/D	0.5	1.2	3	10	50	#succ
f24	*2.5e+2*:208	*1.6e+2*:918	*1.0e+2*:6628	*6.3e+1*:9885	*4.0e+1*:31629	15/15
LHD-Default	15(11)	∞	∞	∞	∞ *1000*	0/15
LHD-2xDefault	7.5(18)	7.8(15)	∞	∞	∞ *1000*	0/15
RAND-2xDefault	9.3(10)	∞	∞	∞	∞ *1000*	0/15
LHD-10xDefault	3.0(0.2)	**1.6**(1)	**0.74**(0.6)	∞	∞ *1000*	0/15
SMAC	**0.65**(0.8)^{*2}	10(9)	∞	∞	∞ *2000*	0/15
RANDOMSEARCH	1.7e5(4e5)	∞	∞	∞	∞ *2e7*	0/15

Table 4: Expected running time (ERT in number of function evaluations) divided by the respective best ERT measured during BBOB-2009 in dimension 20. The ERT and in braces, as dispersion measure, the half difference between 90 and 10%-tile of bootstrapped run lengths appear for each algorithm and run-length based target, the corresponding best ERT (preceded by the target Δf-value in *italics*) in the first row. #succ is the number of trials that reached the target value of the last column. The median number of conducted function evaluations is additionally given in *italics*, if the target in the last column was never reached. Entries, succeeded by a star, are statistically significantly better (according to the rank-sum test) when compared to all other algorithms of the table, with $p = 0.05$ or $p = 10^{-k}$ when the number k following the star is larger than 1, with Bonferroni correction by the number of instances.