Niching the CMA-ES via Nearest-Better Clustering

Mike Preuss
Computational Intelligence Group, Chair of Algorithm Engineering
Technische Universität Dortmund, Germany
mike.preuss@tu-dortmund.de

ABSTRACT

We investigate how a niching based evolutionary algorithm fares on the BBOB function test set, knowing that most problems are not very well suited to this algorithm class. However, as the CMA-ES is included as basic local search algorithm, the niching approach still performs fairly well, with some potential to improve. Basin identification is done via the heuristic nearest-best clustering scheme.

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

Keywords

Benchmarking, Black-box optimization, Niching

1. INTRODUCTION

Without any doubt, the CMA-ES [6] (we employ the variant described in [2] in the following) is an optimization method capable of dealing with many difficulties that can occur in an optimization problem. However, by its very nature, it acts rather *locally* then *globally* as a single run tends to concentrate rapidly on a very small search space portion. This is no problem of course if the chosen portion is the 'right' one, i.e. contains the global optimum. Thus, for convex problems, the CMA-ES is most likely the best method one can get from the field of *evolutionary computation* (EC). Nevertheless, it seems that many real-world problems are multimodal, and this leads to problems for all locally oriented optimization algorithms.

In the CMA-ES, this is overcome by a restart mechanism that detects stagnation and sets up a new population in a different (usually randomly determined) search space area. One could argue that this already is a kind of niching. In principle, the meaning of niching is nothing else than that search is parallelized, and this can be done either in (search) space or time, or both. What makes a niching method a niching method is probably that the multiple algorithm runs (further on for reasons of simplification termed *local searches* are coordinated. The basic idea is that by avoiding multiple local searches in the same area, function evaluations are saved. In our case, this may be seen as a *hybridization* of the CMA-ES with a high-level control strategy, and we employ a simple heuristic to identify different search space portions where to run local search.

However, niching is NOT a universal cure: It is an open question under which conditions coordinated parallelization makes sense, and it seems that for very highly multimodal cases, the effort invested into coordination does not pay off as it is simply impossible to identify enough basins of attraction to obtain a clear advantage over uncoordinated restarts [7]. Therefore, one can hardly expect any niching method to improve over a good restart method on functions 1-15, 19 and 23 of this test set as they are out of scope for these methods (either unimodal or too highly multimodal). The Lunacek bi-Rastrigin function may be a special case as it is highly multimodal but also has a higher order structure (in the function outlined by the end points of local searches) as all funnel problems (see [1] for more examples). Additionally, setting up a good optimization algorithm for a (multimodal) real-world problem usually requires adjusting the algorithm which is rather discouraged in this comparison, and for good reason, because it complicates assessment tremendously. Thus, for now, we concentrate on the remaining set of functions (16-18, 20-22, and 24) and try to obtain hints concerning the conditions under which the standard CMA-ES can be improved.

As a large number of different niching methods have been suggested in the recent years (not to count the classic methods *fitness sharing* and *crowding* and their various derivatives), we abstain from refering to all of them in this context but rather give two links: The method(s) of Shir and Bäck ([9] and follow-ups) are similar to ours in that they also employ a CMA-ES as local search base algorithm. However, they make specific prerequisites concerning the shape of the identified basins which is assumed to be spherical or elipsoid. The method of Stoean and Preuss [10] removes this requirement via the criterion of Ursem [11], but the obtained freedom makes the resulting algorithm a bit harder to control. The approach suggested here has inherited from both

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'10, July 7–11, 2010, Portland, Oregon, USA.
Copyright 2010 ACM 978-1-4503-0073-5/10/07 ...$10.00.

by building on the CMA-ES as local search procedure and employing a radius-free *basin identification* method. which is going to be explained in the following section.

2. ALGORITHM PRESENTATION

Apart from the embedded CMA-ES, our algorithm has two specific constituents, a basin identification system which shall actually detect the niches to pursue, and a high level strategy which controls how to use this information. We start with the basin identification that can be successfully run on any search point set, even on a completely randomized sample distribution.

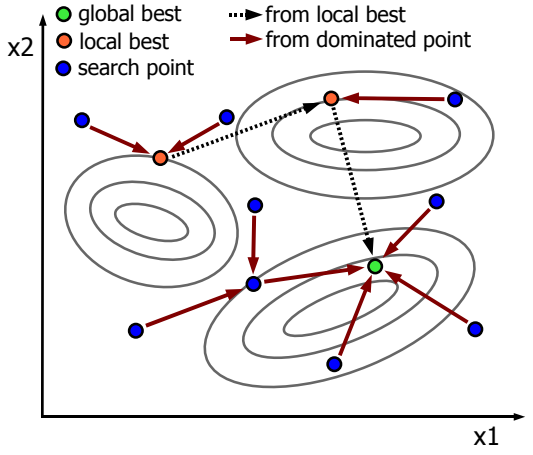

Figure 1: Nearest-better clustering example; basins of attraction are indicated by contour lines. Search points each connect to their nearest better neighbor. The illustration shows a random sample on a 3-basin 2 dimensional landscape. Search points each connect to the nearest individual which is better; clustering is done via cutting the longest lines.

2.1 Nearest-Better Clustering

The nearest-better clustering algorithm (NBC) has first been presented in [8] and is outlined in high-level pseudocode in Algorithm 1. It rests upon the assumption that the best yet found search points in different basins of attraction—approximations of different local optima—are much further away from each other than all search points are from their nearest better neighbor on average. Thus, it first creates a spanning tree out of the current population (see Figure 1, left hand side) by connecting each search point to the nearest one that is genuinely better. The longest edges are then cut, each time splitting a connected component into two. We identify these by means of a heuristic rule that uses the new parameter ϕ. Experimentation with varying ϕ led to the conjecture that it is much less sensitive to changes than a niche radius parameter would be and that it may be set to 2 as a very good default value.

There is one notable exception where executing lines 2 and 3 does not lead to a spanning tree, but a spanning forest. This is the case when there are multiple best points, so that the number of edges created within the **forall** loop decreases below $n-1$ for n search points (by definition, there are no better points best points could be connected to). However,

Algorithm 1: Nearest-better clustering (NBC)

```
1 compute all search points mutual distances;
2 create an empty graph with num(search points) nodes;
  // make spanning tree:
3 forall the search points do
4 |   find nearest search point that is better; create edge
  |   to it;
  // cut spanning tree into clusters:
5 delete edges of length > φ · mean(lengths of all edges) ;
  // find clusters:
6 find connected components;
```

the effect on NBC shall be negligible in most cases. In line 5 we consider all edges, be they connected or not, and having less edges only means that we start cutting from a number of connected components that is greater than 1.

2.2 Control Strategy

The NBC is very dynamic in the number of potential basins it returns. This must be considered by the control strategy and is in stark contrast to many other niching techniques which identify a preset number of bases. However, practical considerations lead to setting a limit to the maximum number of niches pursued at the same time (and thus parallel CMA-ES runs). Of the identified niches, we thus select the best ones up to $nich_{max}$ and only consider clusters as separated if the have a minimum distance of at least $dist_{min}$.

The algorithm starts by randomly distributing *startpop* individuals in the search space and evaluating them. Then, NBC is run and the individuals are gathered in their respective populations, employing the grouping produced by the clustering, capping to $nich_{max}$ populations. Then, we run each population for one step (generation) as a usual CMA-ES (with standard operators and matrix/step-size updating local to this population; each population carries its own covariance matrix and step-size) and copy all produced offspring individuals into an additional set on which the clustering is run again. As well as the non-selected offspring is used for the clustering, it is not copied to the populations for the next generation as this happens only with the selected individuals.

We then establish the new set of populations from the clustering. An attempt to preserve existing populations is made by letting the largest group of individuals that is sorted into one specific cluster stay in their population (and thus also preserve the covariance matrix and step-size). If the largest groups used for building a new population are equally sized but stem from different parent populations, the meta-parameters are copied from the one that achieved the best fitness value, and if these are also equal, from the first one detected by the clustering.

Where needed, new populations are set up when no matching one does exist. This often happens with 'outliers' if very large mutations appear. In this case, already learned step-sizes and covariance matrices are copied from the parent population, that one that produced the individual(s) of the new population. In case of multiple ancestors, the same rule as above is applied.

We continue this loop of clustering, population assignment and one-generation local population runs and stop

only when the algorithm needs to terminate due to external reasons (max evaluations or global optimum found). For each population, the usual CMA-ES stop conditions are also checked after each one-generation step and if they apply, the population simply dies out.

3. EXPERIMENTAL PROCEDURE

We run the full BBOB 2010 noiseless test set with a maximum number of $3x10^5$ function evaluations per run. If the global optimum is obtained with the required accuracy as detected via the `ftarget()` method, a run is immediately stopped.

The parameters are fixed for all runs and set to default values (or default rules) for the CMA-ES, except the population size which is fixed at $\mu = 5, \lambda = 10$.

The niching parameters are set to these values:

- $startpop = 100$
- $nich_{max} = 20$
- $dist_{min} = \sqrt[D]{10^{-6}}$ where D is number of dimensions

As no parameters have been specifically adapted to any problem or dimension, the crafting effort is 0.

4. RESULTS

Results from experiments according to [4] on the benchmark functions given in [3, 5] are presented in Figures 2, 3 and 4 and in Tables 1 and 2.

We make the following observations: Generally, our method enables rapid progress at the start and gets slower during the run. It copes especially well with functions 17, 18, 21, 22 and 23 and is unexpectedly weak on function 20. For the unimodal and highly multimodal functions, it performs a bit worse than the BI-POP CMA-ES (comparison data not shown here). In some cases (e.g. function 5 in 20+ dimensions) it is much worse than the CMA-ES. In no case, the respective best method from the BBOB 2009 entries is beaten for high accuracies.

5. CPU TIMING EXPERIMENT

For the timing experiment the NBC-CMA-ES was run on f8 and restarted until at least 30 seconds had passed. These experiments have been conducted with an Intel dual core T5550 processor with 1.83 GHz under Windows Vista using Java 1.6.0_17. The results were 1.7; 1.7; 1.7; 1.4; 0.8; $1.3x10^{-07}$ per function evaluation in dimension 2; 3; 5; 10; 20; 40 and 80, respectively. Up to 80-D a dependency of CPU time on the search space dimensionality is hardly visible.

6. CONCLUSIONS

In general, our approach appears as a good compromise between the overall behavior of the CMA-ES and niching-based rapid progress up to low accuracies. We see some potential to improve the method especially in reviewing the ad-hoc values chosen for our parameters and dealing with cases where wrongly, many basins are identified during a run which seems to be an artifact of different populations differing too much in progress. We thus conclude that the control strategy should be improved to better handle incoming clustering information in mid-run.

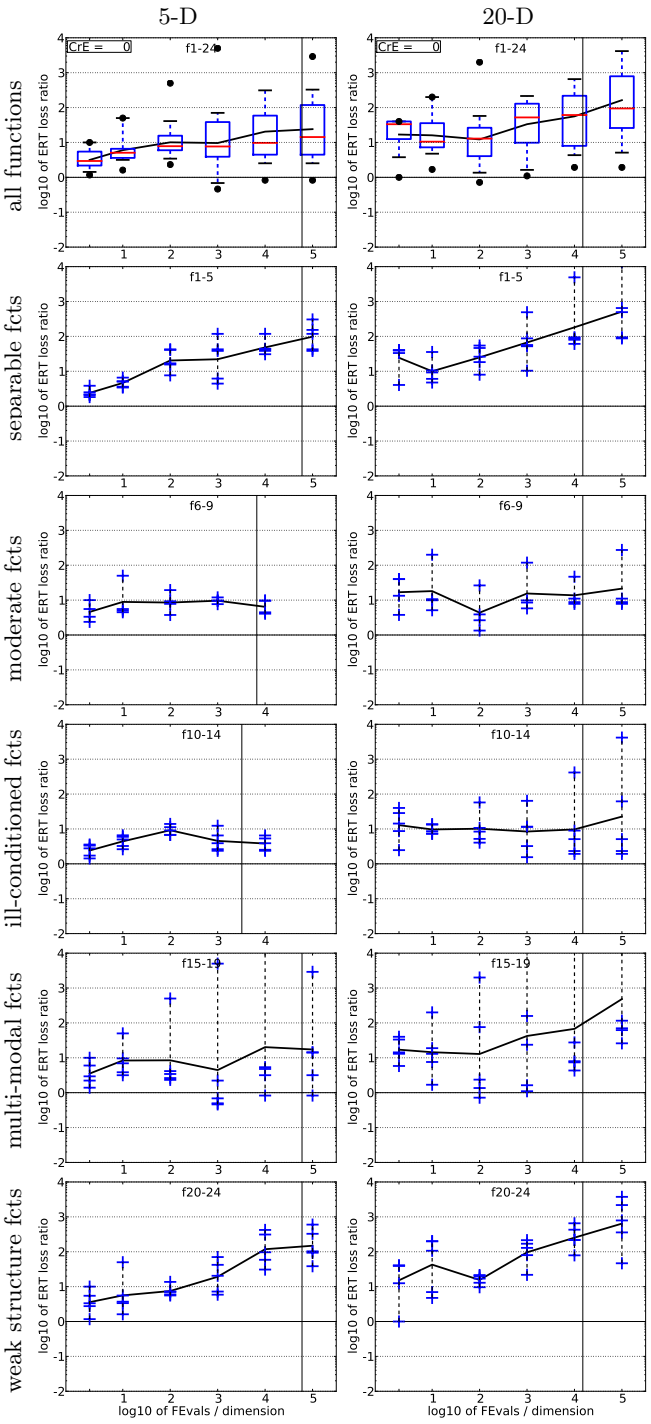

Figure 4: ERT loss ratio versus given budget FEvals. The target value f_t for ERT (see Figure 2) is the smallest (best) recorded function value such that ERT(f_t) $\leq$ FEvals for the presented algorithm. Shown is FEvals divided by the respective best ERT(f_t) from BBOB-2009 for functions f_1–f_{24} in 5-D and 20-D. Each ERT is multiplied by exp(CrE) correcting for the parameter crafting effort. Line: geometric mean. Box-Whisker error bar: 25-75%-ile with median (box), 10-90%-ile (caps), and minimum and maximum ERT loss ratio (points). The vertical line gives the maximal number of function evaluations in this function subset.

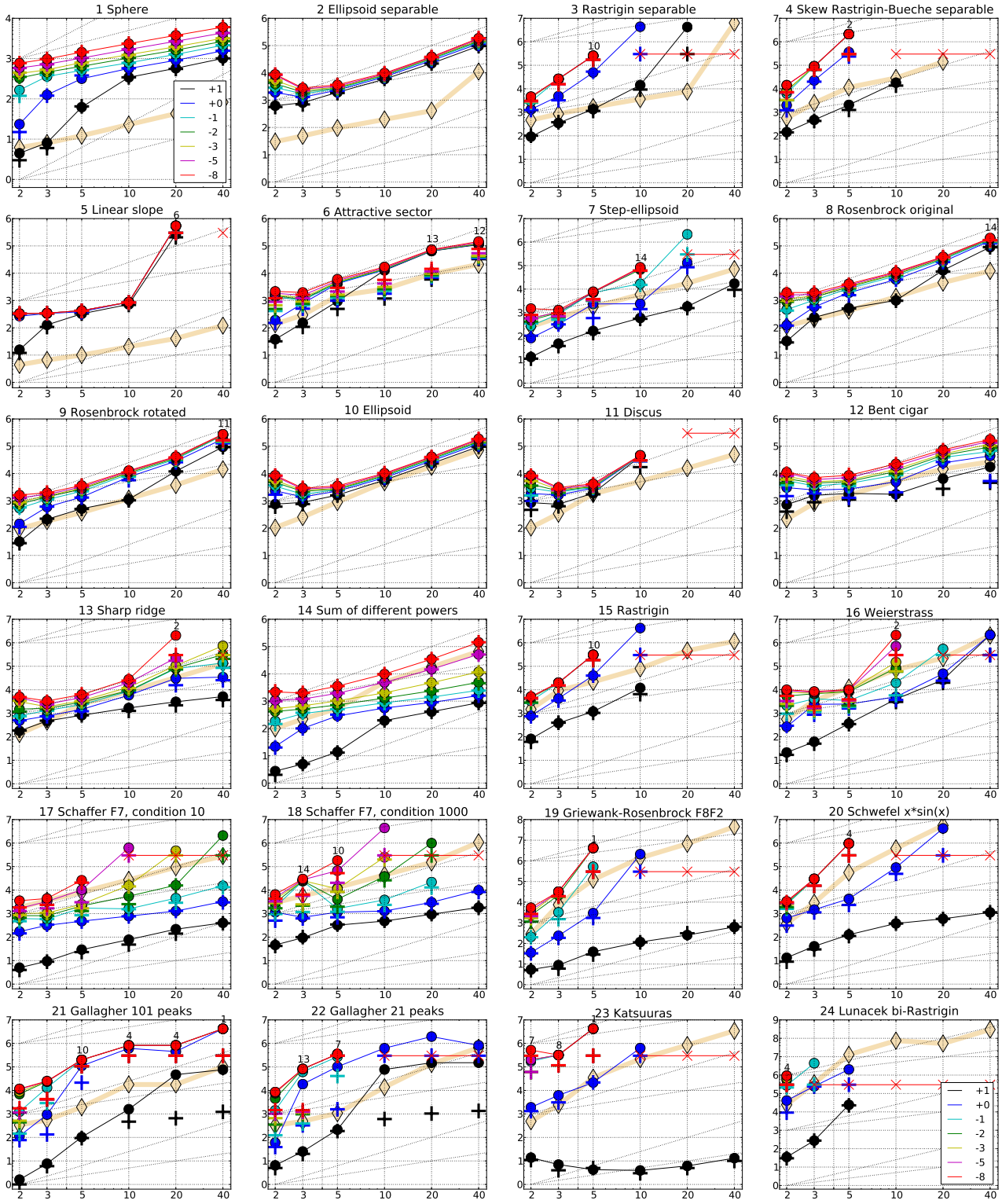

Figure 2: Expected Running Time (ERT, ●) to reach $f_{\mathrm{opt}} + \Delta f$ and median number of f-evaluations from successful trials (+), for $\Delta f = 10^{\{+1,0,-1,-2,-3,-5,-8\}}$ (the exponent is given in the legend of f_1 and f_{24}) versus dimension in log-log presentation. For each function and dimension, $\mathrm{ERT}(\Delta f)$ 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. The $\#\mathrm{FEs}(\Delta f)$ are the total number (sum) of f-evaluations while $f_{\mathrm{opt}} + \Delta f$ was not surpassed in the trial, from all (successful and unsuccessful) trials, and f_{opt} is the optimal function value. Crosses (×) indicate the total number of f-evaluations, $\#\mathrm{FEs}(-\infty)$, divided by the number of trials. Numbers above ERT-symbols indicate the number of successful trials. Y-axis annotations are decimal logarithms. The thick light line with diamonds shows the single best results from BBOB-2009 for $\Delta f = 10^{-8}$. Additional grid lines show linear and quadratic scaling.

Δf	#	ERT	10%	90%	RT_{succ}
f_1 in 5-D, N=15, mFE=1540					
10	15	6.2e1	7.0e0	1.2e2	6.2e1
1	15	3.2e2	1.1e2	4.3e2	3.2e2
1e−1	15	5.1e2	4.3e2	5.7e2	5.1e2
1e−3	15	8.0e2	7.4e2	8.6e2	8.0e2
1e−5	15	1.0e3	9.6e2	1.1e3	1.0e3
1e−8	15	1.4e3	1.4e3	1.5e3	1.4e3
f_1 in 20-D, N=15, mFE=4010					
10	15	5.7e2	5.0e2	6.8e2	5.7e2
1	15	9.0e2	8.0e2	9.7e2	9.0e2
1e−1	15	1.3e3	1.2e3	1.3e3	1.3e3
1e−3	15	2.0e3	1.9e3	2.0e3	2.0e3
1e−5	15	2.7e3	2.6e3	2.8e3	2.7e3
1e−8	15	3.8e3	3.6e3	3.9e3	3.8e3
f_2 in 5-D, N=15, mFE=4440					
10	15	1.9e3	1.4e3	2.8e3	1.9e3
1	15	2.3e3	1.7e3	3.1e3	2.3e3
1e−1	15	2.6e3	2.2e3	3.2e3	2.6e3
1e−3	15	3.0e3	2.5e3	3.5e3	3.0e3
1e−5	15	3.2e3	2.7e3	3.7e3	3.2e3
1e−8	15	3.6e3	3.2e3	4.2e3	3.6e3
f_2 in 20-D, N=15, mFE=44900					
10	15	2.2e4	1.9e4	2.6e4	2.2e4
1	15	2.8e4	2.5e4	3.1e4	2.8e4
1e−1	15	3.2e4	2.9e4	3.5e4	3.2e4
1e−3	15	3.5e4	3.3e4	3.8e4	3.5e4
1e−5	15	3.7e4	3.5e4	4.0e4	3.7e4
1e−8	15	3.9e4	3.6e4	4.2e4	3.9e4
f_3 in 5-D, N=15, mFE=300000					
10	15	1.4e3	8.2e2	2.1e3	1.4e3
1	15	4.9e4	4.6e3	1.0e5	4.9e4
1e−1	10	2.5e5	4.2e4	6.1e5	1.0e5
1e−3	10	2.5e5	4.2e4	6.4e5	1.0e5
1e−5	10	2.5e5	4.2e4	6.4e5	1.0e5
1e−8	10	2.5e5	4.3e4	6.1e5	1.0e5
f_3 in 20-D, N=15, mFE=300000					
10	1	4.3e6	3.5e5	9.7e6	5.3e4
1	0	*18e+0*	*14e+0*	*21e+0*	1.3e5
1e−1	.	.	.	.	.
1e−3	.	.	.	.	.
1e−5	.	.	.	.	.
1e−8	.	.	.	.	.
f_4 in 5-D, N=15, mFE=300005					
10	15	2.1e3	9.9e2	4.3e3	2.1e3
1	9	3.6e5	9.5e4	7.7e5	1.6e5
1e−1	2	2.1e6	1.7e5	5.0e6	1.6e5
1e−3	2	2.1e6	1.7e5	5.0e6	1.6e5
1e−5	2	2.1e6	1.7e5	5.0e6	1.6e5
1e−8	2	2.1e6	1.7e5	4.7e6	1.6e5
f_4 in 20-D, N=15, mFE=300000					
10	0	*25e+0*	*20e+0*	*30e+0*	1.3e5
1	.	.	.	.	.
1e−1	.	.	.	.	.
1e−3	.	.	.	.	.
1e−5	.	.	.	.	.
1e−8	.	.	.	.	.
f_5 in 5-D, N=15, mFE=550					
10	15	3.3e2	3.1e2	3.6e2	3.3e2
1	15	3.9e2	3.4e2	4.5e2	3.9e2
1e−1	15	4.3e2	3.5e2	5.3e2	4.3e2
1e−3	15	4.3e2	3.5e2	5.3e2	4.3e2
1e−5	15	4.3e2	3.5e2	5.3e2	4.3e2
1e−8	15	4.3e2	3.5e2	5.3e2	4.3e2
f_5 in 20-D, N=15, mFE=300000					
10	10	2.7e5	5.7e3	6.0e5	1.2e5
1	6	5.4e5	4.1e4	1.2e6	9.5e4
1e−1	6	5.4e5	4.1e4	1.3e6	9.5e4
1e−3	6	5.4e5	4.1e4	1.3e6	9.5e4
1e−5	6	5.4e5	4.1e4	1.2e6	9.5e4
1e−8	6	5.4e5	4.1e4	1.3e6	9.5e4
f_6 in 5-D, N=15, mFE=33420					
10	15	9.4e2	3.3e2	1.2e3	9.4e2
1	15	4.1e3	6.9e2	1.1e4	4.1e3
1e−1	15	4.4e3	9.0e2	1.2e4	4.4e3
1e−3	15	4.9e3	1.4e3	1.2e4	4.9e3
1e−5	15	5.3e3	1.9e3	1.3e4	5.3e3
1e−8	15	6.0e3	2.5e3	1.3e4	6.0e3
f_6 in 20-D, N=15, mFE=300000					
10	13	6.5e4	2.3e3	3.0e5	1.8e4
1	13	6.6e4	3.4e3	3.0e5	1.9e4
1e−1	13	6.6e4	4.4e3	3.0e5	2.0e4
1e−3	13	6.8e4	6.4e3	3.1e5	2.2e4
1e−5	13	7.0e4	8.2e3	3.1e5	2.4e4
1e−8	13	7.3e4	1.1e4	3.1e5	2.7e4
f_7 in 5-D, N=15, mFE=18760					
10	15	1.7e2	3.0e1	3.3e2	1.7e2
1	15	2.3e3	5.1e2	1.2e4	2.3e3
1e−1	15	7.1e3	6.6e2	1.6e4	7.1e3
1e−3	15	7.4e3	9.4e2	1.6e4	7.4e3
1e−5	15	7.4e3	9.4e2	1.6e4	7.4e3
1e−8	15	7.8e3	1.2e3	1.6e4	7.8e3
f_7 in 20-D, N=15, mFE=300000					
10	15	1.8e3	1.3e3	2.1e3	1.8e3
1	14	1.3e5	2.9e4	2.3e5	1.1e5
1e−1	2	2.1e6	2.1e5	5.0e6	1.9e5
1e−3	0	*55e−2*	*44e−3*	*97e−2*	2.0e5
1e−5	.	.	.	.	.
1e−8	.	.	.	.	.
f_8 in 5-D, N=15, mFE=8305					
10	15	5.2e2	4.4e2	6.1e2	5.2e2
1	15	1.9e3	1.2e3	2.9e3	1.9e3
1e−1	15	2.7e3	1.9e3	3.8e3	2.7e3
1e−3	15	3.4e3	2.4e3	4.6e3	3.4e3
1e−5	15	3.7e3	2.7e3	4.8e3	3.7e3
1e−8	15	4.1e3	3.1e3	5.3e3	4.1e3
f_8 in 20-D, N=15, mFE=45240					
10	15	1.3e4	9.3e3	1.7e4	1.3e4
1	15	2.6e4	2.3e4	3.1e4	2.6e4
1e−1	15	3.1e4	2.8e4	3.6e4	3.1e4
1e−3	15	3.6e4	3.3e4	4.1e4	3.6e4
1e−5	15	3.8e4	3.5e4	4.3e4	3.8e4
1e−8	15	4.0e4	3.7e4	4.5e4	4.0e4
f_9 in 5-D, N=15, mFE=4030					
10	15	5.2e2	4.6e2	5.9e2	5.2e2
1	15	1.4e3	1.1e3	1.7e3	1.4e3
1e−1	15	2.3e3	2.0e3	2.6e3	2.3e3
1e−3	15	2.9e3	2.6e3	3.2e3	2.9e3
1e−5	15	3.2e3	2.9e3	3.5e3	3.2e3
1e−8	15	3.6e3	3.3e3	3.9e3	3.6e3
f_9 in 20-D, N=15, mFE=79105					
10	15	1.2e4	9.0e3	1.6e4	1.2e4
1	15	2.8e4	2.3e4	3.1e4	2.8e4
1e−1	15	3.3e4	2.8e4	3.6e4	3.3e4
1e−3	15	3.8e4	3.2e4	4.2e4	3.8e4
1e−5	15	4.0e4	3.4e4	4.4e4	4.0e4
1e−8	15	4.2e4	3.6e4	4.6e4	4.2e4
f_{10} in 5-D, N=15, mFE=3890					
10	15	1.6e3	1.1e3	2.2e3	1.6e3
1	15	2.3e3	1.7e3	2.7e3	2.3e3
1e−1	15	2.5e3	2.3e3	2.9e3	2.5e3
1e−3	15	2.8e3	2.5e3	3.1e3	2.8e3
1e−5	15	3.1e3	2.8e3	3.3e3	3.1e3
1e−8	15	3.5e3	3.2e3	3.7e3	3.5e3
f_{10} in 20-D, N=15, mFE=48760					
10	15	2.3e4	1.9e4	2.7e4	2.3e4
1	15	2.9e4	2.3e4	3.6e4	2.9e4
1e−1	15	3.3e4	2.6e4	3.9e4	3.3e4
1e−3	15	3.7e4	3.2e4	4.4e4	3.7e4
1e−5	15	3.9e4	3.5e4	4.6e4	3.9e4
1e−8	15	4.1e4	3.8e4	4.7e4	4.1e4
f_{11} in 5-D, N=15, mFE=5285					
10	15	1.9e3	6.2e2	3.0e3	1.9e3
1	15	2.9e3	2.0e3	4.0e3	2.9e3
1e−1	15	3.1e3	2.2e3	4.2e3	3.1e3
1e−3	15	3.5e3	2.5e3	4.6e3	3.5e3
1e−5	15	3.8e3	2.8e3	4.9e3	3.8e3
1e−8	15	4.2e3	3.2e3	5.3e3	4.2e3
f_{11} in 20-D, N=15, mFE=300000					
10	0	*85e+0*	*70e+0*	*10e+1*	2.0e5
1	.	.	.	.	.
1e−1	.	.	.	.	.
1e−3	.	.	.	.	.
1e−5	.	.	.	.	.
1e−8	.	.	.	.	.
f_{12} in 5-D, N=15, mFE=16160					
10	15	1.9e3	1.0e3	5.4e3	1.9e3
1	15	2.7e3	1.1e3	6.9e3	2.7e3
1e−1	15	4.3e3	1.4e3	9.2e3	4.3e3
1e−3	15	5.7e3	3.0e3	1.0e4	5.7e3
1e−5	15	7.2e3	3.3e3	1.2e4	7.2e3
1e−8	15	8.6e3	3.7e3	1.3e4	8.6e3
f_{12} in 20-D, N=15, mFE=113470					
10	15	6.5e3	2.7e3	2.2e4	6.5e3
1	15	2.4e4	3.1e3	4.3e4	2.4e4
1e−1	15	3.9e4	2.3e4	5.7e4	3.9e4
1e−3	15	5.2e4	3.5e4	6.9e4	5.2e4
1e−5	15	6.4e4	4.8e4	8.0e4	6.4e4
1e−8	15	7.4e4	5.8e4	9.0e4	7.4e4
f_{13} in 5-D, N=15, mFE=6930					
10	15	8.2e2	5.9e2	9.5e2	8.2e2
1	15	1.5e3	9.3e2	2.4e3	1.5e3
1e−1	15	2.3e3	1.3e3	3.1e3	2.3e3
1e−3	15	3.4e3	2.9e3	4.1e3	3.4e3
1e−5	15	4.6e3	4.2e3	5.1e3	4.6e3
1e−8	15	6.3e3	5.5e3	6.9e3	6.3e3
f_{13} in 20-D, N=15, mFE=300000					
10	15	3.1e3	1.9e3	2.5e3	3.1e3
1	15	3.1e4	2.9e3	1.2e5	3.1e4
1e−1	15	8.0e4	1.6e4	1.7e5	8.0e4
1e−3	15	1.1e5	5.7e4	1.7e5	1.1e5
1e−5	12	2.3e5	8.8e4	4.7e5	1.5e5
1e−8	2	2.0e6	9.1e4	4.7e6	9.0e4
f_{14} in 5-D, N=15, mFE=3990					
10	15	1.4e1	6.0e0	2.6e1	1.4e1
1	15	2.9e2	1.3e2	3.7e2	2.9e2
1e−1	15	5.0e2	4.1e2	5.4e2	5.0e2
1e−3	15	1.0e3	9.1e2	1.1e3	1.0e3
1e−5	15	1.9e3	1.7e3	2.2e3	1.9e3
1e−8	15	3.5e3	3.3e3	3.8e3	3.5e3
f_{14} in 20-D, N=15, mFE=37460					
10	15	4.1e2	3.4e2	4.8e2	4.1e2
1	15	8.9e2	7.9e2	1.0e3	8.9e2
1e−1	15	1.4e3	1.2e3	1.6e3	1.4e3
1e−3	15	4.7e3	4.1e3	5.2e3	4.7e3
1e−5	15	1.5e4	1.4e4	1.6e4	1.5e4
1e−8	15	3.5e4	3.2e4	3.7e4	3.5e4
f_{15} in 5-D, N=15, mFE=300000					
10	15	1.2e3	6.9e2	1.7e3	1.2e3
1	15	4.0e4	1.2e4	7.0e4	4.0e4
1e−1	10	3.1e5	8.2e4	6.8e5	1.6e5
1e−3	10	3.1e5	8.2e4	7.3e5	1.6e5
1e−5	10	3.1e5	8.2e4	6.8e5	1.6e5
1e−8	10	3.1e5	8.3e4	6.5e5	1.6e5
f_{15} in 20-D, N=15, mFE=300000					
10	0	*20e+0*	*14e+0*	*25e+0*	1.4e5
1	.	.	.	.	.
1e−1	.	.	.	.	.
1e−3	.	.	.	.	.
1e−5	.	.	.	.	.
1e−8	.	.	.	.	.
f_{16} in 5-D, N=15, mFE=50040					
10	15	3.7e2	1.0e2	6.2e2	3.7e2
1	15	2.5e3	6.7e2	6.5e3	2.5e3
1e−1	15	3.5e3	1.0e3	9.8e3	3.5e3
1e−3	15	9.0e3	1.4e3	2.9e4	9.0e3
1e−5	15	9.7e3	1.8e3	2.9e4	9.7e3
1e−8	15	1.0e4	2.2e3	2.9e4	1.0e4
f_{16} in 20-D, N=15, mFE=300000					
10	15	2.6e4	5.4e3	6.0e4	2.6e4
1	15	4.8e4	6.5e3	1.0e5	4.8e4
1e−1	6	5.5e5	4.7e4	1.3e6	9.6e4
1e−3	0	*15e−2*	*26e−3*	*38e−2*	1.3e5
1e−5	.	.	.	.	.
1e−8	.	.	.	.	.
f_{17} in 5-D, N=15, mFE=80870					
10	15	2.9e1	5.0e0	6.4e1	2.9e1
1	15	4.8e2	4.2e2	6.2e2	4.8e2
1e−1	15	1.7e3	7.3e2	7.2e3	1.7e3
1e−3	15	2.5e3	1.4e3	8.0e3	2.5e3
1e−5	15	1.0e4	2.2e3	2.7e4	1.0e4
1e−8	15	2.6e4	3.5e3	5.3e4	2.6e4
f_{17} in 20-D, N=15, mFE=300000					
10	15	2.2e2	1.0e2	3.7e2	2.2e2
1	15	1.3e3	1.1e3	1.5e3	1.3e3
1e−1	15	4.5e3	2.6e3	7.9e3	4.5e3
1e−3	7	4.7e5	3.7e4	1.1e6	1.3e5
1e−5	0	*10e−4*	*46e−5*	*15e−4*	1.6e5
1e−8	.	.	.	.	.
f_{18} in 5-D, N=15, mFE=300000					
10	15	3.2e2	1.4e2	4.5e2	3.2e2
1	15	1.1e3	6.0e2	1.1e3	1.1e3
1e−1	15	1.6e3	1.1e3	1.5e3	1.6e3
1e−3	15	1.1e4	2.0e3	2.6e4	1.1e4
1e−5	13	6.7e4	3.5e3	3.1e5	2.1e4
1e−8	10	1.8e5	9.9e3	6.1e5	3.3e4
f_{18} in 20-D, N=15, mFE=300000					
10	15	9.4e2	7.7e2	1.1e3	9.4e2
1	15	3.0e3	2.0e3	3.6e3	3.0e3
1e−1	15	2.2e4	3.9e3	5.1e4	2.2e4
1e−3	0	*16e−3*	*54e−4*	*35e−3*	1.3e5
1e−5	.	.	.	.	.
1e−8	.	.	.	.	.
f_{19} in 5-D, N=15, mFE=300010					
10	15	3.9e1	3.0e0	9.3e1	3.9e1
1	15	2.9e3	8.4e2	6.1e3	2.9e3
1e−1	6	5.2e5	1.4e4	1.3e6	6.9e4
1e−3	1	4.2e6	3.1e5	1.0e7	9.5e3
1e−5	1	4.2e6	3.1e5	9.9e6	9.9e3
1e−8	1	4.2e6	3.1e5	9.9e6	1.0e4
f_{19} in 20-D, N=15, mFE=300000					
10	15	2.6e2	1.1e2	3.8e2	2.6e2
1	0	*41e−1*	*38e−1*	*44e−1*	2.0e5
1e−1	.	.	.	.	.
1e−3	.	.	.	.	.
1e−5	.	.	.	.	.
1e−8	.	.	.	.	.
f_{20} in 5-D, N=15, mFE=300000					
10	15	1.3e2	1.0e2	1.5e2	1.3e2
1	15	4.2e3	8.5e2	8.6e3	4.2e3
1e−1	4	9.6e5	1.4e5	2.2e6	1.4e5
1e−3	4	9.6e5	1.4e5	2.0e6	1.4e5
1e−5	4	9.6e5	1.4e5	2.2e6	1.4e5
1e−8	4	9.6e5	1.4e5	2.2e6	1.4e5
f_{20} in 20-D, N=15, mFE=300000					
10	15	6.0e2	4.8e2	7.1e2	6.0e2
1	1	4.2e6	3.1e5	9.9e6	1.0e4
1e−1	0	*12e−1*	*11e−1*	*13e−1*	1.6e5
1e−3	.	.	.	.	.
1e−5	.	.	.	.	.
1e−8	.	.	.	.	.
f_{21} in 5-D, N=15, mFE=300000					
10	15	1.0e2	1.0e1	2.8e2	1.0e2
1	12	1.1e5	4.1e2	3.2e5	3.6e4
1e−1	10	2.0e5	5.0e2	6.0e5	4.9e4
1e−3	10	2.0e5	6.6e2	6.0e5	4.9e4
1e−5	10	2.0e5	7.8e2	6.0e5	5.0e4
1e−8	10	2.0e5	9.9e2	6.0e5	5.0e4
f_{21} in 20-D, N=15, mFE=300000					
10	13	4.7e4	5.5e2	3.0e5	7.4e2
1	6	4.5e5	9.2e2	1.2e6	1.3e3
1e−1	4	8.3e5	1.1e3	2.1e6	1.1e3
1e−3	4	8.3e5	1.5e3	2.1e6	1.6e3
1e−5	4	8.3e5	1.9e3	2.1e6	2.1e3
1e−8	4	8.3e5	2.8e3	2.1e6	2.8e3
f_{22} in 5-D, N=15, mFE=300000					
10	15	2.1e2	8.5e1	3.8e2	2.1e2
1	12	1.0e5	4.0e2	3.1e5	2.7e4
1e−1	8	2.7e5	4.0e2	9.0e5	8.1e3
1e−3	7	3.5e5	9.0e2	9.1e5	9.6e3
1e−5	7	3.5e5	1.2e3	9.1e5	9.8e3
1e−8	7	3.5e5	1.4e3	9.1e5	1.0e4
f_{22} in 20-D, N=15, mFE=300000					
10	10	1.5e5	5.4e2	6.0e5	8.2e2
1	2	2.0e6	1.1e3	4.5e6	1.1e3
1e−1	0	*73e−1*	*69e−2*	*18e+0*	4.5e3
1e−3	.	.	.	.	.
1e−5	.	.	.	.	.
1e−8	.	.	.	.	.
f_{23} in 5-D, N=15, mFE=300010					
10	15	4.3e0	1.0e0	9.0e0	4.3e0
1	15	2.2e4	2.7e3	3.4e4	2.2e4
1e−1	1	4.3e6	3.6e5	1.0e7	6.2e4
1e−3	1	4.3e6	3.6e5	9.7e6	6.3e4
1e−5	1	4.3e6	3.6e5	1.0e7	6.3e4
1e−8	1	4.3e6	3.6e5	1.0e7	6.4e4
f_{23} in 20-D, N=15, mFE=300000					
10	15	6.1e0	2.0e0	1.4e1	6.1e0
1	0	*18e−1*	*13e−1*	*19e−1*	1.6e5
1e−1	.	.	.	.	.
1e−3	.	.	.	.	.
1e−5	.	.	.	.	.
1e−8	.	.	.	.	.
f_{24} in 5-D, N=15, mFE=300005					
10	15	2.4e4	5.4e3	4.7e4	2.4e4
1	2	2.0e6	1.1e5	4.9e6	8.2e4
1e−1	0	*56e−1*	*93e−2*	*79e−1*	1.1e5
1e−3	.	.	.	.	.
1e−5	.	.	.	.	.
1e−8	.	.	.	.	.
f_{24} in 20-D, N=15, mFE=300000					
10	0	*12e+1*	*10e+1*	*13e+1*	1.3e5
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.

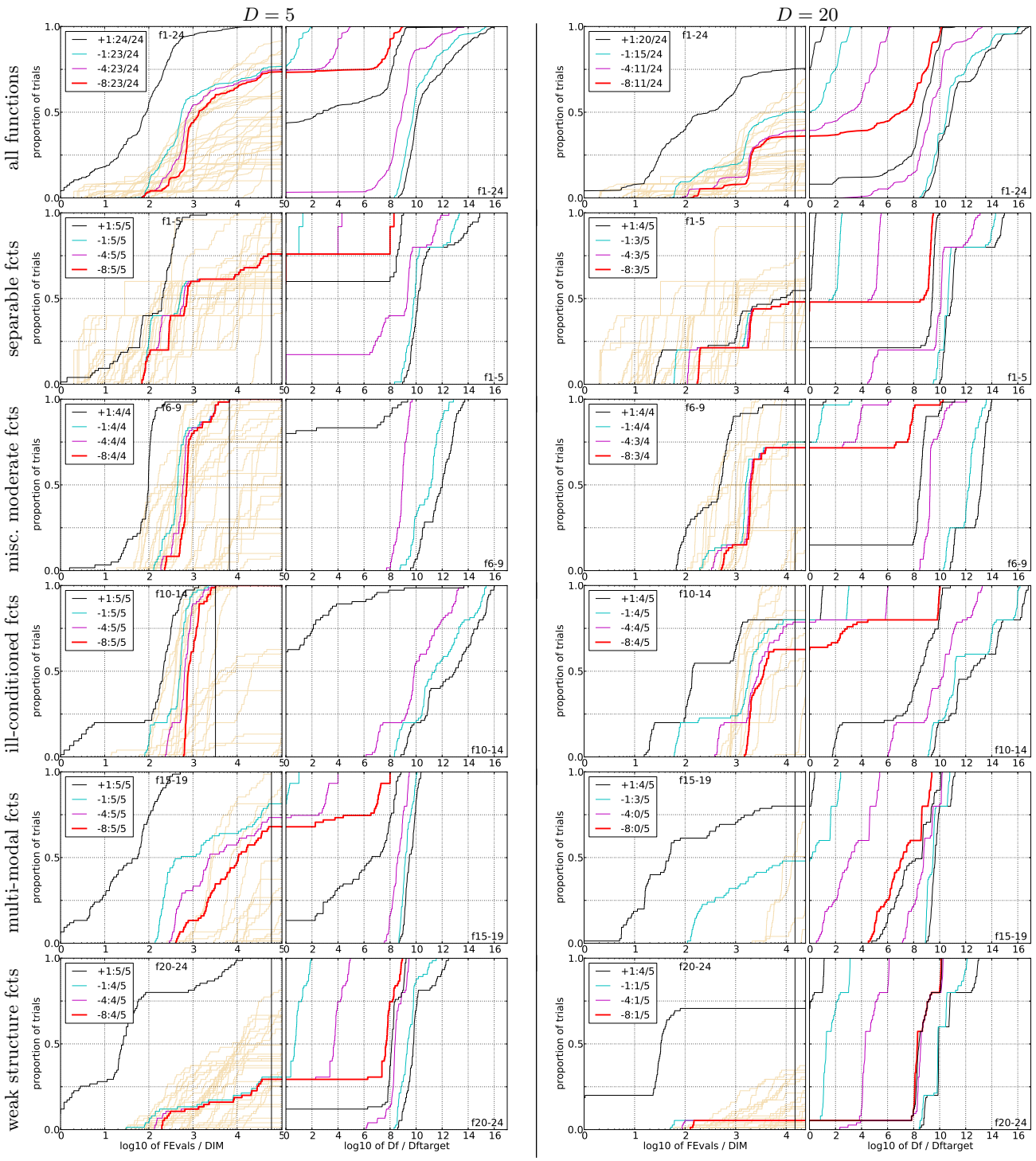

Figure 3: Empirical cumulative distribution functions (ECDFs), plotting the fraction of trials versus running time (left subplots) or versus Δf (right subplots). The thick red line represents the best achieved results. Left subplots: ECDF of the running time (number of function evaluations), divided by search space dimension D, to fall below $f_{\mathrm{opt}} + \Delta f$ 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). 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. Light brown lines in the background show ECDFs for target value 10^{-8} of all algorithms benchmarked during BBOB-2009.

Table 2: ERT **loss ratio (see Figure 4) compared to the respective best result from BBOB-2009 for budgets given in the first column. The last row RL_{US}/D gives the number of function evaluations in unsuccessful runs divided by dimension. Shown are the smallest, 10%-ile, 25%-ile, 50%-ile, 75%-ile and 90%-ile value (smaller values are better).**

	f_1–f_{24} **in 5-D**, maxFE/D=60002					
#FEs/D	best	10%	25%	**med**	75%	90%
2	1.2	1.4	2.1	2.9	4.7	10
10	1.6	3.1	3.5	5.0	6.6	50
100	2.3	3.4	5.8	7.7	15	41
1e3	0.46	0.67	3.3	7.4	29	75
1e4	0.83	2.5	4.3	9.7	51	3.2e2
1e5	0.83	2.5	4.3	14	1.1e2	3.5e2
RL_{US}/D	6e4	6e4	6e4	6e4	6e4	6e4
	f_1–f_{24} **in 20-D**, maxFE/D=15000					
#FEs/D	best	10%	25%	**med**	75%	90%
2	1.0	3.6	11	31	40	40
10	1.7	4.8	7.1	10	27	2.0e2
100	0.72	1.4	4.0	12	26	59
1e3	1.1	1.6	9.2	38	1.2e2	2.4e2
1e4	1.9	4.1	7.9	54	2.2e2	1.1e3
1e5	1.9	4.8	19	91	7.2e2	5.1e3
RL_{US}/D	1e4	1e4	1e4	1e4	2e4	2e4

7. REFERENCES

[1] B. Addis. *Global Optimization using Local Searches*. PhD thesis, Department Systems and Computer Science, of University of Florence, 2005.

[2] A. Auger and N. Hansen. A Restart CMA Evolution Strategy With Increasing Population Size. In B. McKay et al., editors, *Proc. 2005 Congress on Evolutionary Computation (CEC'05)*, pages 1769–1776, Piscataway NJ, 2005. IEEE Press.

[3] 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.

[4] N. Hansen, A. Auger, S. Finck, and R. Ros. Real-parameter black-box optimization benchmarking 2010: Experimental setup. Technical Report RR-7215, INRIA, 2010.

[5] 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.

[6] N. Hansen and A. Ostermeier. Completely Derandomized Self-Adaptation in Evolution Strategies. *IEEE Computational Intelligence Magazine*, 9(2):159–195, 2001.

[7] M. Preuss. Niching Prospects. In *Proceedings of Bioinspired Optimization Methods and their Applications (BIOMA 2006)*, pages 25–34. Jozef Stefan Institute, Ljubljana, Slovenia, 2006.

[8] M. Preuss, L. Schönemann, and M. Emmerich. Counteracting genetic drift and disruptive recombination in $(\mu + /, \lambda)$-EA on multimodal fitness landscapes. In *Proceedings of Genetic and Evolutionary Computation Conference (GECCO 2005)*, volume 1, pages 865–872, New York, 2005. ACM Press.

[9] O. M. Shir and T. Baeck. Niche radius adaptation in the cma-es niching algorithm. In *Proceedings of Parallel Problem Solving from Nature (PPSN 2006), Lecture Notes in Computer Science*, volume 4193, pages 142–151, Berlin, Germany, 2006. Springer-Verlag.

[10] C. Stoean, M. Preuss, R. Stoean, and D. Dumitrescu. Disburdening the species conservation evolutionary algorithm of arguing with radii. In *Proceedings of Genetic and Evolutionary Computation Conference (GECCO 2007)*, pages 1420–1427, New York, NY, USA, 2007. ACM Press.

[11] R. K. Ursem. Multinational evolutionary algorithms. In *Proceedings of Congress of Evolutionary Computation (CEC 1999)*, volume 3, pages 1633–1640, Piscataway, NJ, 1999. IEEE Press.