Benchmarking a MOS-based algorithm on the BBOB-2010 Noiseless Function Testbed

Draft version *

A. LaTorre
Department of Computer Systems Architecture and Technology
Facultad de Informática
Universidad Politécnica de Madrid, Spain
atorre@fi.upm.es

S. Muelas
Department of Computer Systems Architecture and Technology
Facultad de Informática
Universidad Politécnica de Madrid, Spain
smuelas@fi.upm.es

J.M. Peña
Department of Computer Systems Architecture and Technology
Facultad de Informática
Universidad Politécnica de Madrid, Spain
jmpena@fi.upm.es

ABSTRACT

In this paper, a hybrid algorithm combining Differential Evolution and CMA-ES is presented and benchmarked on the BBOB 2010 noiseless testbed. The hybrid algorithm has been constructed within the Multiple Offspring Sampling framework, which allows the seamless combination of multiple metaheuristics in a dynamic algorithm capable of adjusting the participation of each of the composing algorithms according to their current performance. The experimental results show a robust behavior of the algorithm and a good scalability as the dimensionality increases.

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, Black-box optimization, Continuous optimization, CMA-ES, Differential Evolution, Multiple Offspring Sampling

1. INTRODUCTION

In this contribution, the Multiple Offspring Sampling (MOS) framework has been applied to the Black Box Optimization 2010 Noiseless Function Testbed. This framework allows the combination of different evolutionary models following an HRH approach in which the number of evaluations that each algorithm can carry out is dynamically adjusted. For this paper, the IPOP-CMAE-ES [1] and the Differential Evolution (DE) algorithm [8] have been combined within this framework in a multistart strategy and has been benchmarked on 24 different functions. Detailed results regarding the number of evaluations needed to reach a target function on each dimension along with the CPU times are also given.

*Submission deadline: March 25th.

GECCO'10, July 7–11, 2010, Portland, Oregon, USA.

2. ALGORITHM PRESENTATION

Multiple Offspring Sampling (MOS) is a framework for the development of Dynamic Hybrid Evolutionary Algorithms. MOS provides the functional formalization necessary to design this type of algorithms, as well as the tools to identify and select the best performing configuration for the problem under study. In this context, the hybridization of several algorithms can lead to the following two situations:

- A collaborative synergy emerges among the different algorithms that improves the performance of the best one when it is used individually.
- A competitive selection of the best one takes place, in which a similar performance (often the same) is obtained with a minimum overhead.

In MOS, a key term is the concept of *technique*, which is a mechanism, decoupled from the main algorithm, to generate new candidate solutions. This means that, within a MOS-based algorithm, several reproductive mechanisms can be used simultaneously, and it is the main algorithm which selects among the available optimization techniques the most appropriate for the particular problem and search phase. A more concrete definition for these reproductive mechanisms follows:

Definition 1. A MOS reproductive technique is a mechanism to create new individuals in which: (a) a particular evolutionary algorithm model, (b) an appropriate solution encoding, (c) specific operators (if required), and (d) necessary parameters have been defined.

Furthermore, the use of multiple reproductive mechanisms simultaneously has to be controlled in some way. The MOS

Algorithm 1 HRH MOS Algorithm

1: Create initial overall population of candidate solutions P_0
2: Uniformly distribute participation among the n used techniques $\rightarrow \forall j\ \Pi_0^{(j)} = \frac{|P_0|}{n}$. Each technique produces a subset of individuals according to its participation ($\Pi_0^{(j)}$)
3: Evaluate initial population P_0
4: **while** number of steps not exceeded **do**
5: Update Quality of T_j computed as the average quality of all the individuals created by technique T_j in the previous step
6: Update participation ratios from Quality values computed in Step 5 $\rightarrow \forall j\ \Pi_{i+1}^{(j)} = PF(Q_i^{(j)})$
7: Update FEs allocated for each technique at this step: $\rightarrow \forall j\ FEs_i^{(j)} = \Pi_{i+1}^{(j)} \cdot FEs_i$
8: **for** every available technique T_j **do**
9: **while** $FEs_i^{(j)}$ not exceeded **do**
10: Evolve
11: **end while**
12: **end for**
13: **end while**

framework offers two groups of functions to deal with this issue: Quality and Participation functions. The first group of functions evaluate how good a set of new individuals is from the point of view of a desirable characteristic. The second group of functions consider the quality values computed by the first group and adjust the number of new individuals that each reproductive technique will be allowed to generate in the next step of the search. This way, the algorithm is able to dynamically adjust the participation of each of the available techniques and exploit the benefits of each of them at different stages of the search process.

Finally, the Multiple Offspring Sampling framework allows the development of both HTH and HRH algorithms (according to Talbi's nomenclature [9]). As the proposed algorithm is of the HRH type, only this type of algorithms will be reviewed. In an HRH hybrid algorithm, the available techniques are used in sequence, one after the other, each of them reusing the output population of the previous technique. This approach fits better when there are non-population-based techniques, such as local searches, as techniques are not constrained to produce a % of the common population. In this case, the search process is divided into a fixed number of *steps* that is established at the beginning of the execution. Each step is assigned an amount of Fitness Evaluations (FEs_i in Algorithm 1), which are distributed by the Participation Function (PF). Each technique can manage its number of allocated FEs at each step of the algorithm ($FEs_i^{(j)}$) in its own particular way. For example, a population-based technique, such as Differential Evolution, could execute several iterations of the algorithm, whereas a Local Search could decide to spend all its assigned evaluations in improving just one individual. The quality of the new individuals of each technique will be averaged at the end of the whole set of evaluations, as the division of the search into generations depends on each of the techniques. A pseudocode of this approach is given in Algorithm 1. Further information about the MOS framework can be found in [6].

In this contribution, an HRH Dynamic algorithm is proposed. This algorithm combines the explorative/exploitative strength of two heuristic search methods, that separately have proven to obtain very competitive results in either low or high dimensional problems. These algorithms are: the IPOP-CMA-ES algorithm [1], the best algorithm of the "'*Special Session on Real-Parameter Optimization*" held at the CEC 2005 Congress, and the DE algorithm [8] which has demonstrated to obtain competitive results when executed independently and when combined with other algorithms [3, 7].

For the adjustment of the participation of each technique in the overall search process, a new Quality Function (QF) has been proposed. This QF takes into account two desirable characteristics in a search algorithm: the Average Fitness Increment of the newly created individuals after a set of allocated Fitness Evaluations and the number of times that these improvements take place (Equation 1).

$$Q_i^{(j)} = \begin{cases} \Sigma_{i-1}^{(j)} & if\ \Sigma_{i-1}^{(j)} > \Sigma_{i-1}^{(k)} \wedge \\ & \quad \Gamma_{i-1}^{(j)} > \Gamma_{i-1}^{(k)} \qquad \forall j,k \in [1,n] \\ \Gamma_{i-1}^{(j)} & otherwise \end{cases}$$

$$Q_i^{(j)} \equiv \text{Quality of technique } T_j \text{ in step } i$$
$$\Sigma_i^{(j)} \equiv \text{Average Fitness Increment of } T_j \text{ in step } i$$
$$\Gamma_i^{(j)} \equiv \text{Number of Fitness improvements of } T_j \text{ in step } i \quad (1)$$

This Quality Function uses the Average Fitness Increment as the effective QF only if there is consensus among both measures. If this is not the case, the raw number of fitness improvements is used. The logic behind this function is that, in some functions, the use of the Average Fitness Increment QF could be very elitist. In some particular situations, a technique which is not carrying out an effective search could introduce, for some reason, a large increment in the average fitness value of the new individuals. This could be due, for example, to a recombination of poor solutions. In such a case, it is easy for a technique to improve previous solutions. However, it could be more adequate to carry out small changes to good individuals in order to find the right "path" to the global optimum rather than carrying out substantial modifications to poor solutions. For this reason, a consensus of both measures is required in order to apply the more elitist Average Fitness Increment QF. If this is not the case, the number of fitness improvements is used to guarantee a softer adjustment of participation.

The quality values computed by this QF are used by a Dynamic Participation Function to adjust the number of Fitness Evaluations allocated for each technique at each step (Equation 2). This PF computes, at each step, a trade-off factor for each technique, $\Delta_i^{(j)}$, that represents the decrease in participation for the $j-th$ technique at the $i-th$ step, for every technique except the best performing ones. These techniques will increase their participation by the sum of all those $\Delta_i^{(j)}$ divided by the number of techniques with the best quality values.

$$PF_{dyn}(Q_i^{(j)}) = \begin{cases} \Pi_{i-1}^{(j)} + \eta & \text{if } j \in best, \\ \Pi_{i-1}^{(j)} - \Delta_i^{(j)} & \text{otherwise} \end{cases} \quad (2)$$

$$\eta = \frac{\sum_{k \notin best} \Delta_i^{(k)}}{|best|}$$

$$best = \{l \ / \ Q_i^{(l)} \geq Q_i^{(m)} \ \forall l, m \in [1, n]\}$$

The above-mentioned $\Delta_i^{(j)}$ values are computed as shown in Equation 3. These $\Delta_i^{(j)}$ factors are computed from the relative difference between the quality of the *best* and the $j-th$ techniques, n being the number of available techniques. In this equation, ξ represents a reduction factor, i.e., the ratio that is transferred from one technique to the other(s) (usually set to a value of 0.05). Finally, a minimum participation ratio can be established to guarantee that all the techniques are represented through all the search. This is done to avoid, if possible, premature convergence to undesired solutions caused by a technique that obtains all the participation in the early steps of the search and quickly converges to poor regions of the solution space, preventing the other techniques to collaborate at later stages of the process, in which they could be more beneficial.

$$\Delta_i^{(j)} = \xi \cdot \frac{Q_i^{(best)} - Q_i^{(j)}}{Q_i^{(best)}} \cdot \Pi_{i-1}^{(j)} \quad \forall j \in [1, n] \ / \ j \neq best \quad (3)$$

To summarize, the presented algorithm works as follows. All the available techniques are allocated the same number of FEs at the beginning of the execution. At the end of each step, the quality of the new solutions created by each technique is evaluated and, based on this quality, its participation ratio is adjusted accordingly. This participation ratio is used to compute the number of FEs that each technique will be allowed to use in the next step of the search. If a minimum participation ratio has been established, then the number of FEs can not go below this threshold.

Finally, a restart mechanism, similar to the one used by the IPOP-CMA-ES algorithm, was also used within the proposed algorithm. With this strategy, the algorithm is stopped whenever a restart stopping criteria is met, reinitializing the population and increasing its size by a factor of two until a maximum population size is reached.

3. EXPERIMENTAL PROCEDURE

The results reported for this work have been obtained from 15 independent executions executed on the computer configuration displayed in Table 1.

Table 1: Computer Configuration

PC	Intel Xeon 8 cores 1.86Ghz CPU
Operating System	Ubuntu Linux 8.04
Prog. Language	C++
Compiler	GNU C++ 4.3.2

Regarding the parameter tuning, no thorough parameter study has been conducted for this work. The parameters of the algorithm were selected based on the extensive parameter tuning that was carried out for the HRH algorithm presented in [7] and for a similar study that considers the same benchmark and experiments with a MOS based algorithm, submitted for publication to an international journal and currently[1] under review. Table 2 displays the final values that were selected for this experimentation, both for the DE, the CMA-ES and also for the main algorithm.

Table 2: Parameters of the algorithm

Parameter	Value
Initial Population Size	15
Maximum Pop. Size (after restarts)	6400
DE CR	0.5
DE F	0.5
DE Crossover Operator	Exponential
DE Selection Operator	Tournament 2
DE Model	classic
Minimum Participation Ratio	5%
Number of Steps	85

4. CPU TIMING EXPERIMENT

For the timing experiment the proposed algorithm was run on f_8 for at least 30 seconds. This experimentation has been conducted on the aforementioned computer configuration depicted in Table 1. The results of this study are reported in Table 3.

Table 3: CPU Timing

D	2	3	5	10	20	40
runs	182	161	143	112	82	52
seconds x 10^{-6}	3.5	3.9	4.5	4.5	4.7	5.8

The CPU-time per function evaluation grows linearly up to 5 dimensions, probably due to the overhead of the hybridization procedures, and then it gets stabilized for dimensions 5, 10 and 20. Finally, for 40 dimensions, the CPU-time starts to grow again, this time due to the increased complexity for this problem size.

5. RESULTS

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

The overall results in the noiseless testbed are quite satisfactory in terms of achieved precision and scalability. The hybrid algorithm here presented is able to solve 24, 24, 24, 24, 21 and 20 functions out of 24 in 2, 3, 5, 10, 20 and 40 dimensions, respectively.

Compared to the individual use of its composing algorithms, the hybrid algorithm obtains more stable results than any of them. Furthermore, functions $f3$ and $f4$ which are practically unsolvable for the IPOP-CMA-ES algorithm, are now solved thanks to the hybridization with the DE algorithm. On the other hand, the most difficult function for our approach is f_{24}, for which convergence is never reached for dimension 20 or above. Nevertheless, this is somehow

[1]March 2010

Table 5: ERT **loss ratio (see Figure 3) 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=727186					
#FEs/D	best	10%	25%	**med**	75%	90%
2	1.3	1.6	1.9	2.9	4.2	8.5
10	2.3	3.3	3.8	5.1	6.8	50
100	2.8	6.6	8.7	12	16	42
1e3	2.2	2.2	3.2	9.3	25	51
1e4	0.72	1.6	3.1	6.6	34	1.3e2
1e5	1.2	1.6	3.1	5.6	17	2.5e2
1e6	0.52	1.2	2.6	4.7	16	2.7e2
RL_{US}/D	1e5	1e5	1e5	1e5	2e5	3e5
	f_1–f_{24} **in 20-D**, maxFE/D=199139					
#FEs/D	best	10%	25%	**med**	75%	90%
2	1.0	2.4	9.4	31	40	40
10	4.8	6.9	10	51	2.0e2	2.0e2
100	5.6	6.5	8.8	17	24	47
1e3	0.26	1.6	3.2	9.8	44	1.4e2
1e4	0.09	1.1	2.0	3.9	67	2.8e2
1e5	0.07	0.94	2.1	4.0	14	2.2e2
1e6	0.07	0.94	2.1	4.4	30	1.3e2
RL_{US}/D	1e5	1e5	1e5	1e5	2e5	2e5

reasonable, as this function has been designed to be deceptive for Evolution Strategies (and the DE is also unable to deal with it).

It can also be seen that the proposed algorithm achieves one of the best results in terms of ECDFs values, compared with the algorithms presented in the BBOB-2009 workshop, for all the groups of functions, as it can be seen in Figure 2.

Finally, regarding the number of Fitness Evaluations needed to reach a particular precision, it can be higher than for other algorithms, such as the IPOP-CMA-ES when it is used individually. This is normal, as the regulatory mechanisms implemented by the MOS framework need some time to take a decission and adjust the participation of each technique accordingly.

6. CONCLUSIONS

In this paper a hybrid algorithm combining Differential Evolution and CMA-ES has been presented and benchmarked on the BBOB-2010 noiseless testbed. The experimental results show a good performance on all the groups of functions and a good scalability. The proposed algorithm has been able to solve 24, 24, 24, 24, 21 and 20 functions out of 24 in 2, 3, 5, 10, 20 and 40 dimensions, respectively. Furthermore, it obtains better results than its composing algorithms when used individually. Additionally, a comparative analysis with the algorithms presented at the BBOB-2009 workshop reveals that our approach obtains one of the best results in terms of convergence. Further research will investigate with new techniques to complement the two used algorithms in those functions in which the hybrid algorithms obtains worse results. A more thorough study on the control mechanisms, specially those related to the detection of the stagnation and the restart of the search process, could

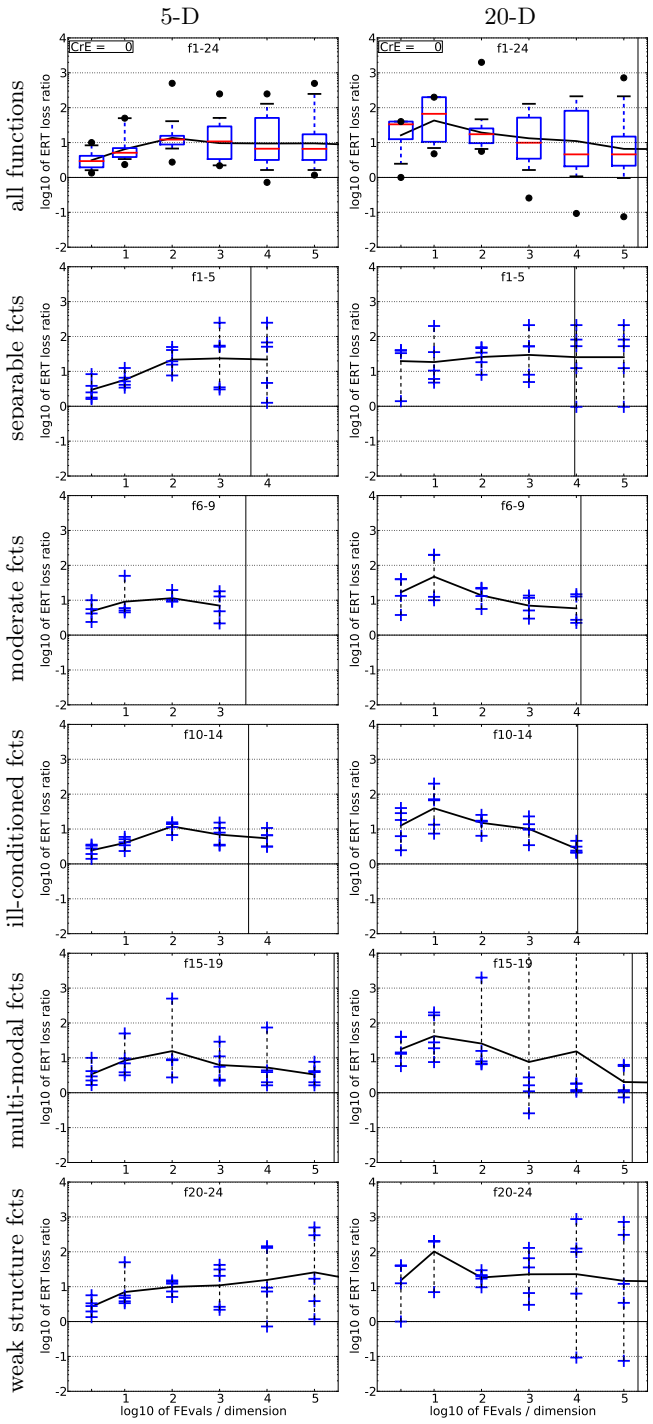

Figure 3: ERT **loss ratio versus given budget** FEvals. **The target value f_t for** ERT **(see Figure 1) is the smallest (best) recorded function value such that** $\mathrm{ERT}(f_t) \leq \mathrm{FEvals}$ **for the presented algorithm. Shown is** FEvals **divided by the respective best** $\mathrm{ERT}(f_t)$ **from BBOB-2009 for functions f_1–f_{24} in 5-D and 20-D. Each** ERT **is multiplied by** $\exp(\mathrm{CrE})$ **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.**

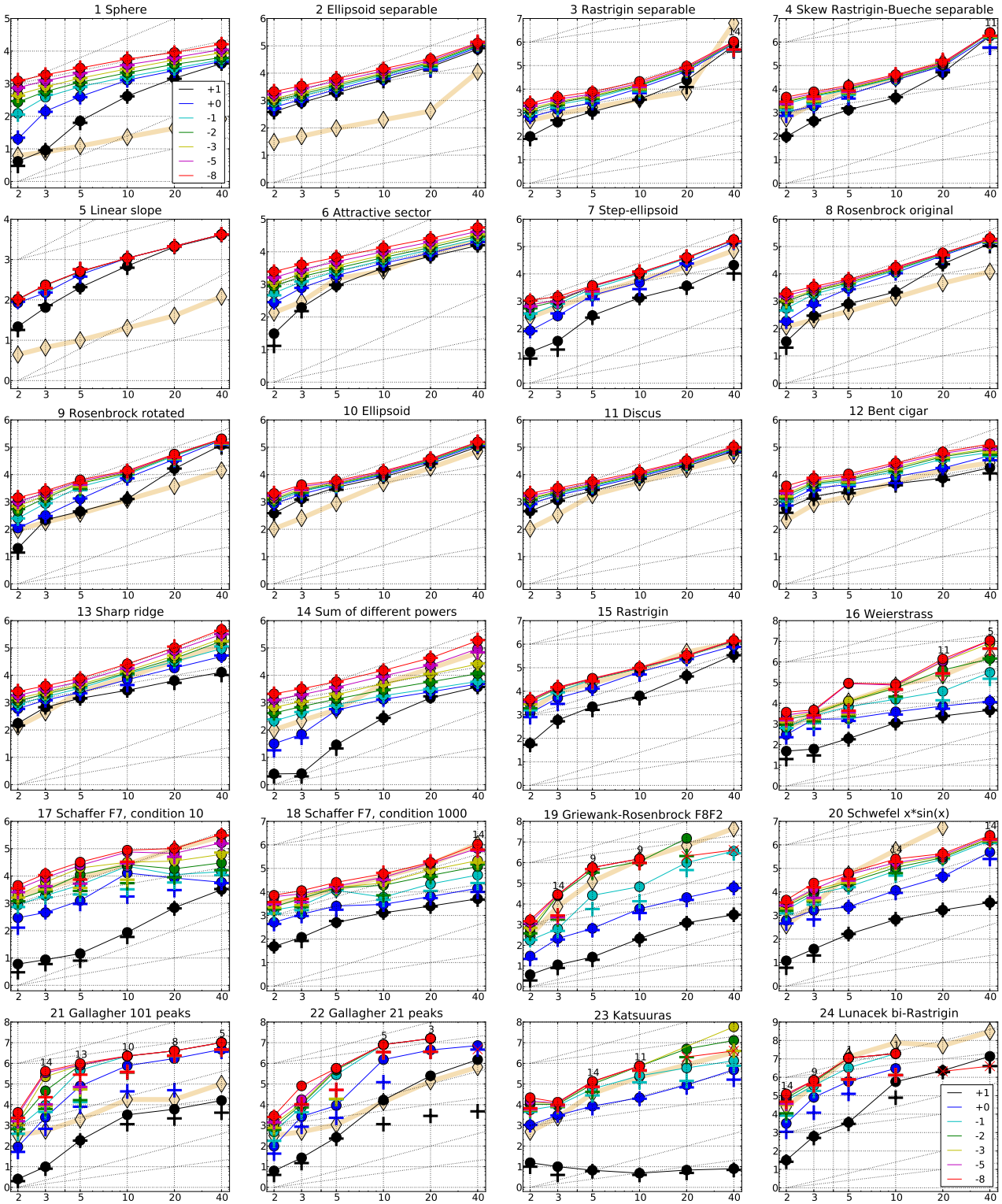

Figure 1: Expected Running Time (ERT, ●) to reach $f_{\mathrm{opt}} + \Delta f$ and median number of f-evaluations from successful trials (+), for $\Delta f = 10^{\{+1,0,-1,-2,-3,-5,-8\}}$ (the exponent is given in the legend of f_1 and f_{24}) versus dimension in log-log presentation. For each function 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	f_1 in 5-D, N=15, mFE=3144 #	ERT	10%	90%	RT_{succ}	f_1 in 20-D, N=15, mFE=10101 #	ERT	10%	90%	RT_{succ}
10	15	7.1e1	2.0e0	1.3e2	7.1e1	15	1.5e3	1.3e3	2.0e3	1.5e3
1	15	4.0e2	2.3e2	5.5e2	4.0e2	15	2.5e3	2.4e3	2.6e3	2.5e3
1e−1	15	8.1e2	7.2e2	9.2e2	8.1e2	15	2.9e3	2.9e3	3.0e3	2.9e3
1e−3	15	1.5e3	1.4e3	1.6e3	1.5e3	15	5.3e3	5.1e3	5.5e3	5.3e3
1e−5	15	2.1e3	1.8e3	2.3e3	2.1e3	15	6.4e3	5.9e3	7.0e3	6.4e3
1e−8	15	3.0e3	2.9e3	3.1e3	3.0e3	15	9.1e3	8.8e3	1.0e4	9.1e3

Δf	f_3 in 5-D, N=15, mFE=20423 #	ERT	10%	90%	RT_{succ}	f_3 in 20-D, N=15, mFE=184160 #	ERT	10%	90%	RT_{succ}
10	15	1.1e3	8.8e2	1.6e3	1.1e3	15	2.4e4	9.4e3	5.1e4	2.4e4
1	15	2.7e3	2.4e3	3.4e3	2.7e3	15	5.6e4	1.5e4	1.2e5	5.6e4
1e−1	15	4.0e3	2.6e3	4.1e3	4.0e3	15	6.8e4	1.7e4	1.4e5	6.8e4
1e−3	15	5.3e3	4.0e3	5.4e3	5.3e3	15	7.5e4	2.1e4	1.5e5	7.5e4
1e−5	15	6.2e3	4.9e3	6.2e3	6.2e3	15	8.2e4	2.5e4	1.6e5	8.2e4
1e−8	15	7.7e3	6.4e3	7.6e3	7.7e3	15	9.5e4	3.1e4	1.7e5	9.5e4

Δf	f_5 in 5-D, N=15, mFE=650 #	ERT	10%	90%	RT_{succ}	f_5 in 20-D, N=15, mFE=2200 #	ERT	10%	90%	RT_{succ}
10	15	2.0e2	8.8e1	2.9e2	2.0e2	15	2.1e3	2.0e3	2.2e3	2.1e3
1	15	4.2e2	3.0e2	5.8e2	4.2e2	15	2.1e3	2.1e3	2.2e3	2.1e3
1e−1	15	5.0e2	3.3e2	6.3e2	5.0e2	15	2.1e3	2.1e3	2.2e3	2.1e3
1e−3	15	5.1e2	3.3e2	6.3e2	5.1e2	15	2.1e3	2.1e3	2.2e3	2.1e3
1e−5	15	5.1e2	3.3e2	6.3e2	5.1e2	15	2.1e3	2.1e3	2.2e3	2.1e3
1e−8	15	5.1e2	3.3e2	6.3e2	5.1e2	15	2.1e3	2.1e3	2.2e3	2.1e3

Δf	f_7 in 5-D, N=15, mFE=6887 #	ERT	10%	90%	RT_{succ}	f_7 in 20-D, N=15, mFE=53357 #	ERT	10%	90%	RT_{succ}
10	15	3.0e2	1.5e2	6.1e2	3.0e2	15	3.6e3	2.8e3	5.0e3	3.6e3
1	15	1.6e3	7.6e2	3.9e3	1.6e3	15	2.6e4	1.5e4	3.9e4	2.6e4
1e−1	15	2.9e3	1.4e3	5.2e3	2.9e3	15	3.7e4	2.1e4	4.6e4	3.7e4
1e−3	15	3.6e3	2.1e3	6.0e3	3.6e3	15	3.9e4	2.3e4	4.8e4	3.9e4
1e−5	15	3.6e3	2.1e3	6.0e3	3.6e3	15	3.9e4	2.3e4	4.8e4	3.9e4
1e−8	15	3.8e3	2.1e3	6.5e3	3.8e3	15	4.0e4	2.4e4	4.8e4	4.0e4

Δf	f_9 in 5-D, N=15, mFE=17883 #	ERT	10%	90%	RT_{succ}	f_9 in 20-D, N=15, mFE=249530 #	ERT	10%	90%	RT_{succ}
10	15	4.5e2	4.1e2	4.8e2	4.5e2	15	1.6e4	1.1e4	2.1e4	1.6e4
1	15	1.3e3	7.6e2	1.6e3	1.3e3	15	3.6e4	2.7e4	4.2e4	3.6e4
1e−1	15	3.6e3	2.1e3	8.0e3	3.6e3	15	4.8e4	3.1e4	4.6e4	4.8e4
1e−3	15	5.2e3	3.1e3	1.4e4	5.2e3	15	5.3e4	3.4e4	4.9e4	5.3e4
1e−5	15	5.7e3	3.7e3	1.5e4	5.7e3	15	5.4e4	3.6e4	5.0e4	5.4e4
1e−8	15	6.4e3	4.4e3	1.5e4	6.4e3	15	5.6e4	3.8e4	5.2e4	5.6e4

Δf	f_{11} in 5-D, N=15, mFE=5992 #	ERT	10%	90%	RT_{succ}	f_{11} in 20-D, N=15, mFE=38386 #	ERT	10%	90%	RT_{succ}
10	15	2.5e3	1.7e3	2.9e3	2.5e3	15	2.1e4	1.8e4	2.6e4	2.1e4
1	15	3.3e3	3.0e3	3.7e3	3.3e3	15	2.3e4	2.0e4	2.7e4	2.3e4
1e−1	15	3.8e3	3.6e3	4.2e3	3.8e3	15	2.5e4	2.3e4	2.9e4	2.5e4
1e−3	15	4.4e3	4.2e3	4.5e3	4.4e3	15	2.8e4	2.6e4	3.2e4	2.8e4
1e−5	15	4.8e3	4.5e3	5.2e3	4.8e3	15	3.0e4	2.7e4	3.5e4	3.0e4
1e−8	15	5.6e3	5.2e3	5.9e3	5.6e3	15	3.3e4	3.0e4	3.7e4	3.3e4

Δf	f_{13} in 5-D, N=15, mFE=8389 #	ERT	10%	90%	RT_{succ}	f_{13} in 20-D, N=15, mFE=154040 #	ERT	10%	90%	RT_{succ}
10	15	1.4e3	1.2e3	1.7e3	1.4e3	15	6.4e3	5.2e3	5.9e3	6.4e3
1	15	2.3e3	2.1e3	2.8e3	2.3e3	15	1.8e4	7.8e3	3.2e4	1.8e4
1e−1	15	3.4e3	2.9e3	3.8e3	3.4e3	15	3.2e4	8.9e3	5.3e4	3.2e4
1e−3	15	4.6e3	4.2e3	5.1e3	4.6e3	15	5.2e4	2.8e4	8.5e4	5.2e4
1e−5	15	5.9e3	5.3e3	6.6e3	5.9e3	15	7.8e4	6.3e4	1.0e5	7.8e4
1e−8	15	7.7e3	7.3e3	8.2e3	7.7e3	15	1.0e5	6.9e4	1.5e5	1.0e5

Δf	f_{15} in 5-D, N=15, mFE=75737 #	ERT	10%	90%	RT_{succ}	f_{15} in 20-D, N=15, mFE=545987 #	ERT	10%	90%	RT_{succ}
10	15	2.2e3	1.1e3	3.0e3	2.2e3	15	4.6e4	3.9e4	7.5e4	4.6e4
1	15	1.3e4	3.1e3	2.3e4	1.3e4	15	2.3e5	1.6e5	3.0e5	2.3e5
1e−1	15	3.1e4	1.3e4	6.4e4	3.1e4	15	3.1e5	1.6e5	4.7e5	3.1e5
1e−3	15	3.3e4	1.4e4	6.6e4	3.3e4	15	3.2e5	1.7e5	4.8e5	3.2e5
1e−5	15	3.4e4	1.5e4	6.7e4	3.4e4	15	3.3e5	1.8e5	5.0e5	3.3e5
1e−8	15	3.5e4	1.6e4	7.1e4	3.5e4	15	3.4e5	1.9e5	5.2e5	3.4e5

Δf	f_{17} in 5-D, N=15, mFE=189784 #	ERT	10%	90%	RT_{succ}	f_{17} in 20-D, N=15, mFE=197594 #	ERT	10%	90%	RT_{succ}
10	15	1.4e1	2.0e0	3.6e1	1.4e1	15	6.9e2	2.5e2	1.0e3	6.9e2
1	15	1.3e3	8.0e2	2.7e3	1.3e3	15	8.5e3	2.7e3	6.3e3	8.5e3
1e−1	15	4.7e3	1.6e3	1.6e4	4.7e3	15	1.1e4	5.5e3	9.0e3	1.1e4
1e−3	15	2.0e4	3.1e3	6.7e4	2.0e4	15	3.5e4	1.1e4	4.5e4	3.5e4
1e−5	15	2.5e4	4.5e3	9.7e4	2.5e4	15	6.8e4	4.7e4	9.7e4	6.8e4
1e−8	15	3.2e4	6.8e3	1.0e5	3.2e4	15	1.0e5	5.7e4	1.1e5	1.0e5

Δf	f_{19} in 5-D, N=15, mFE=564073 #	ERT	10%	90%	RT_{succ}	f_{19} in 20-D, N=15, mFE=2.28e6 #	ERT	10%	90%	RT_{succ}
10	15	2.7e1	1.2e1	5.2e1	2.7e1	15	1.2e3	8.8e2	1.5e3	1.2e3
1	15	6.6e2	4.8e2	9.7e2	6.6e2	15	2.1e4	9.0e3	4.2e4	2.1e4
1e−1	15	2.6e4	2.5e3	1.1e5	2.6e4	11	1.0e6	3.9e4	2.5e6	2.8e5
1e−3	9	5.8e5	4.9e3	1.4e6	2.3e5	0	*47e−3*	*84e−4*	*20e−2*	2.0e6
1e−5	9	5.8e5	5.3e3	1.4e6	2.3e5	.	.	.	.	.
1e−8	9	5.8e5	6.4e3	1.4e6	2.3e5	.	.	.	.	.

Δf	f_{21} in 5-D, N=15, mFE=3.64e6 #	ERT	10%	90%	RT_{succ}	f_{21} in 20-D, N=15, mFE=3.85e6 #	ERT	10%	90%	RT_{succ}
10	15	1.9e2	2.3e1	3.8e2	1.9e2	15	6.0e3	2.1e3	2.1e4	6.0e3
1	15	7.9e4	5.8e2	3.8e5	7.9e4	11	1.7e6	2.4e3	5.4e6	3.5e5
1e−1	15	4.7e5	7.2e2	1.5e6	4.7e5	8	4.0e6	2.7e3	9.8e6	7.6e5
1e−3	13	8.3e5	1.3e3	3.6e6	2.7e5	8	4.0e6	4.2e3	1.1e7	7.7e5
1e−5	13	8.4e5	1.5e3	3.6e6	2.8e5	8	4.0e6	5.4e3	1.1e7	7.9e5
1e−8	13	9.9e5	2.1e3	3.6e6	4.4e5	8	4.1e6	7.5e3	1.1e7	8.1e5

Δf	f_{23} in 5-D, N=15, mFE=584896 #	ERT	10%	90%	RT_{succ}	f_{23} in 20-D, N=15, mFE=2.11e6 #	ERT	10%	90%	RT_{succ}
10	15	6.7e0	2.0e0	1.4e1	6.7e0	15	6.9e0	1.0e0	1.5e1	6.9e0
1	15	8.5e3	2.4e3	1.6e4	8.5e3	15	9.1e4	2.1e4	1.9e5	9.1e4
1e−1	15	5.8e4	9.5e3	1.1e5	5.8e4	13	6.0e5	2.5e4	2.0e6	2.9e5
1e−3	15	1.2e5	1.1e4	3.1e5	1.2e5	0	*29e−3*	*50e−4*	*25e−2*	7.1e5
1e−5	14	1.3e5	1.2e4	3.2e5	9.0e4	.	.	.	.	.
1e−8	14	1.3e5	1.3e4	3.2e5	9.3e4	.	.	.	.	.

Δf	f_2 in 5-D, N=15, mFE=7050 #	ERT	10%	90%	RT_{succ}	f_2 in 20-D, N=15, mFE=45943 #	ERT	10%	90%	RT_{succ}
10	15	1.9e3	1.7e3	2.4e3	1.9e3	15	1.7e4	1.0e4	2.9e4	1.7e4
1	15	2.5e3	2.4e3	3.0e3	2.5e3	15	1.9e4	1.3e4	3.3e4	1.9e4
1e−1	15	3.1e3	2.4e3	3.4e3	3.1e3	15	2.2e4	1.5e4	3.6e4	2.2e4
1e−3	15	4.1e3	3.9e3	4.7e3	4.1e3	15	2.6e4	1.9e4	4.1e4	2.6e4
1e−5	15	5.0e3	4.6e3	5.6e3	5.0e3	15	2.9e4	2.2e4	4.2e4	2.9e4
1e−8	15	6.4e3	5.5e3	6.9e3	6.4e3	15	3.4e4	2.8e4	4.4e4	3.4e4

Δf	f_4 in 5-D, N=15, mFE=22862 #	ERT	10%	90%	RT_{succ}	f_4 in 20-D, N=15, mFE=184687 #	ERT	10%	90%	RT_{succ}
10	15	1.3e3	8.7e2	1.7e3	1.3e3	15	4.9e4	1.4e4	6.3e4	4.9e4
1	15	5.8e3	3.1e3	1.4e4	5.8e3	15	9.4e4	5.8e4	1.3e5	9.4e4
1e−1	15	9.4e3	3.4e3	1.6e4	9.4e3	15	1.0e5	6.2e4	1.4e5	1.0e5
1e−3	15	1.1e4	4.8e3	1.8e4	1.1e4	15	1.1e5	6.8e4	1.5e5	1.1e5
1e−5	15	1.2e4	6.3e3	2.0e4	1.2e4	15	1.2e5	7.4e4	1.6e5	1.2e5
1e−8	15	1.4e4	7.7e3	2.2e4	1.4e4	15	1.4e5	8.5e4	1.8e5	1.4e5

Δf	f_6 in 5-D, N=15, mFE=7366 #	ERT	10%	90%	RT_{succ}	f_6 in 20-D, N=15, mFE=29420 #	ERT	10%	90%	RT_{succ}
10	15	9.1e2	5.0e2	1.5e3	9.1e2	15	7.2e3	5.5e3	8.4e3	7.2e3
1	15	1.8e3	1.5e3	2.2e3	1.8e3	15	9.1e3	8.2e3	1.1e4	9.1e3
1e−1	15	2.5e3	2.1e3	2.9e3	2.5e3	15	1.1e4	1.1e4	1.4e4	1.1e4
1e−3	15	3.7e3	3.2e3	4.0e3	3.7e3	15	1.6e4	1.5e4	1.7e4	1.6e4
1e−5	15	4.9e3	4.5e3	5.3e3	4.9e3	15	2.0e4	1.9e4	2.1e4	2.0e4
1e−8	15	6.8e3	6.2e3	7.3e3	6.8e3	15	2.6e4	2.4e4	2.7e4	2.6e4

Δf	f_8 in 5-D, N=15, mFE=7515 #	ERT	10%	90%	RT_{succ}	f_8 in 20-D, N=15, mFE=122595 #	ERT	10%	90%	RT_{succ}
10	15	8.0e2	7.0e2	1.0e3	8.0e2	15	2.3e4	1.8e4	3.0e4	2.3e4
1	15	2.9e3	2.1e3	3.6e3	2.9e3	15	4.7e4	3.3e4	8.8e4	4.7e4
1e−1	15	4.1e3	3.4e3	5.0e3	4.1e3	15	5.1e4	3.8e4	9.6e4	5.1e4
1e−3	15	5.1e3	4.3e3	6.0e3	5.1e3	15	5.4e4	4.1e4	1.0e5	5.4e4
1e−5	15	5.7e3	5.0e3	6.6e3	5.7e3	15	5.6e4	4.2e4	1.0e5	5.6e4
1e−8	15	6.4e3	5.7e3	7.5e3	6.4e3	15	5.8e4	4.4e4	1.1e5	5.8e4

Δf	f_{10} in 5-D, N=15, mFE=13486 #	ERT	10%	90%	RT_{succ}	f_{10} in 20-D, N=15, mFE=40013 #	ERT	10%	90%	RT_{succ}
10	15	3.4e3	2.3e3	3.6e3	3.4e3	15	2.5e4	2.1e4	2.8e4	2.5e4
1	15	4.1e3	2.9e3	4.0e3	4.1e3	15	2.9e4	2.6e4	3.2e4	2.9e4
1e−1	15	4.4e3	3.0e3	4.3e3	4.4e3	15	3.2e4	3.1e4	3.3e4	3.2e4
1e−3	15	4.8e3	3.7e3	4.6e3	4.8e3	15	3.5e4	3.3e4	3.6e4	3.5e4
1e−5	15	5.3e3	4.4e3	5.1e3	5.3e3	15	3.7e4	3.6e4	3.8e4	3.7e4
1e−8	15	6.0e3	5.0e3	5.9e3	6.0e3	15	3.8e4	3.8e4	3.9e4	3.8e4

Δf	f_{12} in 5-D, N=15, mFE=20426 #	ERT	10%	90%	RT_{succ}	f_{12} in 20-D, N=15, mFE=213163 #	ERT	10%	90%	RT_{succ}
10	15	2.7e3	1.8e3	4.2e3	2.7e3	15	7.4e3	6.0e3	8.0e3	7.4e3
1	15	4.3e3	2.2e3	8.8e3	4.3e3	15	1.7e4	7.8e3	3.0e4	1.7e4
1e−1	15	5.8e3	2.7e3	1.2e4	5.8e3	15	3.7e4	8.6e3	4.1e4	3.7e4
1e−3	15	7.4e3	4.5e3	1.4e4	7.4e3	15	5.0e4	2.9e4	5.2e4	5.0e4
1e−5	15	9.0e3	5.6e3	1.7e4	9.0e3	15	5.9e4	3.7e4	6.0e4	5.9e4
1e−8	15	1.1e4	6.2e3	1.9e4	1.1e4	15	6.7e4	4.5e4	6.8e4	6.7e4

Δf	f_{14} in 5-D, N=15, mFE=6517 #	ERT	10%	90%	RT_{succ}	f_{14} in 20-D, N=15, mFE=44700 #	ERT	10%	90%	RT_{succ}
10	15	2.9e1	1.0e0	6.3e1	2.9e1	15	1.5e3	1.0e3	1.8e3	1.5e3
1	15	5.3e2	3.5e2	7.6e2	5.3e2	15	2.5e3	2.3e3	2.6e3	2.5e3
1e−1	15	8.4e2	5.9e2	1.1e3	8.4e2	15	3.2e3	2.9e3	3.8e3	3.2e3
1e−3	15	1.9e3	1.5e3	2.3e3	1.9e3	15	1.1e4	1.1e4	1.2e4	1.1e4
1e−5	15	3.6e3	3.4e3	3.9e3	3.6e3	15	2.3e4	2.2e4	2.4e4	2.3e4
1e−8	15	5.8e3	5.4e3	6.1e3	5.8e3	15	4.3e4	4.1e4	4.4e4	4.3e4

Δf	f_{16} in 5-D, N=15, mFE=1.30e6 #	ERT	10%	90%	RT_{succ}	f_{16} in 20-D, N=15, mFE=3.01e6 #	ERT	10%	90%	RT_{succ}
10	15	2.0e2	5.6e1	3.3e2	2.0e2	15	2.4e3	2.0e3	2.8e3	2.4e3
1	15	1.8e3	6.9e2	2.2e3	1.8e3	15	7.3e3	5.2e3	6.0e3	7.3e3
1e−1	15	6.9e3	1.2e3	1.8e4	6.9e3	15	3.8e4	8.7e3	1.2e5	3.8e4
1e−3	15	1.3e4	2.1e3	2.1e4	1.3e4	12	1.0e6	7.2e4	3.2e6	3.0e5
1e−5	15	9.0e4	2.9e3	2.2e4	9.0e4	12	1.1e6	7.5e4	3.2e6	3.9e5
1e−8	15	9.4e4	3.7e3	2.3e4	9.4e4	11	1.4e6	8.0e4	3.5e6	3.3e5

Δf	f_{18} in 5-D, N=15, mFE=77728 #	ERT	10%	90%	RT_{succ}	f_{18} in 20-D, N=15, mFE=375028 #	ERT	10%	90%	RT_{succ}
10	15	5.0e2	3.2e1	8.5e2	5.0e2	15	2.5e3	2.3e3	3.0e3	2.5e3
1	15	2.5e3	1.4e3	5.4e3	2.5e3	15	6.4e3	5.1e3	1.2e4	6.4e3
1e−1	15	1.2e4	2.1e3	2.5e4	1.2e4	15	2.2e4	8.0e3	5.0e4	2.2e4
1e−3	15	1.6e4	3.8e3	4.1e4	1.6e4	15	8.5e4	4.7e4	1.7e5	8.5e4
1e−5	15	1.8e4	5.6e3	4.5e4	1.8e4	15	1.6e5	9.9e4	2.1e5	1.6e5
1e−8	15	2.6e4	7.5e3	5.4e4	2.6e4	15	1.8e5	1.1e5	2.3e5	1.8e5

Δf	f_{20} in 5-D, N=15, mFE=279314 #	ERT	10%	90%	RT_{succ}	f_{20} in 20-D, N=15, mFE=941105 #	ERT	10%	90%	RT_{succ}
10	15	1.6e2	5.5e1	2.3e2	1.6e2	15	1.7e3	1.7e3	1.8e3	1.7e3
1	15	2.3e3	1.6e3	3.5e3	2.3e3	15	4.5e4	2.0e4	6.1e4	4.5e4
1e−1	15	1.6e4	4.1e3	3.0e4	1.6e4	15	2.4e5	2.2e5	2.8e5	2.4e5
1e−3	15	3.0e4	8.4e3	4.9e4	3.0e4	15	3.2e5	2.9e5	3.5e5	3.2e5
1e−5	15	5.5e4	2.4e4	1.0e5	5.5e4	15	3.6e5	3.3e5	4.0e5	3.6e5
1e−8	15	6.5e4	2.8e4	1.1e5	6.5e4	15	4.2e5	3.5e5	4.3e5	4.2e5

Δf	f_{22} in 5-D, N=15, mFE=2.73e6 #	ERT	10%	90%	RT_{succ}	f_{22} in 20-D, N=15, mFE=3.98e6 #	ERT	10%	90%	RT_{succ}
10	15	2.6e2	5.7e1	5.3e2	2.6e2	15	2.5e5	2.1e3	2.8e5	2.5e5
1	15	9.1e3	6.1e2	1.5e4	9.1e3	7	4.4e6	2.1e4	1.1e7	2.0e5
1e−1	15	2.8e5	8.8e2	1.1e6	2.8e5	3	1.6e7	5.9e5	3.7e7	1.1e6
1e−3	15	5.6e5	1.5e3	2.4e6	5.6e5	3	1.6e7	6.0e5	3.7e7	1.1e6
1e−5	15	5.7e5	2.1e3	2.4e6	5.7e5	3	1.6e7	6.2e5	3.6e7	1.1e6
1e−8	15	5.8e5	2.8e3	2.4e6	5.8e5	3	1.6e7	6.5e5	3.8e7	1.1e6

Δf	f_{24} in 5-D, N=15, mFE=839219 #	ERT	10%	90%	RT_{succ}	f_{24} in 20-D, N=15, mFE=2.32e6 #	ERT	10%	90%	RT_{succ}
10	15	3.5e3	1.7e3	6.1e3	3.5e3	8	2.3e6	4.4e4	6.5e6	4.3e5
1	9	6.0e5	6.3e3	1.7e6	6.8e4	0	*89e−1*	*24e−1*	*23e+0*	1.0e6
1e−1	3	3.3e6	1.6e5	7.8e6	1.8e5	.	.	.	.	.
1e−3	1	1.1e7	1.0e6	2.6e7	2.5e5	.	.	.	.	.
1e−5	1	1.1e7	1.1e6	2.5e7	2.5e5	.	.	.	.	.
1e−8	1	1.1e7	1.0e6	2.6e7	2.6e5	.	.	.	.	.

Table 4: 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 1); 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 1 for the names of functions.

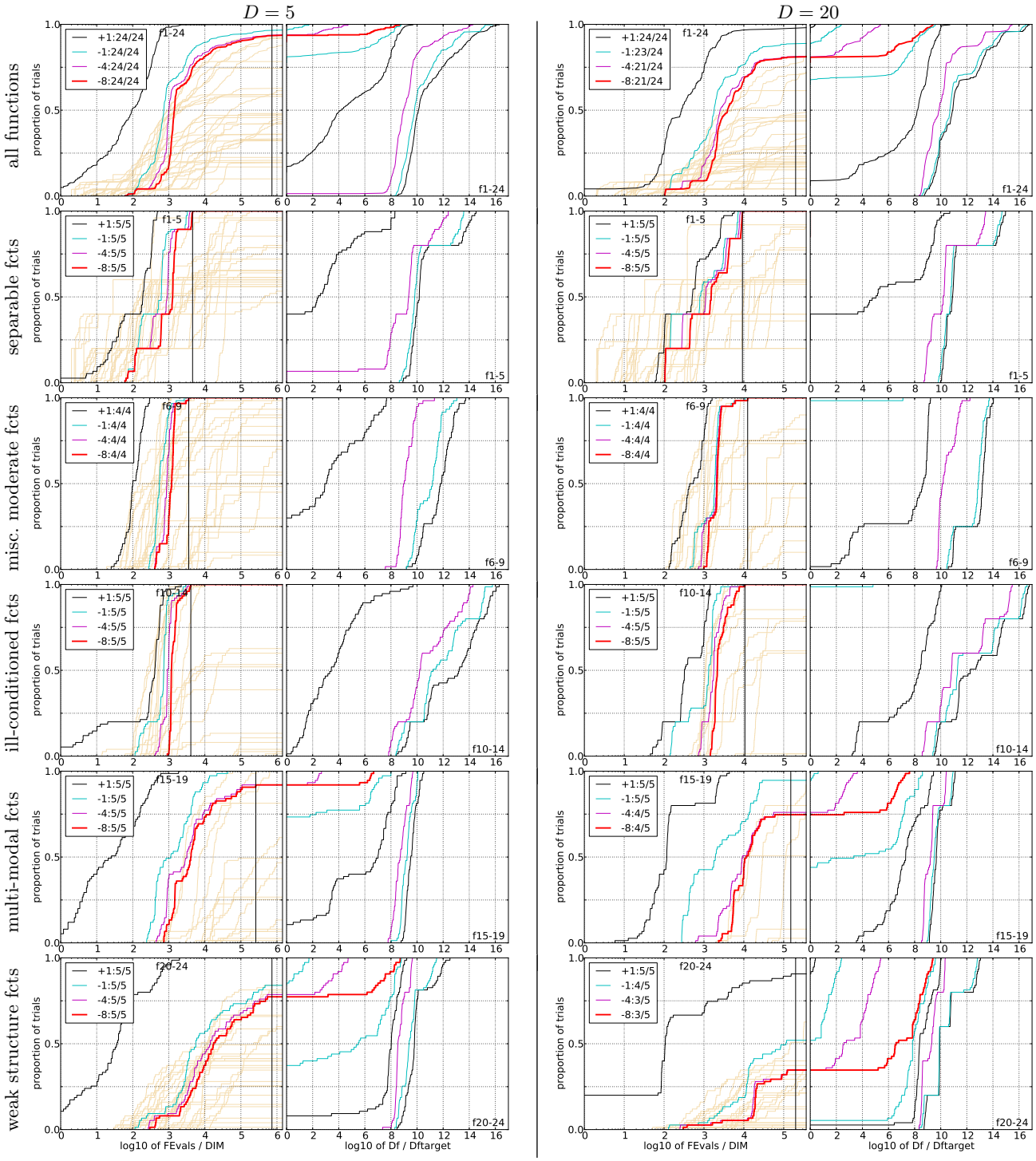

Figure 2: Empirical cumulative distribution functions (ECDFs), plotting the fraction of trials versus running time (left subplots) or versus Δf (right subplots). The thick red line represents the best achieved results. Left subplots: ECDF of the running time (number of function evaluations), divided 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.

be also useful to increase the stability in those functions in which the convergence is not always obtained.

7. ACKNOWLEDGMENTS

This work was supported by the Madrid Regional Education Ministry and the European Social Fund and financed by the Spanish Ministry of Science TIN2007-67148. The authors thankfully acknowledge the computer resources, technical expertise and assistance provided by the Centro de Supercomputación y Visualización de Madrid (CeSViMa) and the Spanish Supercomputing Network.

8. REFERENCES

[1] A. Auger and N. Hansen. A restart cma evolution strategy with increasing population size. In *Proceedings of the IEEE Congress on Evolutionary Computation, CEC 2005*, pages 1769–1776. IEEE Press, 2005.

[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. Updated February 2010.

[3] Y. Gao and Y.-J. Wang. A memetic differential evolutionary algorithm for high dimensional functions' optimization. In *Proceedings of the Third International Conference on Natural Computation (ICNC 2007)*, pages 188–192. IEEE Press, 2007.

[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] A. LaTorre. *A Framework for Hybrid Dynamic Evolutionary Algorithms: Multiple Offspring Sampling (MOS)*. PhD thesis, Universidad Politécnica de Madrid, November 2009.

[7] S. Muelas, A. LaTorre, and J. Peña. A memetic differential evolution algorithm for continuous optimization. In *Proceedings of the 9th International Conference on Intelligent Systems Design and Applications, ISDA 2009*, pages 1080–1084. IEEE Press, November 2009.

[8] R. Storn and K. Price. Differential evolution- a simple and efficient adaptive scheme for global optimization over continuous spaces. Technical report, 1995.

[9] E.-G. Talbi. A taxonomy of hybrid metaheuristics. *Journal of Heuristics*, 8(5):541–564, September 2002.