Black-Box Optimization Benchmarking for Noiseless Function Testbed using an EDA and PSO Hybrid

Mohammed El-Abd
University of Waterloo
Waterloo, Ontario, Canada
mhelabd@pami.uwaterloo.ca

Mohamed S. Kamel
University of Waterloo
Waterloo, Ontario, Canada
mkamel@pami.uwaterloo.ca

ABSTRACT

This paper benchmarks an Estimation of Distribution Algorithm (EDA) and Particle Swarm Optimizer (PSO) on noise-free BBOB 2009 testbed. The algorithm is referred to as EDA-PSO and further enhanced with correlation-triggered adaptive variance scaling.

Categories and Subject Descriptors

G.1.6 [**Numerical Analysis**]: OptimizationGlobal Optimization, Unconstrained Optimization; F.2.1 [**Analysis of Algorithms and Problem Complexity**]: Numerical Algorithms and Problems

General Terms

Algorithms

Keywords

Benchmarking, Black-box optimization, Evolutionary computation, Particle Swarm Optimization, Estimation of Distribution Algorithm, Hybrid Algorithms

1. INTRODUCTION

Particle Swarm Optimization (PSO) [1, 7] is an optimization method widely used to solve continuous nonlinear functions. It is a stochastic optimization technique that emerged from simulations of the birds flocking and fish schooling behaviors.

The algorithm used in this work hybridizes PSO and an EDA.

2. ALGORITHM PRESENTATION

A hybrid EDA-PSO approach was proposed in [8]. The algorithm works by sampling an independent univariate Gaussian distribution based on the best half of the swarm. The mean and standard deviation of the model is calculated in every iteration as:

$$\boldsymbol{\mu} = \frac{1}{M}\sum_{i=1}^{M}\mathbf{x}_i$$
$$\sigma_j = \sqrt{\frac{1}{M}\sum_{i=1}^{M}(\mathbf{x}_{ij} - \mu_j)^2}, \qquad (1)$$

where $M = N/2$ for a swarm with N particles and i is the particle number.

The choice of whether to update the particle using the normal PSO equations or to sample the particle using the estimated distribution is made with a probability p, referred to as the *participation ratio*. If $p = 0$, the algorithm will behave as a pure EDA algorithm and if $p = 1$, it will be a pure PSO algorithm. In the hybrid approach, where $0 < p < 1$, each particle is either totally updated by the PSO equations or totally sampled from the estimated distribution. Finally, the particle gets updated only if its fitness improves.

In this paper, we adaptively set the parameter p depending on the success rate of both the PSO and EDA parts in improving the particles' fitness using the *All historical information*, where the success rates are calculated based on the information gathered during the entire search:

$$p^{t+1} = \frac{\sum_{i=1}^{t}\frac{num\_PSO^i}{tot\_PSO^i}}{\sum_{i=1}^{t}\frac{num\_PSO^i}{tot\_PSO^i} + \sum_{i=1}^{t}\frac{num\_EDA^i}{tot\_EDA^i}} \qquad (2)$$

In all the previous equation num_PSO^t and tot_PSO^t refers to the number of improvements done by the PSO component at iteration t and the total number of times PSO was executed. While num_EDA^t and tot_EDA^t refers to the number of improvements done by the EDA component at iteration t and the total number of times EDA was executed

Fig. 1 shows the MATLAB code for EDA-PSO.

The performance of EDA-PSO is also enhanced by incorporating the method used in [4] for updating the variance of the Gaussian model. It was shown in [2], that this approach produces the best results when incorporated into EDA-PSO. In this method, the variance of the Gaussian model is either enlarged or reduced based on the area covered by the model. If the model is following a slope, the variance of the Gaussian model is adjusted according to the performance of the algorithm. If the Gaussian model is covering an optimum, the variance is kept as is. Wether the Gaussian model is covering a slope or an optimum is determined by calculating the correlation between the ranks of the fitness values and densities of the sampled individuals. If the correlation

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

```
function PSOEDA(FUN, DIM, ftarget, maxfunevals)

% Set algorithm parameters
popsize = 40;
c1 = 2;
c2 = 2;
xbound = 5;
vbound = 5;
p = 0.5;
num_PSO = 0;
tot_PSO = 0;
num_EDA = 0;
tot_EDA = 0;
C_AVS = 1;
C_AVS_Max = 10;
C_AVS_Min = 0.1;
E_Dec = 0.9;
E_Inc = 1 / E_Dec;

% Allocate memory and initialize
xmin = -xbound * ones(1,DIM);
xmax = xbound * ones(1,DIM);
vmin = -vbound * ones(1,DIM);
vmax = vbound * ones(1,DIM);

x = 2 * xbound * rand(popsize,DIM) - xbound;
v = 2 * vbound * rand(popsize,DIM) - vbound;
pbest = x;

% update pbest and gbest
cost_x = feval(FUN, x');
cost_p = cost_x;
[cost,index] = min(cost_p);
gbest = pbest(index,:);

maxfunevals = min(1e5 * DIM, maxfunevals);
maxiterations = ceil(maxfunevals/popsize);

for iter = 2 : maxiterations
    % PSO part
    % Update inertia weight
    w = 0.9 - 0.8*(iter-2)/(maxiterations-2);

    prev_num_EDA = num_EDA;

    % Update velocity
    candidate_v = w*v + c1*rand(popsize,DIM).*(pbest-x) +
c2*rand(popsize,DIM).* (repmat(gbest,popsize,1)-x);

    % Clamp veloctiy
    s = candidate_v < repmat(vmin,popsize,1);
    candidate_v = (1-s).*candidate_v + s.*repmat(vmin,popsize,1);
    b = candidate_v > repmat(vmax,popsize,1);
    candidate_v = (1-b).*candidate_v + b.*repmat(vmax,popsize,1);



    % Update position
    candidate_x = x + candidate_v;

    % Clamp position - Absorbing boundaries
    % Set candidate x to the boundary
    s = candidate_x < repmat(xmin,popsize,1);
    candidate_x = (1-s).*candidate_x + s.*repmat(xmin,popsize,1);
    b = candidate_x > repmat(xmax,popsize,1);
    candidate_x = (1-b).*candidate_x + b.*repmat(xmax,popsize,1);

    % Clamp position - Absorbing boundaries
    % Set candidate v to zero
    b = s | b;
    candidate_v = (1-b).*candidate_v + b.*zeros(popsize,DIM);

    % EDA part
    % Calculate Gaussian model where Mus and Sigma
    % are based on the best half of the swarm
    [cost_p_sorted,indices] = sort(cost_p);
    Mue = mean(pbest(indices(1:popsize/2),:));
    Sigma = std(pbest(indices(1:popsize/2),:));

    % Calculate correlation
    D = pbest(indices(1:popsize/2),:) - repmat(Mue,popsize/2,1);
    Distances = sum(abs(D'));
    Fitness = cost_p_sorted(1:popsize/2);
    Rho = corr(Distances', Fitness', 'type', 'spearman');
    if(Rho>-0.55)
        Sigma = sqrt(C_AVS) * Sigma;
    end

    % Generate candidate solution
    candidate_EDA_x = repmat(Mue,popsize,1) +
repmat(Sigma,popsize,1).*randn(popsize,DIM);

    % Depending on which component to choose
    % select candidates for consideration and
    % update the equivelant counters
    r = rand(popsize,1)<p;
    R = repmat(r,1,DIM);
    candidates = (1-R).*candidate_EDA_x + R.*candidate_x;
    candidates_fitness = feval(FUN, candidates');
    tot_PSO = tot_PSO + sum(r);
    tot_EDA = tot_EDA + popsize - sum(r);

    % Update x if candidates are better
    c = candidates_fitness<cost_x;
    C = repmat(c',1,DIM);
    x = (1-C).*x + C.*candidates;
    v = (1-(R&C)).*v + (R&C).*candidate_v;
    cost_x = (1-c).*cost_x + c.*candidates_fitness;

    % Update success counters

    % Update success counters
    num_PSO = num_PSO + sum(r&c');
    num_EDA = num_EDA + sum((1-r)&c');

    % Update pbest if necessary
    c = cost_x<cost_p;
    C = repmat(c',1,DIM);
    pbest = (1-C).*pbest + C.*x;
    cost_p = (1-c).*cost_p + c.*cost_x;

    % Update gbest if necessary
    [cost,index] = min(cost_p);
    gbest = pbest(index,:);

    % Update C_AVS based on the
    % EDA component performance
    if(num_EDA>prev_num_EDA)
        C_AVS = E_Inc * C_AVS;
    else
        C_AVS = E_Dec * C_AVS;
    end

      if((C_AVS<C_AVS_Min)||(C_AVS>C_AVS_Max))
            C_AVS = C_AVS_Max;
    end

    % Update probability using
    % percentage of improvements
    PSO_Imp_Perc = num_PSO / tot_PSO;
    EDA_Imp_Perc = num_EDA / tot_EDA;
    p = PSO_Imp_Perc/(PSO_Imp_Perc+EDA_Imp_Perc);

    % Exit if target is reached
    if feval(FUN, 'fbest') < ftarget
      break;
    end
end
```

Figure 1: EDA-PSO MATLAB-code.

$r > -0.55$, the adaptive scaling is used (the model is covering a slope). Otherwise, the variance is kept the same (the model is covering an optimum).

In the case of the model covering a slope, the adjustment of the variance is done by scaling the variance with a coefficient C^{AVS}, which has an initial value of 1. This coefficient is adjusted according to the performance of the algorithm. The coefficient is increased (i.e. multiplied by E_{Inc}) if the best individual has improved from the previous iteration or decreased (i.e. multiplied by E_{Dec}) otherwise. If the value for C^{AVS} drops below C^{AVS}_{Min} or higher than C^{AVS}_{Max}, it is reset to C^{AVS}_{Max} to encourage exploration. All these values are the same as used in [4]. This approach is applied after calculating μ and σ of the Gaussian model and before continuing with the update of the different particles. The only modification is that C^{AVS} is adjusted if the EDA component of the algorithm is successful in improving any particle (not necessarily the best particle).

3. RESULTS

The parameters are set as $c1 = c2 = 2$, w linearly decreases from 0.9 to 0.1 with the number of iterations, 40 particles are used, C^{AVS} is initialized to 1, $E_{Inc} = 1.1$, $E_{Dec} = 0.9$, $C^{AVS}_{Min} = 0.1$ and $C^{AVS}_{Max} = 10$.

The simulations for 2; 3; 5; 10 and 20 D were done with the MATLAB-code and took 15 hours and 45 minutes. No parameter tuning was done and the crafting effort CrE [6] is computed to zero.

Results from experiments according to [5] on the benchmark functions given in [3, 6] are presented in Figures 2 and 3 and in Table 1.

4. CPU TIMING EXPERIMENT

For the timing experiment PSO_Bounds was run with a maximum of 10^4 function evaluations and restarted until 30 seconds has passed (according to Figure 2 in [6]). The experiments have been conducted with an Intel Core 2 Quad 2.4 GHz under Windows XP using the MATLAB-code provided. The time per function evaluation was 2.0; 2.1; 2.4; 3.3; 3.8 times 10^{-5} seconds in dimensions 2; 3; 5; 10; 20 respectively.

5. REFERENCES

[1] R. C. Eberhart and J. Kennedy. A new optimizer using particle swarm theory. In *Proc. of the 6th International Symposium on Micro Machine and Human Science*, pages 39–43, 1995.

[2] M. El-Abd and M. S. Kamel. Preventing premature convergence in a pso and eda hybrid. In *Accepted IEEE Congress on Evolutionary Computation*, 2009.

[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.

[4] J. Grahl, P. A. N. Bosman, and F. Rothlauf. The correlation-triggered adaptive variance scaling idea. In *GECCO '06: Proceedings of the 8th annual conference on Genetic and evolutionary computation*, pages 397–404, 2006.

[5] N. Hansen, A. Auger, S. Finck, and R. Ros. Real-parameter black-box optimization benchmarking 2009: Experimental setup. Technical Report RR-6828, INRIA, 2009.

[6] 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.

[7] J. Kennedy and R. C. Eberhart. Particle swarm optimization. In *Proc. of IEEE International Conference on Neural Networks*, volume 4, pages 1942–1948, 1995.

[8] Y. Zhou and J. Jin. Eda-pso - a new hybrid intelligent optimization algorithm. In *Proc. of the Michigan University Graduate Student Symposium*, 2006.

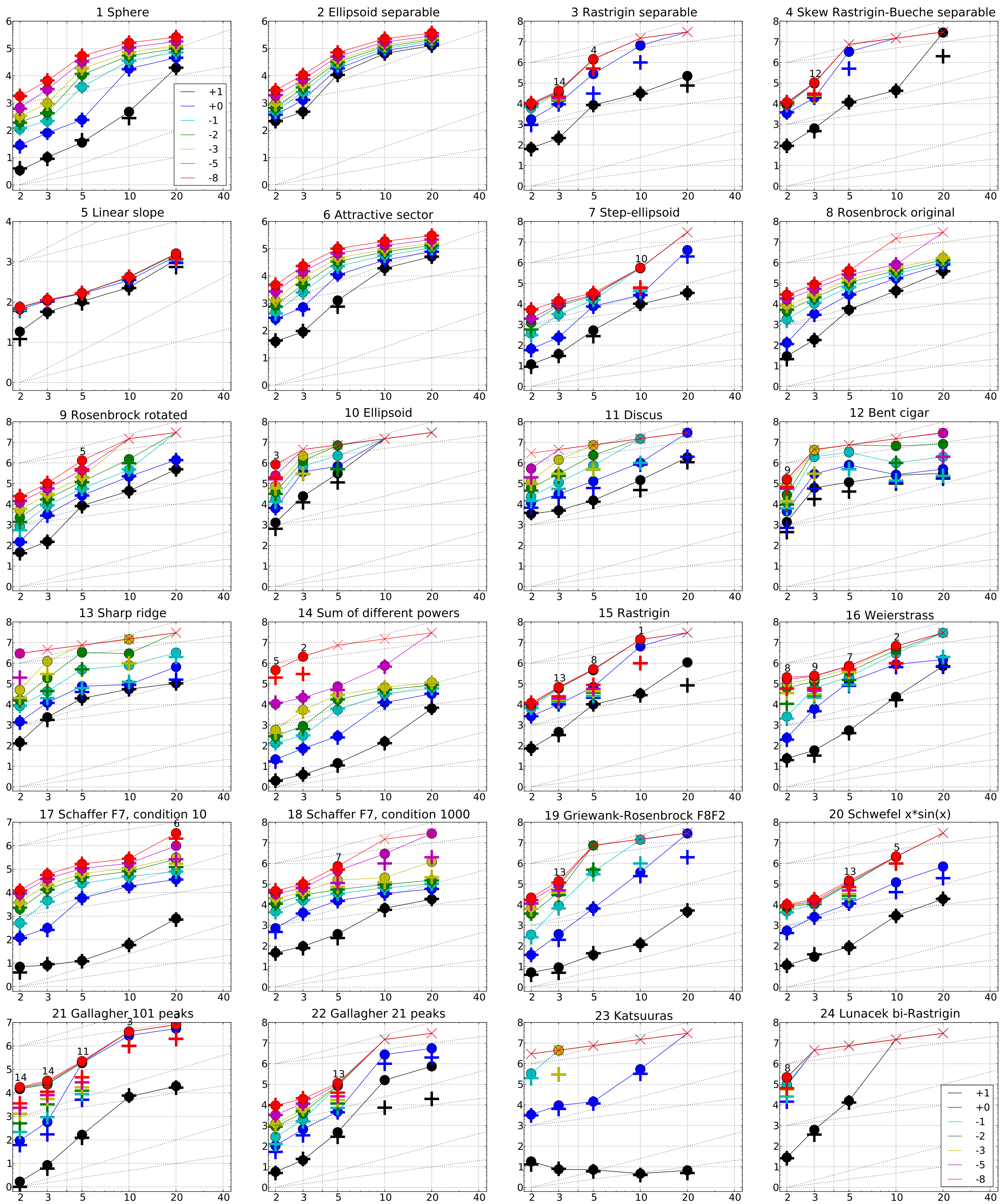

Figure 2: Expected Running Time (ERT, ●) to reach $f_{\rm opt} + \Delta f$ and median number of function evaluations of successful trials (+), shown for $\Delta f = 10, 1, 10^{-1}, 10^{-2}, 10^{-3}, 10^{-5}, 10^{-8}$ (the exponent is given in the legend of f_1 and f_{24}) versus dimension in log-log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

Δf	f_1 in 5-D, N=15, mFE=60160					f_1 in 20-D, N=15, mFE=271240				
	#	ERT	10%	90%	$\mathrm{RT_{succ}}$	#	ERT	10%	90%	$\mathrm{RT_{succ}}$
10	15	3.6e1	2.9e1	4.3e1	3.6e1	15	2.0e4	1.9e4	2.0e4	2.0e4
1	15	2.5e2	2.2e2	2.7e2	2.5e2	15	4.6e4	4.5e4	4.7e4	4.6e4
1e−1	15	3.9e3	3.3e3	4.4e3	3.9e3	15	7.2e4	7.1e4	7.3e4	7.2e4
1e−3	15	1.8e4	1.7e4	1.9e4	1.8e4	15	1.3e5	1.2e5	1.3e5	1.3e5
1e−5	15	3.3e4	3.2e4	3.4e4	3.3e4	15	1.8e5	1.8e5	1.8e5	1.8e5
1e−8	15	5.3e4	5.2e4	5.5e4	5.3e4	15	2.6e5	2.6e5	2.6e5	2.6e5

Δf	f_2 in 5-D, N=15, mFE=74280					f_2 in 20-D, N=15, mFE=378280				
	#	ERT	10%	90%	$\mathrm{RT_{succ}}$	#	ERT	10%	90%	$\mathrm{RT_{succ}}$
10	15	1.1e4	1.1e4	1.2e4	1.1e4	15	1.3e5	1.3e5	1.3e5	1.3e5
1	15	1.8e4	1.7e4	1.9e4	1.8e4	15	1.5e5	1.5e5	1.6e5	1.5e5
1e−1	15	2.5e4	2.5e4	2.6e4	2.5e4	15	1.8e5	1.8e5	1.8e5	1.8e5
1e−3	15	3.8e4	3.7e4	3.9e4	3.8e4	15	2.4e5	2.3e5	2.4e5	2.4e5
1e−5	15	5.1e4	5.0e4	5.2e4	5.1e4	15	2.9e5	2.9e5	2.9e5	2.9e5
1e−8	15	7.1e4	7.0e4	7.1e4	7.1e4	15	3.7e5	3.7e5	3.7e5	3.7e5

Δf	f_3 in 5-D, N=15, mFE=500000					f_3 in 20-D, N=15, mFE=2.00e6				
	#	ERT	10%	90%	$\mathrm{RT_{succ}}$	#	ERT	10%	90%	$\mathrm{RT_{succ}}$
10	15	8.6e3	7.6e3	9.5e3	8.6e3	14	2.2e5	7.9e4	3.9e5	2.2e5
1	10	2.8e5	1.7e5	4.1e5	2.2e5	0	*70e−1*	*50e−1*	*99e−1*	3.2e5
1e−1	4	1.4e6	9.3e5	2.8e6	5.0e5	.	.	.	.	.
1e−3	4	1.4e6	9.4e5	2.8e6	5.0e5	.	.	.	.	.
1e−5	4	1.4e6	9.5e5	2.9e6	5.0e5	.	.	.	.	.
1e−8	4	1.4e6	9.6e5	2.9e6	5.0e5	.	.	.	.	.

Δf	f_4 in 5-D, N=15, mFE=500000					f_4 in 20-D, N=15, mFE=2.00e6				
	#	ERT	10%	90%	$\mathrm{RT_{succ}}$	#	ERT	10%	90%	$\mathrm{RT_{succ}}$
10	15	1.2e4	1.1e4	1.3e4	1.2e4	1	2.8e7	1.3e7	>3e7	1.2e5
1	2	3.3e6	1.5e6	>7e6	2.7e5	0	*15e+0*	*12e+0*	*19e+0*	2.8e5
1e−1	0	*20e−1*	*99e−2*	*30e−1*	7.9e4	.	.	.	.	.
1e−3	.	.	.	.	.	.	.	.	.	.
1e−5	.	.	.	.	.	.	.	.	.	.
1e−8	.	.	.	.	.	.	.	.	.	.

Δf	f_5 in 5-D, N=15, mFE=240					f_5 in 20-D, N=15, mFE=6960				
	#	ERT	10%	90%	$\mathrm{RT_{succ}}$	#	ERT	10%	90%	$\mathrm{RT_{succ}}$
10	15	1.0e2	9.6e1	1.1e2	1.0e2	15	1.1e3	7.1e2	1.5e3	1.1e3
1	15	1.6e2	1.5e2	1.7e2	1.6e2	15	1.4e3	9.6e2	1.8e3	1.4e3
1e−1	15	1.7e2	1.5e2	1.8e2	1.7e2	15	1.5e3	1.1e3	2.0e3	1.5e3
1e−3	15	1.7e2	1.5e2	1.8e2	1.7e2	15	1.6e3	1.2e3	2.1e3	1.6e3
1e−5	15	1.7e2	1.5e2	1.8e2	1.7e2	15	1.6e3	1.2e3	2.1e3	1.6e3
1e−8	15	1.7e2	1.5e2	1.8e2	1.7e2	15	1.6e3	1.2e3	2.1e3	1.6e3

Δf	f_6 in 5-D, N=15, mFE=105120					f_6 in 20-D, N=15, mFE=328040				
	#	ERT	10%	90%	$\mathrm{RT_{succ}}$	#	ERT	10%	90%	$\mathrm{RT_{succ}}$
10	15	1.3e3	9.0e2	1.6e3	1.3e3	15	5.1e4	4.9e4	5.4e4	5.1e4
1	15	1.1e4	1.0e4	1.2e4	1.1e4	15	7.9e4	7.7e4	8.2e4	7.9e4
1e−1	15	2.3e4	2.2e4	2.4e4	2.3e4	15	1.1e5	1.0e5	1.1e5	1.1e5
1e−3	15	4.6e4	4.5e4	4.7e4	4.6e4	15	1.6e5	1.6e5	1.6e5	1.6e5
1e−5	15	6.8e4	6.7e4	6.8e4	6.8e4	15	2.2e5	2.1e5	2.2e5	2.2e5
1e−8	15	1.0e5	1.0e5	1.0e5	1.0e5	15	3.0e5	3.0e5	3.1e5	3.0e5

Δf	f_7 in 5-D, N=15, mFE=36920					f_7 in 20-D, N=15, mFE=2.00e6				
	#	ERT	10%	90%	$\mathrm{RT_{succ}}$	#	ERT	10%	90%	$\mathrm{RT_{succ}}$
10	15	5.2e2	3.0e2	7.8e2	5.2e2	15	3.5e4	3.4e4	3.6e4	3.5e4
1	15	7.8e3	7.1e3	8.4e3	7.8e3	5	4.1e6	2.7e6	7.4e6	1.6e6
1e−1	15	1.6e4	1.5e4	1.6e4	1.6e4	0	*15e−1*	*52e−2*	*47e−1*	8.9e4
1e−3	15	2.8e4	2.7e4	2.9e4	2.8e4	.	.	.	.	.
1e−5	15	2.8e4	2.7e4	2.9e4	2.8e4	.	.	.	.	.
1e−8	15	3.4e4	3.3e4	3.4e4	3.4e4	.	.	.	.	.

Δf	f_8 in 5-D, N=15, mFE=421960					f_8 in 20-D, N=15, mFE=2.00e6				
	#	ERT	10%	90%	$\mathrm{RT_{succ}}$	#	ERT	10%	90%	$\mathrm{RT_{succ}}$
10	15	5.2e3	4.2e3	6.3e3	5.2e3	15	3.8e5	3.8e5	3.9e5	3.8e5
1	15	2.8e4	2.7e4	2.9e4	2.8e4	15	8.2e5	8.1e5	8.3e5	8.2e5
1e−1	15	6.6e4	6.4e4	6.8e4	6.6e4	15	1.1e6	1.1e6	1.1e6	1.1e6
1e−3	15	1.6e5	1.6e5	1.6e5	1.6e5	15	1.7e6	1.7e6	1.7e6	1.7e6
1e−5	15	2.6e5	2.5e5	2.6e5	2.6e5	0	*17e−5*	*11e−5*	*27e−5*	2.0e6
1e−8	15	4.0e5	4.0e5	4.1e5	4.0e5	.	.	.	.	.

Δf	f_9 in 5-D, N=15, mFE=500000					f_9 in 20-D, N=15, mFE=2.00e6				
	#	ERT	10%	90%	$\mathrm{RT_{succ}}$	#	ERT	10%	90%	$\mathrm{RT_{succ}}$
10	15	8.5e3	7.5e3	9.5e3	8.5e3	15	4.9e5	4.8e5	4.9e5	4.9e5
1	15	2.6e4	2.5e4	2.7e4	2.6e4	15	1.4e6	1.4e6	1.4e6	1.4e6
1e−1	15	5.7e4	5.3e4	6.2e4	5.7e4	0	*14e−2*	*10e−2*	*23e−2*	2.0e6
1e−3	15	2.1e5	1.8e5	2.4e5	2.1e5	.	.	.	.	.
1e−5	12	4.5e5	3.8e5	5.4e5	3.7e5	.	.	.	.	.
1e−8	5	1.3e6	8.9e5	2.3e6	4.4e5	.	.	.	.	.

Δf	f_{10} in 5-D, N=15, mFE=500000					f_{10} in 20-D, N=15, mFE=2.00e6				
	#	ERT	10%	90%	$\mathrm{RT_{succ}}$	#	ERT	10%	90%	$\mathrm{RT_{succ}}$
10	10	3.1e5	2.0e5	4.5e5	2.2e5	0	*36e+1*	*19e+1*	*78e+1*	2.0e6
1	7	7.0e5	4.8e5	1.1e6	3.3e5	.	.	.	.	.
1e−1	3	2.3e6	1.4e6	6.8e6	4.8e5	.	.	.	.	.
1e−3	0	*21e−1*	*18e−3*	*23e+0*	4.5e5	.	.	.	.	.
1e−5	.	.	.	.	.	.	.	.	.	.
1e−8	.	.	.	.	.	.	.	.	.	.

Δf	f_{11} in 5-D, N=15, mFE=500000					f_{11} in 20-D, N=15, mFE=2.00e6				
	#	ERT	10%	90%	$\mathrm{RT_{succ}}$	#	ERT	10%	90%	$\mathrm{RT_{succ}}$
10	15	1.5e4	1.2e4	1.8e4	1.5e4	10	1.9e6	1.4e6	2.6e6	1.2e6
1	14	1.3e5	8.0e4	1.8e5	1.3e5	1	2.9e7	1.4e7	>3e7	2.0e6
1e−1	8	7.4e5	5.4e5	1.1e6	3.8e5	0	*79e−1*	*15e−1*	*16e+0*	2.0e6
1e−3	1	7.3e6	3.6e6	>7e6	5.0e5	.	.	.	.	.
1e−5	0	*70e−3*	*47e−4*	*86e−2*	4.5e5	.	.	.	.	.
1e−8	.	.	.	.	.	.	.	.	.	.

Δf	f_{12} in 5-D, N=15, mFE=500000					f_{12} in 20-D, N=15, mFE=2.00e6				
	#	ERT	10%	90%	$\mathrm{RT_{succ}}$	#	ERT	10%	90%	$\mathrm{RT_{succ}}$
10	13	1.2e5	5.3e4	1.8e5	1.1e5	14	3.1e5	1.7e5	4.8e5	3.0e5
1	6	8.0e5	4.8e5	1.6e6	1.2e5	13	5.1e5	2.7e5	7.6e5	4.8e5
1e−1	2	3.3e6	1.7e6	>7e6	2.8e5	8	2.0e6	1.4e6	2.8e6	1.3e6
1e−3	0	*21e−1*	*81e−3*	*12e+0*	4.5e5	1	2.8e7	1.3e7	>3e7	2.0e6
1e−5	.	.	.	.	.	1	2.8e7	1.3e7	>3e7	2.0e6
1e−8	.	.	.	.	.	0	*57e−3*	*21e−4*	*14e−1*	2.0e6

Δf	f_{13} in 5-D, N=15, mFE=500000					f_{13} in 20-D, N=15, mFE=2.00e6				
	#	ERT	10%	90%	$\mathrm{RT_{succ}}$	#	ERT	10%	90%	$\mathrm{RT_{succ}}$
10	15	2.0e4	1.9e4	2.1e4	2.0e4	15	1.1e5	1.0e5	1.1e5	1.1e5
1	14	7.6e4	4.1e4	1.2e5	7.3e4	12	6.6e5	3.2e5	1.1e6	3.1e5
1e−1	8	5.0e5	2.8e5	8.8e5	1.1e5	6	3.2e6	2.2e6	5.3e6	1.4e6
1e−3	0	*64e−3*	*65e−4*	*66e−2*	2.2e5	0	*12e−2*	*34e−3*	*12e−1*	6.3e5
1e−5	.	.	.	.	.	.	.	.	.	.
1e−8	.	.	.	.	.	.	.	.	.	.

Δf	f_{14} in 5-D, N=15, mFE=500000					f_{14} in 20-D, N=15, mFE=2.00e6				
	#	ERT	10%	90%	$\mathrm{RT_{succ}}$	#	ERT	10%	90%	$\mathrm{RT_{succ}}$
10	15	1.4e1	1.0e1	1.8e1	1.4e1	15	6.4e3	5.8e3	6.9e3	6.4e3
1	15	2.9e2	2.5e2	3.3e2	2.9e2	15	3.3e4	3.2e4	3.4e4	3.3e4
1e−1	15	5.6e3	4.6e3	6.6e3	5.6e3	15	6.1e4	6.0e4	6.2e4	6.1e4
1e−3	15	2.9e4	2.8e4	3.0e4	2.9e4	15	1.2e5	1.1e5	1.2e5	1.2e5
1e−5	15	7.4e4	6.0e4	9.1e4	7.4e4	0	*21e−6*	*16e−6*	*32e−6*	2.0e6
1e−8	0	*22e−7*	*77e−8*	*43e−7*	4.5e5	.	.	.	.	.

Δf	f_{15} in 5-D, N=15, mFE=500000					f_{15} in 20-D, N=15, mFE=2.00e6				
	#	ERT	10%	90%	$\mathrm{RT_{succ}}$	#	ERT	10%	90%	$\mathrm{RT_{succ}}$
10	15	1.0e4	9.4e3	1.1e4	1.0e4	10	1.1e6	6.8e5	1.5e6	1.0e6
1	14	7.0e4	3.2e4	1.1e5	6.8e4	0	*80e−1*	*60e−1*	*11e+0*	2.8e5
1e−1	8	4.6e5	2.8e5	7.3e5	2.6e5	.	.	.	.	.
1e−3	8	4.7e5	3.0e5	7.1e5	2.7e5	.	.	.	.	.
1e−5	8	4.9e5	3.2e5	7.5e5	2.8e5	.	.	.	.	.
1e−8	8	5.2e5	3.5e5	8.1e5	2.9e5	.	.	.	.	.

Δf	f_{16} in 5-D, N=15, mFE=500000					f_{16} in 20-D, N=15, mFE=2.00e6				
	#	ERT	10%	90%	$\mathrm{RT_{succ}}$	#	ERT	10%	90%	$\mathrm{RT_{succ}}$
10	15	5.5e2	4.0e2	7.2e2	5.5e2	15	7.3e5	5.3e5	9.2e5	7.3e5
1	15	1.3e5	8.2e4	1.7e5	1.3e5	10	1.5e6	1.0e6	2.2e6	9.6e5
1e−1	13	2.2e5	1.4e5	3.0e5	1.9e5	1	3.0e7	1.5e7	>3e7	2.0e6
1e−3	8	5.5e5	3.6e5	8.8e5	2.6e5	0	*75e−2*	*20e−2*	*72e−1*	1.6e6
1e−5	7	7.1e5	4.9e5	1.2e6	3.3e5	.	.	.	.	.
1e−8	7	7.4e5	5.1e5	1.2e6	3.4e5	.	.	.	.	.

Δf	f_{17} in 5-D, N=15, mFE=175520					f_{17} in 20-D, N=15, mFE=2.00e6				
	#	ERT	10%	90%	$\mathrm{RT_{succ}}$	#	ERT	10%	90%	$\mathrm{RT_{succ}}$
10	15	1.3e1	9.2e0	1.7e1	1.3e1	15	7.9e2	6.4e2	9.4e2	7.9e2
1	15	5.8e3	5.2e3	6.5e3	5.8e3	15	3.7e4	3.5e4	3.8e4	3.7e4
1e−1	15	2.5e4	2.4e4	2.6e4	2.5e4	15	8.1e4	7.8e4	8.4e4	8.1e4
1e−3	15	6.4e4	6.3e4	6.6e4	6.4e4	14	3.1e5	1.7e5	4.8e5	3.0e5
1e−5	15	1.1e5	1.1e5	1.1e5	1.1e5	11	9.9e5	6.2e5	1.4e6	7.4e5
1e−8	15	1.7e5	1.7e5	1.7e5	1.7e5	6	3.4e6	2.2e6	5.9e6	1.2e6

Δf	f_{18} in 5-D, N=15, mFE=500000					f_{18} in 20-D, N=15, mFE=2.00e6				
	#	ERT	10%	90%	$\mathrm{RT_{succ}}$	#	ERT	10%	90%	$\mathrm{RT_{succ}}$
10	15	3.7e2	2.3e2	5.3e2	3.7e2	15	1.8e4	1.8e4	1.9e4	1.8e4
1	15	1.6e4	1.5e4	1.6e4	1.6e4	15	5.8e4	5.6e4	5.9e4	5.8e4
1e−1	15	3.5e4	3.4e4	3.6e4	3.5e4	15	1.0e5	1.0e5	1.0e5	1.0e5
1e−3	13	1.6e5	9.2e4	2.3e5	1.1e5	10	1.2e6	7.4e5	1.9e6	7.5e5
1e−5	9	4.4e5	2.9e5	7.2e5	2.0e5	1	2.8e7	1.3e7	>3e7	2.0e6
1e−8	7	7.4e5	5.0e5	1.2e6	3.1e5	0	*47e−5*	*27e−6*	*53e−4*	1.4e6

Δf	f_{19} in 5-D, N=15, mFE=500000					f_{19} in 20-D, N=15, mFE=2.00e6				
	#	ERT	10%	90%	$\mathrm{RT_{succ}}$	#	ERT	10%	90%	$\mathrm{RT_{succ}}$
10	15	3.7e1	2.9e1	4.5e1	3.7e1	15	4.6e3	3.8e3	5.5e3	4.6e3
1	15	6.7e3	5.7e3	7.7e3	6.7e3	1	2.8e7	1.3e7	>3e7	4.7e5
1e−1	11	4.0e5	2.9e5	5.4e5	2.7e5	0	*26e−1*	*21e−1*	*31e−1*	1.6e6
1e−3	0	*66e−3*	*22e−3*	*12e−2*	4.0e5	.	.	.	.	.
1e−5	.	.	.	.	.	.	.	.	.	.
1e−8	.	.	.	.	.	.	.	.	.	.

Δf	f_{20} in 5-D, N=15, mFE=500000					f_{20} in 20-D, N=15, mFE=2.00e6				
	#	ERT	10%	90%	$\mathrm{RT_{succ}}$	#	ERT	10%	90%	$\mathrm{RT_{succ}}$
10	15	9.1e1	7.4e1	1.1e2	9.1e1	15	1.9e4	1.8e4	2.0e4	1.9e4
1	15	1.1e4	1.0e4	1.3e4	1.1e4	12	7.1e5	4.2e5	1.0e6	6.8e5
1e−1	13	9.6e4	2.3e4	1.7e5	5.6e4	0	*63e−2*	*29e−2*	*11e−1*	3.5e5
1e−3	13	1.1e5	3.9e4	1.9e5	6.9e4	.	.	.	.	.
1e−5	13	1.2e5	5.5e4	2.1e5	8.2e4	.	.	.	.	.
1e−8	13	1.5e5	8.2e4	2.3e5	1.0e5	.	.	.	.	.

Δf	f_{21} in 5-D, N=15, mFE=500000					f_{21} in 20-D, N=15, mFE=2.00e6				
	#	ERT	10%	90%	$\mathrm{RT_{succ}}$	#	ERT	10%	90%	$\mathrm{RT_{succ}}$
10	15	1.6e2	1.0e2	2.3e2	1.6e2	15	2.0e4	1.8e4	2.1e4	2.0e4
1	11	1.9e5	9.6e4	2.8e5	1.8e5	4	5.5e6	2.9e6	1.3e7	5.3e5
1e−1	11	1.9e5	1.0e5	2.8e5	1.9e5	3	8.0e6	4.1e6	2.6e7	6.9e5
1e−3	11	2.0e5	1.1e5	2.9e5	1.9e5	3	8.1e6	4.1e6	2.6e7	7.1e5
1e−5	11	2.1e5	1.2e5	3.0e5	2.0e5	3	8.1e6	4.5e6	2.6e7	7.2e5
1e−8	11	2.3e5	1.4e5	3.1e5	2.1e5	3	8.1e6	4.1e6	2.6e7	7.5e5

Δf	f_{22} in 5-D, N=15, mFE=500000					f_{22} in 20-D, N=15, mFE=2.00e6				
	#	ERT	10%	90%	$\mathrm{RT_{succ}}$	#	ERT	10%	90%	$\mathrm{RT_{succ}}$
10	15	4.7e2	2.9e2	6.7e2	4.7e2	11	7.5e5	3.3e5	1.3e6	3.8e5
1	15	4.7e3	4.0e3	5.5e3	4.7e3	4	5.6e6	3.7e6	1.1e7	2.0e6
1e−1	13	8.3e4	8.4e3	1.5e5	8.3e4	0	*26e−1*	*69e−2*	*15e+0*	2.5e5
1e−3	13	9.3e4	2.2e4	1.6e5	9.1e4	.	.	.	.	.
1e−5	13	1.0e5	3.3e4	1.6e5	9.9e4	.	.	.	.	.
1e−8	13	1.2e5	5.2e4	1.8e5	1.1e5	.	.	.	.	.

Δf	f_{23} in 5-D, N=15, mFE=500000					f_{23} in 20-D, N=15, mFE=2.00e6				
	#	ERT	10%	90%	$\mathrm{RT_{succ}}$	#	ERT	10%	90%	$\mathrm{RT_{succ}}$
10	15	7.2e0	5.6e0	8.9e0	7.2e0	15	6.7e0	5.1e0	8.4e0	6.7e0
1	15	1.5e4	1.1e4	1.8e4	1.5e4	0	*16e−1*	*14e−1*	*17e−1*	7.1e5
1e−1	0	*59e−2*	*47e−2*	*63e−2*	2.5e5	.	.	.	.	.
1e−3	.	.	.	.	.	.	.	.	.	.
1e−5	.	.	.	.	.	.	.	.	.	.
1e−8	.	.	.	.	.	.	.	.	.	.

Δf	f_{24} in 5-D, N=15, mFE=500000					f_{24} in 20-D, N=15, mFE=2.00e6				
	#	ERT	10%	90%	$\mathrm{RT_{succ}}$	#	ERT	10%	90%	$\mathrm{RT_{succ}}$
10	15	1.6e4	1.4e4	1.8e4	1.6e4	0	*86e+0*	*81e+0*	*91e+0*	7.9e5
1	0	*61e−1*	*56e−1*	*64e−1*	4.0e5	.	.	.	.	.
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 $f_{\rm opt}+\Delta f$ (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 ($\mathrm{RT_{succ}}$). If $f_{\rm opt}+\Delta f$ was never reached, figures in *italics* denote the best achieved Δf-value of the median trial and the 10% and 90%-tile trial. Furthermore, N denotes the number of trials, and mFE denotes the maximum of number of function evaluations executed in one trial. See Figure 2 for the names of functions.

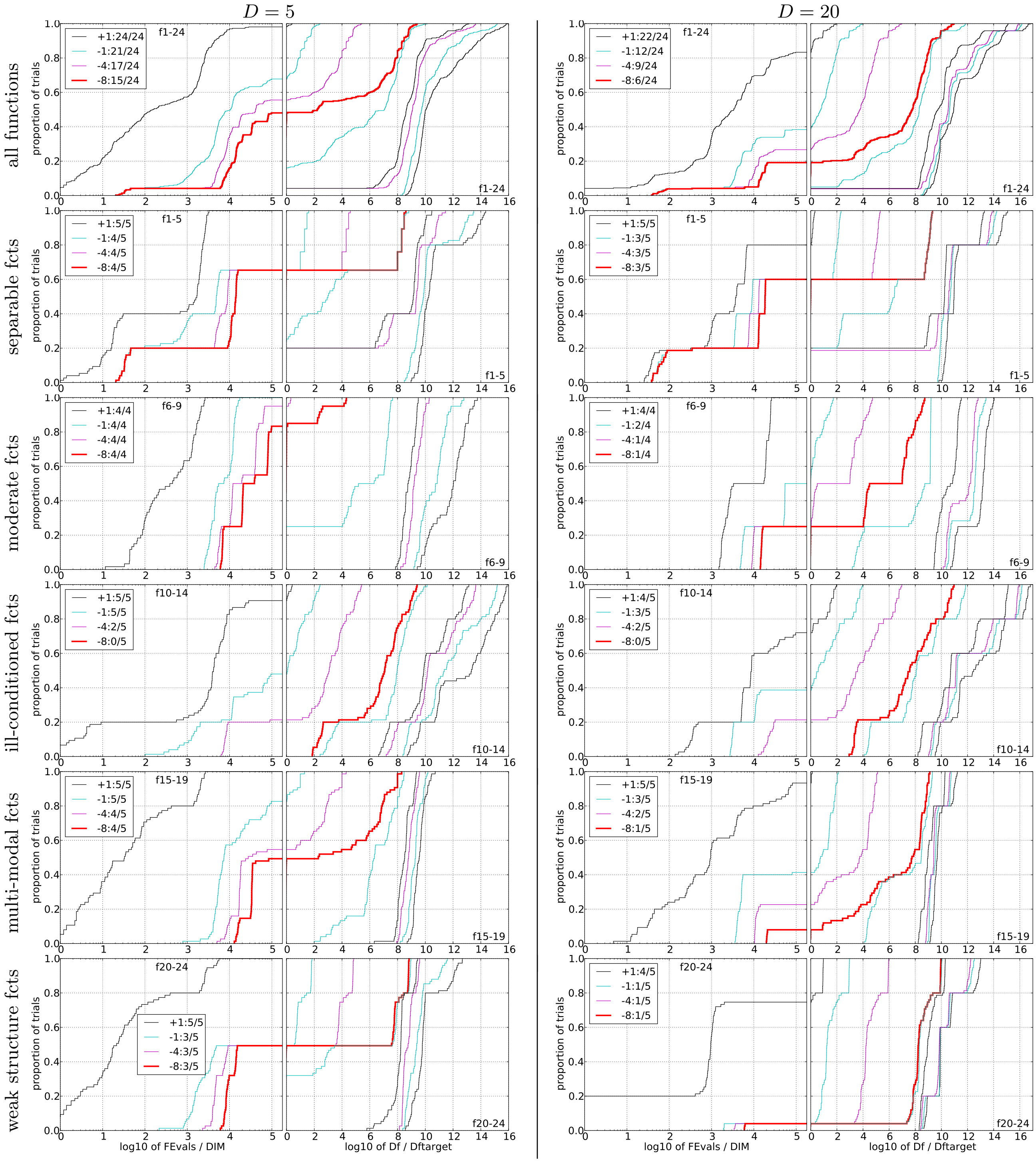

Figure 3: Empirical cumulative distribution functions (ECDFs), plotting the fraction of trials versus running time (left) or Δf. Left subplots: ECDF of the running time (number of function evaluations), divided by search space dimension D, to fall below $f_{\mathrm{opt}} + \Delta f$ with $\Delta f =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