Benchmarking the Nelder-Mead Downhill Simplex Algorithm With Many Local Restarts

Nikolaus Hansen

Benchmarking the Nelder-Mead Downhill Simplex Algorithm With Many Local Restarts

Nikolaus Hansen
Microsoft Research–INRIA Joint Centre
28 rue Jean Rostand
91893 Orsay Cedex, France
Nikolaus.Hansen@inria.fr

ABSTRACT

We benchmark the Nelder-Mead downhill simplex method on the noisefree BBOB-2009 testbed. A multistart strategy is applied on two levels. On a local level, at least ten restarts are conducted with a small number of iterations and reshaped simplex. On the global level independent restarts are launched until 10^5D function evaluations are exceeded, for dimension $D \geq 20$ ten times less. For low search space dimensions the algorithm shows very good results on many functions. It solves 24, 18, 11 and 7 of 24 functions in 2, 5, 10 and 40-D.

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, Evolutionary computation, Simplex downhill

1. INTRODUCTION

The Nelder-Mead method [5] is a real-parameter black-box optimization method that operates, similar to many evolutionary algorithms, on a set of solution points using only the *ranking* of solution. The latter implies that the algorithm is invariant under order-preserving transformations of the objective function values. The Nelder-Mead algorithm is independent of the choice of coordinate system and therefore exhibits more attractive invariance properties. In contrast to most evolutionary algorithms, the Nelder-Mead algorithm does not solely resort to selection for improving the average solution and it does not contain stochastic elements.

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

2. THE ALGORITHM

The Nelder-Mead method [5] operates on a set of $D+1$ solution points, a simplex, where D is the search space dimension. Generally, a new solution is constructed by reflecting the worst solution on the center of the remaining D solutions. Depending on the quality of the new solution additional operations are performed, but details are omitted here. For solving global optimization problems sophisticated restart procedures have been proposed for example in [4]. In this paper, a few different restart mechanism are used.

1. local restarts, where the recent best solution is used as initial solution for the restart and the projection of the range of the recent simplex is used for initialization of the new simplex. The default initialization procedure is applied, which places new simplex points by changing one coordinate at a time. At least ten restarts are conducted and the maximum number of iterations is $200 \times \sqrt{D}$ for all but the last. This number is as small that even on the sphere function f_1 usually some local restarts are conducted.
2. local restarts as above, where some additional perturbation is added to the simplex.
3. global restarts, which are completely independent of previous results.

The applied reshaping exploits the given coordinate system and improves the local search abitities for $D \geq 10$ on functions with comparatively low parameter dependencies. It is also effective on the Rosenbrock function in moderate dimension. All details are given in Figure 1 and in the next sections.

3. PARAMETER TUNING

Exemplary online experiments on f_2 and f_8 have been conducted to verify reasonable constants for the maximum iteration number of local restarts (constant 200 in Figure 1) and the number of local restarts (constants 10 and 0.1). The chosen dependencies on D have not been verified. We added add-hoc termination criteria, where `TolX` turned out to be useful. At most between 10^5D and 2×10^5D function evaluations are conducted (input parameter `maxfunevals` was set to 10^5D), for $D \geq 20$ ten times less. The final parameter setting were identical on all functions and therefore the crafting effort [2] is $\mathrm{CrE} = 0$.

Figure 1: Multistart procedure of Nelder-Mead in Matlab

```
function [x, ilaunch, f] = MY_OPTIMIZER(FUN, DIM, ftarget, maxfunevals)
% minimizes FUN in DIM dimensions by multistarts of fminsearch.
% ftarget and maxfunevals are additional external termination conditions,
% where at most 2 * maxfunevals function evaluations are conducted.
% fminsearch was modified to take as input variable usual_delta to
% generate the first simplex.

  % set options, make sure we always terminate
  % with restarts up to 2*maxfunevals are allowed
  options = optimset('MaxFunEvals', min(1e9*DIM, maxfunevals), ...
                     'MaxIter', 2e3*DIM, ... % overwritten later
                     'Tolfun', 1e-11, ...
                     'TolX', 1e-11, ...
                     'OutputFcn', @callback, ...
                     'Display', 'off');

  ilocal = 0;

  % multistart such that ftarget is reached with reasonable prob.
  for ilaunch = 1:1e5;  % relaunch optimizer up to 1e5 times
    % set initial conditions
    ilocal = ilocal + 1;
    if ilocal == 1  % (re-)start from scratch
      xstart = 8 * rand(DIM, 1) - 4;  % random start solution
      usual_delta = 2;
      options = optimset(options, 'MaxIter', floor(200*sqrt(DIM)));
    else  % refining restart run
      xstart = x;  % try to improve found solution
      usual_delta = 10 * range(v,2);  % a bit of regularization in given coordinate sys
      if rand(1,1) < 0.2  %  a bit of desperatation
        usual_delta = usual_delta + (1/ilocal) * (0.1/ilocal).^rand(DIM,1);
      end
      if rand(1,1) < 0.1 * (ilocal-10)/sqrt(DIM)  % final run
        options = optimset(options, 'MaxIter', 500*DIM); % long run
        ilocal = 0;           % real restart after this run
      end
    end

    % try fminsearch from Matlab, modified to take usual_delta as arg
    [x,f,e,o,v] = fminsearch_mod(FUN, xstart, usual_delta, options);
    % disp(sprintf('%d %d: %e  %e %e', ilocal, feval(FUN, 'evaluations'), f-ftarget, ...
    %               min(usual_delta), max(usual_delta)/min(usual_delta)));

    if feval(FUN, 'fbest') < ftarget || ...
          feval(FUN, 'evaluations') >= maxfunevals
      break;
    end
    % if useful, modify more options here for next launch
  end

  function stop = callback(x, optimValues, state)
    stop = false;
    if optimValues.fval < ftarget
      stop = true;
    end
  end

end
```

4. METHODS

We have used the matlab function `fminsearch`, Revision 1.21.4.7, and made the variable `usual_delta` an additional input parameter. Onto this algorithm we have applied a multistart strategy as given in Figure 1. This procedure has been benchmarked on the noiseless BBOB-2009 testbed [1, 3] according to the experimental design from [2].

The initial solution from which the first simplex is constructed was chosen uniformely distributed in $[4, 4]^D$ or as the former best solution.

5. CPU TIMING EXPERIMENT

For the timing experiment the same multistart algorithm was run on f_8 and restarted until at least 30 seconds had passed (according to Figure 2 in [2]). These experiments have been conducted with an Intel dual core T5600 processor with 1.8 GHz under Linux 2.6.27-11 using Matlab R2008a. The results were $6.2; 5.8; 5.6; 5.7; 5.8; 5.9$ and 6.3×10^{-4} seconds 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. RESULTS AND DISCUSSION

The results are presented in Table 1 and Figures 2 and 3. The method solves 24, 23, 18, 11, 8 and 7 out of 24 functions in 2, 3, 5, 10, 20 and 40-D (Figure 2). The expected number of function evaluations to reach a given target function value scales often quadratically with the dimension on unimodal functions and on functions 21 and 22 (Figure 2). The scaling is remarably better only on f_2 in larger dimension, presumably due to the used simplex reshaping. The scaling is often worse than quadratical, not only but in particular on multi-modal functions, and the algorithm fails within the given budget for larger dimension.

Figure 3 reveals the algorithms main weaknesses on the multimodal functions 15–19. These multimodal functions have a large number of optima and an independent multistart algorithm cannot discover the overall function structure. The performance is also poor in larger dimension in particular on the ill-conditioned functions 10–14. In contrast, the performance is very good on the low dimensional ill-conditioned functions.

7. CONCLUSION

The Nelder-Mead algorithm, as implemented in Matlab, equipped with an additional input vector and applied in a multistart fashion, is a fast and reliable black-box search algorithm for low dimensional search spaces. The applied reshaping of the simplex extends its efficiency to larger dimension only for unimodal functions with little dependencies between variables. The multiple independent restarts allow to searching unstructured multi-modal landscapes comparatively effective, while a global topography within a multimodal or rugged landscape is not well exploited.

Acknowledgments

The author would like to acknowledge the great and hard work of the BBOB team with particular kudos to Raymond Ros, Steffen Finck and Anne Auger, and Anne Auger and Marc Schoenauer for their kind and persistent support.

8. REFERENCES

[1] 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.

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

[3] 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.

[4] M. Luersen and R. Le Riche. Globalized Nelder–Mead method for engineering optimization. *Computers and Structures*, 82(23-26):2251–2260, 2004.

[5] J. Nelder and R. Mead. The downhill simplex method. *Computer Journal*, 7:308–313, 1965.

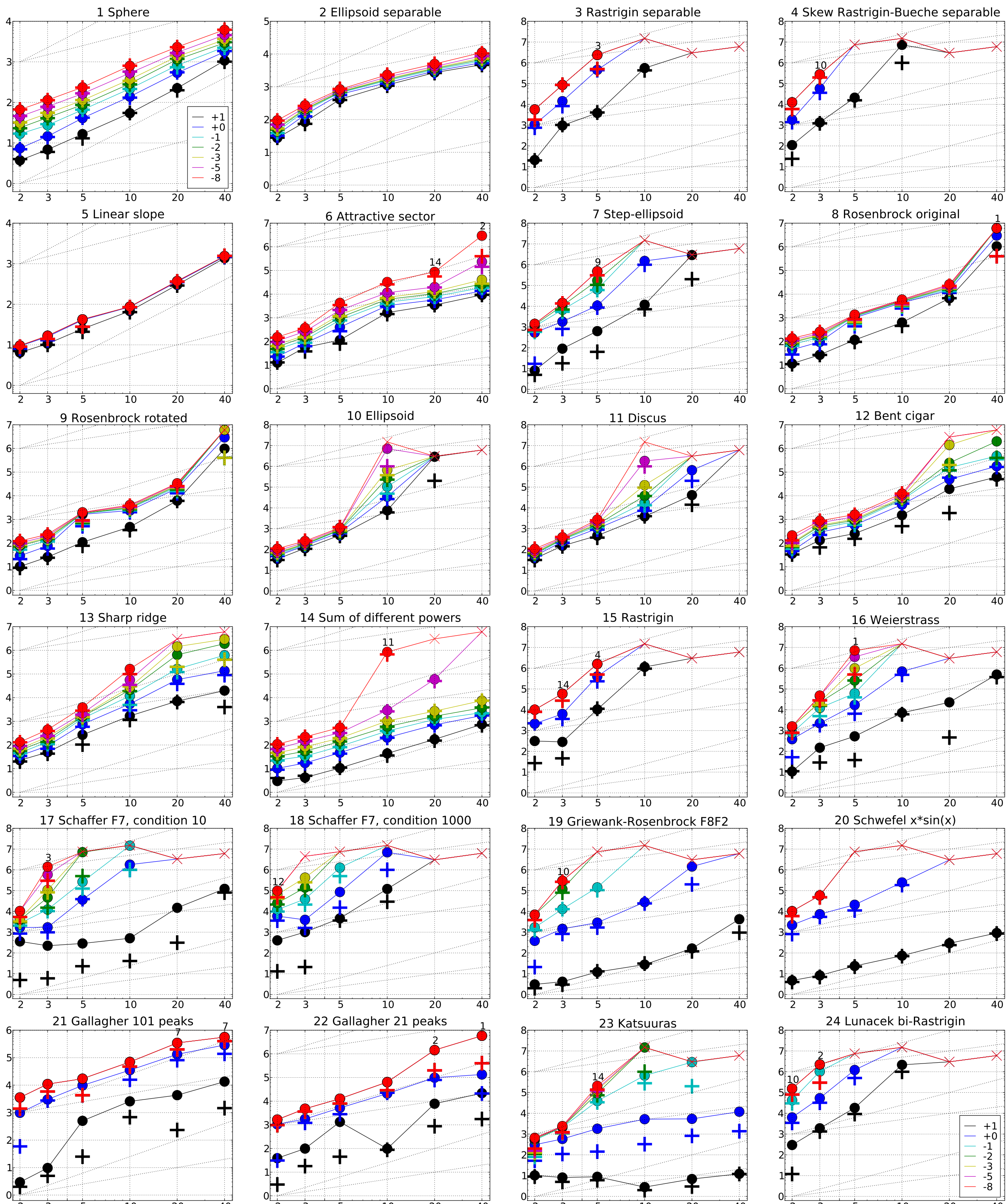

Figure 2: Expected Running Time (ERT, ●) **to reach** $f_{\mathrm{opt}} + \Delta f$ **and median number of function evaluations of successful trials** (+), **shown for** $\Delta f = 10, 1, 10^{-1}, 10^{-2}, 10^{-3}, 10^{-5}, 10^{-8}$ **(the exponent is given in the legend of** f_1 **and** f_{24}**) versus dimension in log-log presentation. The** $\mathrm{ERT}(\Delta f)$ **equals to** $\#\mathrm{FEs}(\Delta f)$ **divided by the number of successful trials, where a trial is successful if** $f_{\mathrm{opt}} + \Delta f$ **was surpassed during the trial. The** $\#\mathrm{FEs}(\Delta f)$ **are the total number of function evaluations while** $f_{\mathrm{opt}} + \Delta f$ **was not surpassed during the trial from all respective trials (successful and unsuccessful), and** f_{opt} **denotes the optimal function value. Crosses** ($\times$) **indicate the total number of function evaluations** $\#\mathrm{FEs}(-\infty)$**. 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=279					f_1 in 20-D, N=15, mFE=2932				
	#	ERT	10%	90%	$\mathrm{RT_{succ}}$	#	ERT	10%	90%	$\mathrm{RT_{succ}}$
10	15	1.7e1	1.4e1	1.9e1	1.7e1	15	2.2e2	2.0e2	2.5e2	2.2e2
1	15	4.1e1	3.6e1	4.6e1	4.1e1	15	5.4e2	4.8e2	5.9e2	5.4e2
1e−1	15	6.6e1	6.1e1	7.0e1	6.6e1	15	8.2e2	7.4e2	9.1e2	8.2e2
1e−3	15	1.1e2	1.1e2	1.2e2	1.1e2	15	1.4e3	1.3e3	1.4e3	1.4e3
1e−5	15	1.6e2	1.6e2	1.7e2	1.6e2	15	1.7e3	1.6e3	1.8e3	1.7e3
1e−8	15	2.4e2	2.3e2	2.4e2	2.4e2	15	2.3e3	2.2e3	2.4e3	2.3e3

Δf	f_2 in 5-D, N=15, mFE=1678					f_2 in 20-D, N=15, mFE=6101				
	#	ERT	10%	90%	$\mathrm{RT_{succ}}$	#	ERT	10%	90%	$\mathrm{RT_{succ}}$
10	15	4.1e2	3.5e2	4.7e2	4.1e2	15	2.7e3	2.6e3	2.8e3	2.7e3
1	15	5.9e2	5.1e2	6.8e2	5.9e2	15	3.0e3	2.9e3	3.1e3	3.0e3
1e−1	15	6.6e2	5.8e2	7.4e2	6.6e2	15	3.3e3	3.2e3	3.4e3	3.3e3
1e−3	15	7.1e2	6.4e2	7.9e2	7.1e2	15	3.8e3	3.7e3	3.9e3	3.8e3
1e−5	15	7.6e2	6.8e2	8.5e2	7.6e2	15	4.1e3	4.1e3	4.2e3	4.1e3
1e−8	15	8.4e2	7.5e2	9.3e2	8.4e2	15	5.0e3	4.8e3	5.1e3	5.0e3

Δf	f_3 in 5-D, N=15, mFE=500412					f_3 in 20-D, N=15, mFE=201261				
	#	ERT	10%	90%	$\mathrm{RT_{succ}}$	#	ERT	10%	90%	$\mathrm{RT_{succ}}$
10	15	3.9e3	2.6e3	5.1e3	3.9e3	0	*81e+0*	*67e+0*	*97e+0*	6.3e4
1	11	4.6e5	3.5e5	6.1e5	3.3e5	.	.	.	.	.
1e−1	3	2.4e6	1.4e6	7.2e6	5.0e5	.	.	.	.	.
1e−3	3	2.4e6	1.4e6	7.1e6	5.0e5	.	.	.	.	.
1e−5	3	2.4e6	1.4e6	7.2e6	5.0e5	.	.	.	.	.
1e−8	3	2.4e6	1.4e6	7.2e6	5.0e5	.	.	.	.	.

Δf	f_4 in 5-D, N=15, mFE=500615					f_4 in 20-D, N=15, mFE=201200				
	#	ERT	10%	90%	$\mathrm{RT_{succ}}$	#	ERT	10%	90%	$\mathrm{RT_{succ}}$
10	15	2.1e4	1.6e4	2.6e4	2.1e4	0	*13e+1*	*11e+1*	*18e+1*	1.3e5
1	0	*30e−1*	*20e−1*	*40e−1*	1.6e5	.	.	.	.	.
1e−1	.	.	.	.	.	.	.	.	.	.
1e−3	.	.	.	.	.	.	.	.	.	.
1e−5	.	.	.	.	.	.	.	.	.	.
1e−8	.	.	.	.	.	.	.	.	.	.

Δf	f_5 in 5-D, N=15, mFE=131					f_5 in 20-D, N=15, mFE=629				
	#	ERT	10%	90%	$\mathrm{RT_{succ}}$	#	ERT	10%	90%	$\mathrm{RT_{succ}}$
10	15	2.5e1	2.2e1	2.8e1	2.5e1	15	3.0e2	2.8e2	3.3e2	3.0e2
1	15	4.1e1	3.1e1	5.2e1	4.1e1	15	3.6e2	3.3e2	3.9e2	3.6e2
1e−1	15	4.2e1	3.1e1	5.3e1	4.2e1	15	3.7e2	3.4e2	4.0e2	3.7e2
1e−3	15	4.2e1	3.3e1	5.4e1	4.2e1	15	3.7e2	3.5e2	4.0e2	3.7e2
1e−5	15	4.2e1	3.2e1	5.4e1	4.2e1	15	3.7e2	3.4e2	4.0e2	3.7e2
1e−8	15	4.2e1	3.2e1	5.3e1	4.2e1	15	3.7e2	3.4e2	4.0e2	3.7e2

Δf	f_6 in 5-D, N=15, mFE=9819					f_6 in 20-D, N=15, mFE=210040				
	#	ERT	10%	90%	$\mathrm{RT_{succ}}$	#	ERT	10%	90%	$\mathrm{RT_{succ}}$
10	15	1.1e2	8.1e1	1.5e2	1.1e2	15	3.5e3	3.1e3	3.8e3	3.5e3
1	15	4.0e2	3.1e2	5.0e2	4.0e2	15	5.7e3	5.2e3	6.2e3	5.7e3
1e−1	15	8.0e2	6.3e2	9.7e2	8.0e2	15	7.9e3	7.2e3	8.7e3	7.9e3
1e−3	15	1.4e3	1.2e3	1.6e3	1.4e3	15	1.3e4	1.2e4	1.4e4	1.3e4
1e−5	15	2.1e3	1.9e3	2.3e3	2.1e3	15	2.0e4	1.7e4	2.3e4	2.0e4
1e−8	15	4.3e3	3.6e3	4.9e3	4.3e3	14	8.7e4	6.6e4	1.1e5	8.1e4

Δf	f_7 in 5-D, N=15, mFE=500446					f_7 in 20-D, N=15, mFE=201231				
	#	ERT	10%	90%	$\mathrm{RT_{succ}}$	#	ERT	10%	90%	$\mathrm{RT_{succ}}$
10	15	6.4e2	3.6e2	9.1e2	6.4e2	1	2.9e6	1.4e6	>3e6	2.0e5
1	15	1.1e4	6.9e3	1.5e4	1.1e4	0	*16e+0*	*11e+0*	*26e+0*	7.1e4
1e−1	15	6.5e4	4.9e4	8.3e4	6.5e4	.	.	.	.	.
1e−3	9	4.8e5	3.5e5	6.9e5	3.2e5	.	.	.	.	.
1e−5	9	4.8e5	3.6e5	6.9e5	3.2e5	.	.	.	.	.
1e−8	9	4.8e5	3.5e5	6.8e5	3.2e5	.	.	.	.	.

Δf	f_8 in 5-D, N=15, mFE=5587					f_8 in 20-D, N=15, mFE=51755				
	#	ERT	10%	90%	$\mathrm{RT_{succ}}$	#	ERT	10%	90%	$\mathrm{RT_{succ}}$
10	15	1.2e2	9.2e1	1.4e2	1.2e2	15	6.7e3	6.0e3	7.3e3	6.7e3
1	15	1.0e3	5.8e2	1.5e3	1.0e3	15	1.5e4	1.1e4	1.8e4	1.5e4
1e−1	15	1.1e3	6.9e2	1.6e3	1.1e3	15	1.7e4	1.4e4	2.0e4	1.7e4
1e−3	15	1.2e3	7.9e2	1.7e3	1.2e3	15	2.1e4	1.7e4	2.4e4	2.1e4
1e−5	15	1.3e3	8.6e2	1.8e3	1.3e3	15	2.3e4	2.0e4	2.7e4	2.3e4
1e−8	15	1.4e3	9.2e2	1.8e3	1.4e3	15	2.6e4	2.3e4	2.9e4	2.6e4

Δf	f_9 in 5-D, N=15, mFE=7218					f_9 in 20-D, N=15, mFE=93783				
	#	ERT	10%	90%	$\mathrm{RT_{succ}}$	#	ERT	10%	90%	$\mathrm{RT_{succ}}$
10	15	1.1e2	7.9e1	1.4e2	1.1e2	15	6.2e3	5.5e3	6.8e3	6.2e3
1	15	1.6e3	8.3e2	2.4e3	1.6e3	15	2.1e4	1.4e4	2.8e4	2.1e4
1e−1	15	1.8e3	9.7e2	2.6e3	1.8e3	15	2.4e4	1.8e4	3.0e4	2.4e4
1e−3	15	1.9e3	1.1e3	2.7e3	1.9e3	15	2.9e4	2.3e4	3.5e4	2.9e4
1e−5	15	1.9e3	1.1e3	2.7e3	1.9e3	15	3.2e4	2.6e4	3.8e4	3.2e4
1e−8	15	2.0e3	1.2e3	2.8e3	2.0e3	15	3.4e4	2.8e4	4.1e4	3.4e4

Δf	f_{10} in 5-D, N=15, mFE=1999					f_{10} in 20-D, N=15, mFE=211907				
	#	ERT	10%	90%	$\mathrm{RT_{succ}}$	#	ERT	10%	90%	$\mathrm{RT_{succ}}$
10	15	4.8e2	4.1e2	5.5e2	4.8e2	1	2.9e6	1.4e6	>3e6	2.0e5
1	15	6.6e2	5.7e2	7.4e2	6.6e2	0	*30e+0*	*11e+0*	*70e+0*	1.3e5
1e−1	15	8.3e2	7.1e2	9.6e2	8.3e2	.	.	.	.	.
1e−3	15	9.2e2	8.0e2	1.0e3	9.2e2	.	.	.	.	.
1e−5	15	1.0e3	8.9e2	1.1e3	1.0e3	.	.	.	.	.
1e−8	15	1.1e3	1.0e3	1.2e3	1.1e3	.	.	.	.	.

Δf	f_{11} in 5-D, N=15, mFE=5646					f_{11} in 20-D, N=15, mFE=212233				
	#	ERT	10%	90%	$\mathrm{RT_{succ}}$	#	ERT	10%	90%	$\mathrm{RT_{succ}}$
10	15	4.6e2	3.8e2	5.5e2	4.6e2	15	4.1e4	2.5e4	5.8e4	4.1e4
1	15	1.0e3	8.8e2	1.1e3	1.0e3	4	6.5e5	3.9e5	1.4e6	1.3e5
1e−1	15	1.3e3	1.2e3	1.4e3	1.3e3	0	*16e−1*	*70e−2*	*35e−1*	8.9e4
1e−3	15	1.8e3	1.6e3	2.0e3	1.8e3	.	.	.	.	.
1e−5	15	2.2e3	2.0e3	2.4e3	2.2e3	.	.	.	.	.
1e−8	15	2.7e3	2.4e3	3.1e3	2.7e3	.	.	.	.	.

Δf	f_{12} in 5-D, N=15, mFE=4154					f_{12} in 20-D, N=15, mFE=205387				
	#	ERT	10%	90%	$\mathrm{RT_{succ}}$	#	ERT	10%	90%	$\mathrm{RT_{succ}}$
10	15	2.5e2	2.1e2	3.0e2	2.5e2	15	1.9e4	1.2e4	2.8e4	1.9e4
1	15	5.8e2	4.7e2	7.0e2	5.8e2	15	4.9e4	3.8e4	6.0e4	4.9e4
1e−1	15	8.0e2	6.3e2	9.8e2	8.0e2	11	1.6e5	1.1e5	2.1e5	1.1e5
1e−3	15	1.0e3	8.3e2	1.3e3	1.0e3	2	1.4e6	7.2e5	>3e6	2.0e5
1e−5	15	1.3e3	1.1e3	1.6e3	1.3e3	0	*54e−4*	*87e−5*	*35e−2*	1.0e5
1e−8	15	1.6e3	1.3e3	1.9e3	1.6e3	.	.	.	.	.

Δf	f_{13} in 5-D, N=15, mFE=9607					f_{13} in 20-D, N=15, mFE=208304				
	#	ERT	10%	90%	$\mathrm{RT_{succ}}$	#	ERT	10%	90%	$\mathrm{RT_{succ}}$
10	15	2.7e2	1.8e2	3.7e2	2.7e2	15	7.1e3	5.1e3	9.6e3	7.1e3
1	15	7.5e2	5.1e2	1.0e3	7.5e2	15	5.8e4	4.0e4	7.6e4	5.8e4
1e−1	15	1.3e3	1.0e3	1.6e3	1.3e3	11	1.6e5	1.2e5	2.3e5	1.1e5
1e−3	15	1.7e3	1.5e3	2.0e3	1.7e3	2	1.4e6	7.4e5	>3e6	1.9e5
1e−5	15	2.2e3	1.8e3	2.5e3	2.2e3	0	*35e−3*	*34e−5*	*62e−2*	1.3e5
1e−8	15	3.9e3	3.1e3	4.6e3	3.9e3	.	.	.	.	.

Δf	f_{14} in 5-D, N=15, mFE=680					f_{14} in 20-D, N=15, mFE=209057				
	#	ERT	10%	90%	$\mathrm{RT_{succ}}$	#	ERT	10%	90%	$\mathrm{RT_{succ}}$
10	15	1.1e1	8.2e0	1.3e1	1.1e1	15	1.7e2	1.5e2	2.0e2	1.7e2
1	15	4.7e1	4.1e1	5.4e1	4.7e1	15	7.1e2	6.1e2	8.1e2	7.1e2
1e−1	15	8.8e1	8.0e1	9.6e1	8.8e1	15	1.2e3	1.1e3	1.3e3	1.2e3
1e−3	15	1.9e2	1.8e2	2.0e2	1.9e2	15	2.7e3	2.6e3	2.8e3	2.7e3
1e−5	15	3.3e2	3.1e2	3.4e2	3.3e2	15	5.9e4	4.5e4	7.7e4	5.9e4
1e−8	15	5.4e2	5.3e2	5.7e2	5.4e2	0	*44e−7*	*34e−7*	*66e−7*	1.1e5

Δf	f_{15} in 5-D, N=15, mFE=500566					f_{15} in 20-D, N=15, mFE=201195				
	#	ERT	10%	90%	$\mathrm{RT_{succ}}$	#	ERT	10%	90%	$\mathrm{RT_{succ}}$
10	15	1.0e4	7.4e3	1.3e4	1.0e4	0	*80e+0*	*58e+0*	*93e+0*	1.1e5
1	10	4.0e5	3.1e5	5.4e5	3.2e5	.	.	.	.	.
1e−1	4	1.6e6	1.0e6	3.3e6	4.5e5	.	.	.	.	.
1e−3	4	1.6e6	1.0e6	3.3e6	4.5e5	.	.	.	.	.
1e−5	4	1.6e6	1.0e6	3.3e6	4.5e5	.	.	.	.	.
1e−8	4	1.6e6	1.0e6	3.3e6	4.5e5	.	.	.	.	.

Δf	f_{16} in 5-D, N=15, mFE=500609					f_{16} in 20-D, N=15, mFE=204125				
	#	ERT	10%	90%	$\mathrm{RT_{succ}}$	#	ERT	10%	90%	$\mathrm{RT_{succ}}$
10	15	5.3e2	1.8e2	8.7e2	5.3e2	15	2.3e4	1.4e4	3.2e4	2.3e4
1	15	1.7e4	1.1e4	2.4e4	1.7e4	0	*47e−1*	*29e−1*	*66e−1*	1.4e5
1e−1	15	6.2e4	3.7e4	9.0e4	6.2e4	.	.	.	.	.
1e−3	6	9.9e5	6.6e5	1.7e6	3.6e5	.	.	.	.	.
1e−5	2	3.5e6	1.7e6	>7e6	4.1e5	.	.	.	.	.
1e−8	1	7.2e6	3.5e6	>7e6	5.0e5	.	.	.	.	.

Δf	f_{17} in 5-D, N=15, mFE=512162					f_{17} in 20-D, N=15, mFE=355148				
	#	ERT	10%	90%	$\mathrm{RT_{succ}}$	#	ERT	10%	90%	$\mathrm{RT_{succ}}$
10	15	2.9e2	2.7e1	5.5e2	2.9e2	15	1.5e4	5.5e3	2.5e4	1.5e4
1	15	3.7e4	2.5e4	4.9e4	3.7e4	0	*62e−1*	*45e−1*	*77e−1*	1.0e5
1e−1	13	2.6e5	1.9e5	3.4e5	2.3e5	.	.	.	.	.
1e−3	0	*54e−3*	*16e−3*	*11e−2*	2.8e5	.	.	.	.	.
1e−5	.	.	.	.	.	.	.	.	.	.
1e−8	.	.	.	.	.	.	.	.	.	.

Δf	f_{18} in 5-D, N=15, mFE=512528					f_{18} in 20-D, N=15, mFE=221968				
	#	ERT	10%	90%	$\mathrm{RT_{succ}}$	#	ERT	10%	90%	$\mathrm{RT_{succ}}$
10	15	4.6e3	2.9e3	6.6e3	4.6e3	0	*20e+0*	*14e+0*	*30e+0*	7.9e4
1	15	8.6e4	4.9e4	1.2e5	8.6e4	.	.	.	.	.
1e−1	5	1.3e6	8.4e5	2.3e6	4.2e5	.	.	.	.	.
1e−3	0	*17e−2*	*50e−3*	*58e−2*	2.2e5	.	.	.	.	.
1e−5	.	.	.	.	.	.	.	.	.	.
1e−8	.	.	.	.	.	.	.	.	.	.

Δf	f_{19} in 5-D, N=15, mFE=503610					f_{19} in 20-D, N=15, mFE=212448				
	#	ERT	10%	90%	$\mathrm{RT_{succ}}$	#	ERT	10%	90%	$\mathrm{RT_{succ}}$
10	15	1.2e1	1.1e1	1.4e1	1.2e1	15	1.6e2	1.3e2	2.0e2	1.6e2
1	15	2.9e3	1.7e3	4.2e3	2.9e3	2	1.4e6	7.5e5	>3e6	2.0e5
1e−1	15	1.4e5	1.1e5	1.8e5	1.4e5	0	*19e−1*	*91e−2*	*23e−1*	1.3e5
1e−3	0	*59e−3*	*24e−3*	*90e−3*	1.6e5	.	.	.	.	.
1e−5	.	.	.	.	.	.	.	.	.	.
1e−8	.	.	.	.	.	.	.	.	.	.

Δf	f_{20} in 5-D, N=15, mFE=500477					f_{20} in 20-D, N=15, mFE=201399				
	#	ERT	10%	90%	$\mathrm{RT_{succ}}$	#	ERT	10%	90%	$\mathrm{RT_{succ}}$
10	15	2.4e1	1.9e1	2.9e1	2.4e1	15	2.9e2	2.5e2	3.3e2	2.9e2
1	15	2.1e4	1.4e4	2.8e4	2.1e4	0	*13e−1*	*12e−1*	*14e−1*	1.3e5
1e−1	0	*24e−2*	*24e−2*	*47e−2*	2.5e5	.	.	.	.	.
1e−3	.	.	.	.	.	.	.	.	.	.
1e−5	.	.	.	.	.	.	.	.	.	.
1e−8	.	.	.	.	.	.	.	.	.	.

Δf	f_{21} in 5-D, N=15, mFE=66367					f_{21} in 20-D, N=15, mFE=201297				
	#	ERT	10%	90%	$\mathrm{RT_{succ}}$	#	ERT	10%	90%	$\mathrm{RT_{succ}}$
10	15	5.0e2	2.3e2	7.7e2	5.0e2	15	4.3e3	1.8e3	6.9e3	4.3e3
1	15	9.7e3	4.9e3	1.5e4	9.7e3	12	1.3e5	1.1e5	1.6e5	1.1e5
1e−1	15	1.7e4	1.0e4	2.5e4	1.7e4	7	3.4e5	2.4e5	5.5e5	1.4e5
1e−3	15	1.7e4	9.7e3	2.5e4	1.7e4	7	3.4e5	2.4e5	5.5e5	1.4e5
1e−5	15	1.7e4	9.4e3	2.5e4	1.7e4	7	3.4e5	2.4e5	5.5e5	1.5e5
1e−8	15	1.7e4	1.0e4	2.5e4	1.7e4	7	3.4e5	2.4e5	5.6e5	1.5e5

Δf	f_{22} in 5-D, N=15, mFE=84189					f_{22} in 20-D, N=15, mFE=201226				
	#	ERT	10%	90%	$\mathrm{RT_{succ}}$	#	ERT	10%	90%	$\mathrm{RT_{succ}}$
10	15	1.3e3	6.9e2	2.0e3	1.3e3	15	7.8e3	3.3e3	1.2e4	7.8e3
1	15	5.2e3	3.2e3	7.4e3	5.2e3	13	9.8e4	6.6e4	1.4e5	6.7e4
1e−1	15	1.3e4	6.5e3	1.9e4	1.3e4	2	1.4e6	7.3e5	>3e6	1.8e5
1e−3	15	1.3e4	6.5e3	2.0e4	1.3e4	2	1.4e6	7.4e5	>3e6	1.8e5
1e−5	15	1.3e4	6.6e3	2.0e4	1.3e4	2	1.4e6	7.4e5	>3e6	1.9e5
1e−8	15	1.3e4	6.6e3	2.0e4	1.3e4	2	1.4e6	7.3e5	>3e6	1.9e5

Δf	f_{23} in 5-D, N=15, mFE=500023					f_{23} in 20-D, N=15, mFE=202618				
	#	ERT	10%	90%	$\mathrm{RT_{succ}}$	#	ERT	10%	90%	$\mathrm{RT_{succ}}$
10	15	8.8e0	6.5e0	1.1e1	8.8e0	15	6.9e0	4.3e0	9.7e0	6.9e0
1	15	1.8e3	9.6e2	2.8e3	1.8e3	15	5.4e3	3.2e3	7.7e3	5.4e3
1e−1	15	3.9e4	2.7e4	5.2e4	3.9e4	1	2.9e6	1.4e6	>3e6	2.0e5
1e−3	15	1.3e5	9.9e4	1.6e5	1.3e5	0	*20e−2*	*11e−2*	*27e−2*	7.9e4
1e−5	15	1.5e5	1.1e5	1.9e5	1.5e5	.	.	.	.	.
1e−8	14	2.1e5	1.5e5	2.8e5	1.8e5	.	.	.	.	.

Δf	f_{24} in 5-D, N=15, mFE=500713					f_{24} in 20-D, N=15, mFE=201076				
	#	ERT	10%	90%	$\mathrm{RT_{succ}}$	#	ERT	10%	90%	$\mathrm{RT_{succ}}$
10	15	1.8e4	1.1e4	2.6e4	1.8e4	0	*10e+1*	*65e+0*	*14e+1*	1.1e5
1	5	1.2e6	8.3e5	2.2e6	4.5e5	.	.	.	.	.
1e−1	0	*12e−1*	*83e−2*	*30e−1*	1.4e5	.	.	.	.	.
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 ($\mathbf{RT_{succ}}$). 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.

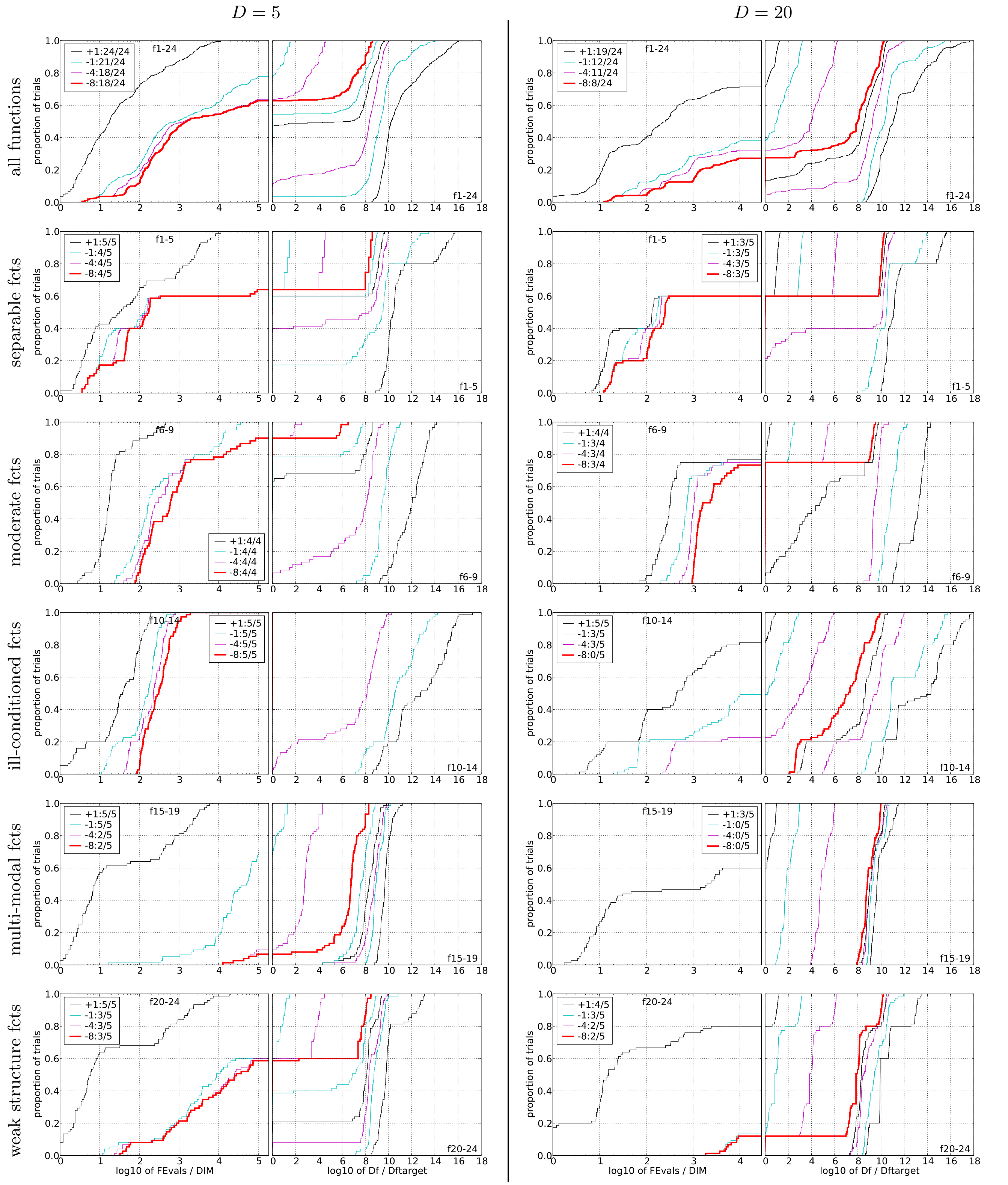

Figure 3: Empirical cumulative distribution functions (ECDFs), plotting the fraction of trials versus running time (left subplots) or versus Δf (right subplots). The thick red line represents the best achieved results. Left subplots: ECDF of the running time (number of function evaluations), divided by search space dimension D, to fall below $f_{\text{opt}} + \Delta f$ with $\Delta f = 10^k$, where k is the first value in the legend. Right subplots: ECDF of the best achieved Δf divided by 10^k (upper left lines in continuation of the left subplot), and best achieved Δf divided by 10^{-8} for running times of $D, 10\,D, 100\,D\ldots$ 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.