BBO-Benchmarking the Nelder-Mead Downhill Simplex Algorithm

An Example BBOB 2009 Workshop Paper

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

The BBOBies

ABSTRACT

As an example, we benchmark the Nelder-Mead downhill simplex method on the noisefree BBOB 2009 testbed. A multistart strategy is applied with a maximum number of function evaluations of about 2×10^4 times the search space dimension. For low search space dimensions the algorithm shows very good results.

Keywords

Benchmarking, Nelder Mead, downhill simplex, black-box optimization, evolutionary computation

1. INTRODUCTION

The Nelder-Mead method [4] 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 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.

In this paper, a multistart version of the Nelder-Mead method is benchmarked on the noiseless BBOB 2009 testbed [1, 3] according to the experimental design from [2], cf. to Figure 1.

2. 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. The Matlab implementation of the evaluated multistart procedure is given in Figure 1 revealing all details. The initial solution from which the first simplex is constructed was chosen uniformely distributed in $[4, 4]^D$ or as the former best solution. We added add-hoc termination criteria, where `TolX` turned out to be useful. At most between 10^4 and 2×10^4 function evaluations are conducted and the overall experiment took about one day for up to 20-D and another two days for 40-D using Matlab under Linux (see below). No further parameter tuning was done and the crafting effort, CrE [2], is computed to zero.

*Dr. Hansen insisted his name be first

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

3. RESULTS AND DISCUSSION

The results are presented in Table 1 and Figures 2 and 3. The method solves 24, 21, 15, 10, 8 and 3 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 usually quadratically with the dimension for moderate dimension (Figure 2), sometimes worse. For larger dimension the scaling often becomes worse or the algorithm fails within the given budget.

Figure 3 reveals the algorithms main weaknesses on the multimodal functions 15–19. These multimodal functions have a large number of optima and a simple multistart algorithm cannot discover the overall function structure. The performance is also poor in larger dimension on the ill-conditioned functions 10–14 and the weakly structured functions 20–24. In contrast, the performance is very good on the low dimensional ill-conditioned functions.

4. 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 $3.9; 3.8; 3.7; 3.9; 4.1; 4.3$ and 4.7×10^{-4} seconds per function evaluation in dimension $2; 3; 5; 10; 20; 40$ and 80, respectively. Up to 80-D the dependency of CPU time on the search space dimensionality is small.

5. CONCLUSION

The Nelder-Mead algorithm, as implemented in Matlab (Revision 1.21.4.7), equipped with an additional input parameter and applied in a multistart fashion, is a fast and re-

Δf	f_1 in 5-D, N=15, mFE=302					f_1 in 20-D, N=15, mFE=4245				
	#	ERT	10%	90%	RT_{succ}	#	ERT	10%	90%	RT_{succ}
10	15	1.8e1	1.5e1	2.1e1	1.8e1	15	2.0e2	1.7e2	2.3e2	2.0e2
1	15	4.2e1	3.8e1	4.7e1	4.2e1	15	4.0e2	3.6e2	4.5e2	4.0e2
1e−1	15	7.0e1	6.4e1	7.7e1	7.0e1	15	6.3e2	5.7e2	6.9e2	6.3e2
1e−3	15	1.2e2	1.1e2	1.3e2	1.2e2	15	1.1e3	1.0e3	1.2e3	1.1e3
1e−5	15	1.7e2	1.6e2	1.8e2	1.7e2	15	1.8e3	1.6e3	1.9e3	1.8e3
1e−8	15	2.4e2	2.3e2	2.5e2	2.4e2	15	3.1e3	2.9e3	3.4e3	3.1e3

Δf	f_2 in 5-D, N=15, mFE=1169					f_2 in 20-D, N=15, mFE=249948				
	#	ERT	10%	90%	RT_{succ}	#	ERT	10%	90%	RT_{succ}
10	15	3.5e2	2.8e2	4.3e2	3.5e2	5	5.0e5	4.4e5	5.6e5	1.7e5
1	15	5.1e2	4.4e2	5.8e2	5.1e2	3	9.5e5	8.9e5	1.0e6	2.0e5
1e−1	15	6.1e2	5.5e2	6.7e2	6.1e2	2	1.5e6	1.4e6	1.5e6	2.0e5
1e−3	15	7.0e2	6.4e2	7.6e2	7.0e2	2	1.5e6	1.5e6	1.6e6	2.0e5
1e−5	15	7.6e2	7.0e2	8.2e2	7.6e2	1	3.0e6	3.0e6	3.1e6	2.0e5
1e−8	15	8.3e2	7.7e2	9.0e2	8.3e2	1	3.0e6	3.0e6	3.1e6	2.0e5

Δf	f_3 in 5-D, N=15, mFE=50602					f_3 in 20-D, N=15, mFE=247014				
	#	ERT	10%	90%	RT_{succ}	#	ERT	10%	90%	RT_{succ}
10	15	1.8e3	1.2e3	2.4e3	1.8e3	0	*64e+0*	*44e+0*	*12e+1*	1.8e5
1	5	1.2e5	1.0e5	1.3e5	3.6e4	.	.	.	.	.
1e−1	0	*20e−1*	*99e−2*	*30e−1*	2.2e4	.	.	.	.	.
1e−3	.	.	.	.	.	.	.	.	.	.
1e−5	.	.	.	.	.	.	.	.	.	.
1e−8	.	.	.	.	.	.	.	.	.	.

Δf	f_4 in 5-D, N=15, mFE=50512					f_4 in 20-D, N=15, mFE=249949				
	#	ERT	10%	90%	RT_{succ}	#	ERT	10%	90%	RT_{succ}
10	15	9.3e3	6.9e3	1.2e4	9.3e3	0	*16e+1*	*99e+0*	*19e+1*	1.3e5
1	0	*40e−1*	*20e−1*	*60e−1*	2.2e4	.	.	.	.	.
1e−1	.	.	.	.	.	.	.	.	.	.
1e−3	.	.	.	.	.	.	.	.	.	.
1e−5	.	.	.	.	.	.	.	.	.	.
1e−8	.	.	.	.	.	.	.	.	.	.

Δf	f_5 in 5-D, N=15, mFE=39					f_5 in 20-D, N=15, mFE=752				
	#	ERT	10%	90%	RT_{succ}	#	ERT	10%	90%	RT_{succ}
10	15	1.8e1	1.6e1	1.9e1	1.8e1	15	3.0e2	2.7e2	3.2e2	3.0e2
1	15	2.5e1	2.3e1	2.7e1	2.5e1	15	3.5e2	3.1e2	3.9e2	3.5e2
1e−1	15	2.6e1	2.4e1	2.8e1	2.6e1	15	3.6e2	3.2e2	4.0e2	3.6e2
1e−3	15	2.6e1	2.4e1	2.8e1	2.6e1	15	3.6e2	3.2e2	4.1e2	3.6e2
1e−5	15	2.6e1	2.4e1	2.8e1	2.6e1	15	3.6e2	3.2e2	4.1e2	3.6e2
1e−8	15	2.6e1	2.4e1	2.8e1	2.6e1	15	3.6e2	3.2e2	4.1e2	3.6e2

Δf	f_6 in 5-D, N=15, mFE=29777					f_6 in 20-D, N=15, mFE=227236				
	#	ERT	10%	90%	RT_{succ}	#	ERT	10%	90%	RT_{succ}
10	15	1.5e2	9.3e1	2.1e2	1.5e2	10	1.8e5	1.4e5	2.1e5	1.4e5
1	15	7.3e2	5.4e2	9.4e2	7.3e2	3	9.8e5	9.1e5	1.1e6	2.2e5
1e−1	15	9.6e2	7.5e2	1.2e3	9.6e2	1	3.1e6	3.0e6	3.2e6	2.1e5
1e−3	15	1.8e3	1.6e3	2.1e3	1.8e3	0	*68e−1*	*10e−2*	*22e+0*	1.1e5
1e−5	15	2.6e3	2.2e3	2.9e3	2.6e3	.	.	.	.	.
1e−8	15	1.3e4	1.1e4	1.6e4	1.3e4	.	.	.	.	.

Δf	f_7 in 5-D, N=15, mFE=48561					f_7 in 20-D, N=15, mFE=214363				
	#	ERT	10%	90%	RT_{succ}	#	ERT	10%	90%	RT_{succ}
10	15	3.3e2	1.2e2	5.5e2	3.3e2	5	4.7e5	4.0e5	5.5e5	2.1e5
1	15	3.2e3	2.0e3	4.4e3	3.2e3	0	*12e+0*	*78e−1*	*26e+0*	8.9e4
1e−1	12	1.7e4	1.3e4	2.2e4	1.4e4	.	.	.	.	.
1e−3	2	2.6e5	2.4e5	2.9e5	3.6e4	.	.	.	.	.
1e−5	2	2.6e5	2.4e5	2.9e5	3.6e4	.	.	.	.	.
1e−8	2	2.6e5	2.4e5	2.9e5	3.6e4	.	.	.	.	.

Δf	f_8 in 5-D, N=15, mFE=3817					f_8 in 20-D, N=15, mFE=247465				
	#	ERT	10%	90%	RT_{succ}	#	ERT	10%	90%	RT_{succ}
10	15	8.2e1	7.0e1	9.5e1	8.2e1	15	3.7e4	3.1e4	4.4e4	3.7e4
1	15	6.7e2	3.8e2	9.9e2	6.7e2	12	1.6e5	1.3e5	1.9e5	1.2e5
1e−1	15	8.0e2	5.1e2	1.1e3	8.0e2	9	2.4e5	2.0e5	2.8e5	1.5e5
1e−3	15	8.9e2	6.0e2	1.2e3	8.9e2	9	2.6e5	2.2e5	2.9e5	1.6e5
1e−5	15	9.5e2	6.5e2	1.3e3	9.5e2	9	2.6e5	2.2e5	2.9e5	1.6e5
1e−8	15	1.0e3	7.3e2	1.3e3	1.0e3	9	2.6e5	2.3e5	3.0e5	1.6e5

Δf	f_9 in 5-D, N=15, mFE=2014					f_9 in 20-D, N=15, mFE=215698				
	#	ERT	10%	90%	RT_{succ}	#	ERT	10%	90%	RT_{succ}
10	15	8.8e1	7.6e1	1.0e2	8.8e1	15	3.6e4	2.7e4	4.5e4	3.6e4
1	15	5.2e2	3.8e2	6.5e2	5.2e2	14	1.0e5	8.3e4	1.2e5	8.7e4
1e−1	15	6.5e2	5.2e2	7.8e2	6.5e2	12	1.4e5	1.1e5	1.7e5	9.7e4
1e−3	15	7.4e2	6.1e2	8.8e2	7.4e2	11	1.7e5	1.4e5	2.0e5	1.1e5
1e−5	15	7.9e2	6.7e2	9.2e2	7.9e2	10	1.9e5	1.5e5	2.2e5	1.0e5
1e−8	15	8.7e2	7.4e2	1.0e3	8.7e2	10	1.9e5	1.6e5	2.2e5	1.1e5

Δf	f_{10} in 5-D, N=15, mFE=1454					f_{10} in 20-D, N=15, mFE=249952				
	#	ERT	10%	90%	RT_{succ}	#	ERT	10%	90%	RT_{succ}
10	15	5.8e2	4.8e2	6.9e2	5.8e2	0	*26e+1*	*64e+0*	*62e+1*	1.4e5
1	15	6.9e2	6.1e2	7.8e2	6.9e2	.	.	.	.	.
1e−1	15	7.3e2	6.5e2	8.2e2	7.3e2	.	.	.	.	.
1e−3	15	8.1e2	7.3e2	9.0e2	8.1e2	.	.	.	.	.
1e−5	15	8.9e2	8.2e2	9.7e2	8.9e2	.	.	.	.	.
1e−8	15	9.9e2	9.2e2	1.1e3	9.9e2	.	.	.	.	.

Δf	f_{11} in 5-D, N=15, mFE=2111					f_{11} in 20-D, N=15, mFE=237329				
	#	ERT	10%	90%	RT_{succ}	#	ERT	10%	90%	RT_{succ}
10	15	6.7e2	5.1e2	8.3e2	6.7e2	10	1.9e5	1.5e5	2.3e5	1.2e5
1	15	1.2e3	1.0e3	1.3e3	1.2e3	1	3.1e6	3.0e6	3.2e6	2.1e5
1e−1	15	1.3e3	1.2e3	1.4e3	1.3e3	0	*72e−1*	*10e−1*	*16e+0*	1.1e5
1e−3	15	1.4e3	1.3e3	1.5e3	1.4e3	.	.	.	.	.
1e−5	15	1.5e3	1.4e3	1.6e3	1.5e3	.	.	.	.	.
1e−8	15	1.6e3	1.5e3	1.7e3	1.6e3	.	.	.	.	.

Δf	f_{12} in 5-D, N=15, mFE=2182					f_{12} in 20-D, N=15, mFE=317904				
	#	ERT	10%	90%	RT_{succ}	#	ERT	10%	90%	RT_{succ}
10	15	2.2e2	1.8e2	2.5e2	2.2e2	15	9.1e3	4.8e3	1.4e4	9.1e3
1	15	4.7e2	3.8e2	5.5e2	4.7e2	15	4.3e4	2.8e4	5.8e4	4.3e4
1e−1	15	6.3e2	5.2e2	7.6e2	6.3e2	12	1.2e5	8.1e4	1.5e5	1.1e5
1e−3	15	9.2e2	8.1e2	1.0e3	9.2e2	8	2.8e5	2.2e5	3.4e5	1.4e5
1e−5	15	1.2e3	1.0e3	1.3e3	1.2e3	6	4.4e5	3.7e5	5.1e5	1.9e5
1e−8	15	1.4e3	1.2e3	1.6e3	1.4e3	3	9.8e5	8.4e5	1.1e6	1.9e5

Δf	f_{13} in 5-D, N=15, mFE=7618					f_{13} in 20-D, N=15, mFE=249444				
	#	ERT	10%	90%	RT_{succ}	#	ERT	10%	90%	RT_{succ}
10	15	2.0e2	1.3e2	2.8e2	2.0e2	15	1.0e4	5.8e3	1.5e4	1.0e4
1	15	7.4e2	5.0e2	9.9e2	7.4e2	15	6.9e4	5.5e4	8.4e4	6.9e4
1e−1	15	1.1e3	7.5e2	1.4e3	1.1e3	10	1.8e5	1.5e5	2.2e5	1.3e5
1e−3	15	1.4e3	1.1e3	1.7e3	1.4e3	4	7.6e5	6.9e5	8.2e5	2.0e5
1e−5	15	1.7e3	1.4e3	2.0e3	1.7e3	2	1.6e6	1.5e6	1.7e6	2.0e5
1e−8	15	2.9e3	2.4e3	3.4e3	2.9e3	0	*68e−4*	*52e−7*	*44e−2*	1.6e5

Δf	f_{14} in 5-D, N=15, mFE=804					f_{14} in 20-D, N=15, mFE=231808				
	#	ERT	10%	90%	RT_{succ}	#	ERT	10%	90%	RT_{succ}
10	15	8.4e0	6.2e0	1.1e1	8.4e0	15	1.5e2	1.4e2	1.7e2	1.5e2
1	15	4.3e1	4.0e1	4.6e1	4.3e1	15	5.8e2	5.2e2	6.5e2	5.8e2
1e−1	15	8.2e1	7.6e1	8.9e1	8.2e1	15	1.2e3	1.1e3	1.3e3	1.2e3
1e−3	15	2.2e2	2.1e2	2.3e2	2.2e2	15	2.2e4	1.7e4	2.8e4	2.2e4
1e−5	15	3.7e2	3.5e2	3.9e2	3.7e2	11	1.4e5	1.1e5	1.8e5	1.0e5
1e−8	15	6.0e2	5.8e2	6.3e2	6.0e2	0	*52e−7*	*72e−8*	*21e−6*	7.9e4

Δf	f_{15} in 5-D, N=15, mFE=50645					f_{15} in 20-D, N=15, mFE=241030				
	#	ERT	10%	90%	RT_{succ}	#	ERT	10%	90%	RT_{succ}
10	15	1.4e3	9.1e2	2.0e3	1.4e3	0	*57e+0*	*44e+0*	*79e+0*	7.1e4
1	4	1.7e5	1.6e5	1.9e5	4.1e4	.	.	.	.	.
1e−1	0	*20e−1*	*99e−2*	*30e−1*	2.2e4	.	.	.	.	.
1e−3	.	.	.	.	.	.	.	.	.	.
1e−5	.	.	.	.	.	.	.	.	.	.
1e−8	.	.	.	.	.	.	.	.	.	.

Δf	f_{16} in 5-D, N=15, mFE=50683					f_{16} in 20-D, N=15, mFE=237055				
	#	ERT	10%	90%	RT_{succ}	#	ERT	10%	90%	RT_{succ}
10	15	8.5e1	4.0e1	1.4e2	8.5e1	14	4.2e4	2.2e4	6.4e4	3.7e4
1	15	5.6e3	3.7e3	7.6e3	5.6e3	0	*58e−1*	*30e−1*	*75e−1*	8.9e4
1e−1	11	4.3e4	3.5e4	5.0e4	2.9e4	.	.	.	.	.
1e−3	0	*60e−3*	*65e−4*	*15e−2*	3.5e4	.	.	.	.	.
1e−5	.	.	.	.	.	.	.	.	.	.
1e−8	.	.	.	.	.	.	.	.	.	.

Δf	f_{17} in 5-D, N=15, mFE=98960					f_{17} in 20-D, N=15, mFE=382145				
	#	ERT	10%	90%	RT_{succ}	#	ERT	10%	90%	RT_{succ}
10	15	1.1e1	7.4e0	1.6e1	1.1e1	15	8.5e3	2.8e2	1.6e4	8.5e3
1	15	4.5e3	2.8e3	6.3e3	4.5e3	0	*40e−1*	*27e−1*	*69e−1*	1.1e5
1e−1	2	4.5e5	3.9e5	5.1e5	3.1e4	.	.	.	.	.
1e−3	0	*18e−2*	*75e−3*	*46e−2*	7.1e3	.	.	.	.	.
1e−5	.	.	.	.	.	.	.	.	.	.
1e−8	.	.	.	.	.	.	.	.	.	.

Δf	f_{18} in 5-D, N=15, mFE=75760					f_{18} in 20-D, N=15, mFE=334499				
	#	ERT	10%	90%	RT_{succ}	#	ERT	10%	90%	RT_{succ}
10	15	8.7e2	4.0e2	1.4e3	8.7e2	1	3.3e6	3.0e6	3.5e6	2.1e5
1	12	2.6e4	1.7e4	3.6e4	2.2e4	0	*18e+0*	*10e+0*	*29e+0*	1.3e5
1e−1	4	1.9e5	1.6e5	2.1e5	5.6e4	.	.	.	.	.
1e−3	0	*44e−2*	*51e−3*	*17e−1*	1.3e4	.	.	.	.	.
1e−5	.	.	.	.	.	.	.	.	.	.
1e−8	.	.	.	.	.	.	.	.	.	.

Δf	f_{19} in 5-D, N=15, mFE=64489					f_{19} in 20-D, N=15, mFE=235753				
	#	ERT	10%	90%	RT_{succ}	#	ERT	10%	90%	RT_{succ}
10	15	1.3e1	1.1e1	1.5e1	1.3e1	15	9.1e1	7.0e1	1.2e2	9.1e1
1	15	3.1e3	7.7e2	5.4e3	3.1e3	4	7.0e5	6.4e5	7.6e5	1.9e5
1e−1	4	1.8e5	1.6e5	2.0e5	4.7e4	0	*10e−1*	*78e−2*	*17e−1*	1.0e5
1e−3	0	*14e−2*	*67e−3*	*27e−2*	2.8e4	.	.	.	.	.
1e−5	.	.	.	.	.	.	.	.	.	.
1e−8	.	.	.	.	.	.	.	.	.	.

Δf	f_{20} in 5-D, N=15, mFE=50543					f_{20} in 20-D, N=15, mFE=246946				
	#	ERT	10%	90%	RT_{succ}	#	ERT	10%	90%	RT_{succ}
10	15	2.6e1	2.2e1	3.0e1	2.6e1	15	2.2e2	1.9e2	2.5e2	2.2e2
1	15	6.6e3	4.6e3	8.6e3	6.6e3	1	3.0e6	2.8e6	3.2e6	2.4e5
1e−1	0	*67e−2*	*47e−2*	*71e−2*	1.4e4	0	*15e−1*	*12e−1*	*18e−1*	1.6e5
1e−3	.	.	.	.	.	.	.	.	.	.
1e−5	.	.	.	.	.	.	.	.	.	.
1e−8	.	.	.	.	.	.	.	.	.	.

Δf	f_{21} in 5-D, N=15, mFE=21852					f_{21} in 20-D, N=15, mFE=209687				
	#	ERT	10%	90%	RT_{succ}	#	ERT	10%	90%	RT_{succ}
10	15	9.1e1	2.2e1	1.6e2	9.1e1	15	7.1e3	3.6e3	1.1e4	7.1e3
1	15	3.1e3	2.2e3	3.9e3	3.1e3	13	8.1e4	5.8e4	1.1e5	7.7e4
1e−1	15	7.1e3	5.3e3	9.2e3	7.1e3	13	1.1e5	8.4e4	1.4e5	1.0e5
1e−3	15	7.1e3	5.2e3	9.2e3	7.1e3	13	1.1e5	8.6e4	1.4e5	1.1e5
1e−5	15	7.2e3	5.2e3	9.0e3	7.2e3	13	1.1e5	8.6e4	1.4e5	1.1e5
1e−8	15	7.2e3	5.3e3	9.4e3	7.2e3	13	1.1e5	8.6e4	1.4e5	1.1e5

Δf	f_{22} in 5-D, N=15, mFE=11449					f_{22} in 20-D, N=15, mFE=211588				
	#	ERT	10%	90%	RT_{succ}	#	ERT	10%	90%	RT_{succ}
10	15	5.4e2	3.0e2	8.1e2	5.4e2	15	8.2e3	5.1e3	1.1e4	8.2e3
1	15	3.0e3	2.2e3	3.8e3	3.0e3	13	7.2e4	4.7e4	9.6e4	6.2e4
1e−1	15	4.2e3	3.2e3	5.1e3	4.2e3	7	3.5e5	3.1e5	3.8e5	1.6e5
1e−3	15	4.2e3	3.3e3	5.2e3	4.2e3	7	3.5e5	3.1e5	3.8e5	1.6e5
1e−5	15	4.3e3	3.4e3	5.3e3	4.3e3	7	3.6e5	3.2e5	3.9e5	1.7e5
1e−8	15	4.3e3	3.4e3	5.4e3	4.3e3	7	3.6e5	3.3e5	3.9e5	1.7e5

Δf	f_{23} in 5-D, N=15, mFE=50964					f_{23} in 20-D, N=15, mFE=204141				
	#	ERT	10%	90%	RT_{succ}	#	ERT	10%	90%	RT_{succ}
10	15	6.2e0	4.7e0	7.6e0	6.2e0	15	6.7e0	5.3e0	7.9e0	6.7e0
1	15	4.7e2	2.3e2	7.4e2	4.7e2	15	3.7e3	2.1e3	5.5e3	3.7e3
1e−1	13	2.7e4	2.0e4	3.4e4	1.9e4	1	2.9e6	2.7e6	3.0e6	2.0e5
1e−3	4	1.6e5	1.3e5	1.7e5	4.3e4	0	*18e−2*	*10e−2*	*26e−2*	1.1e5
1e−5	2	3.4e5	3.0e5	3.8e5	5.0e4	.	.	.	.	.
1e−8	1	7.3e5	7.0e5	7.6e5	5.0e4	.	.	.	.	.

Δf	f_{24} in 5-D, N=15, mFE=50593					f_{24} in 20-D, N=15, mFE=222121				
	#	ERT	10%	90%	RT_{succ}	#	ERT	10%	90%	RT_{succ}
10	15	2.9e3	2.0e3	3.9e3	2.9e3	0	*60e+0*	*31e+0*	*10e+1*	1.1e5
1	2	3.4e5	3.1e5	3.8e5	1.7e4	.	.	.	.	.
1e−1	0	*20e−1*	*98e−2*	*41e−1*	2.0e4	.	.	.	.	.
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 1: Multistart procedure of Nelder-Mead in Matlab

```
function [x, ilaunch] = 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(1e8*DIM, maxfunevals), ...
                     'MaxIter', 2e3*DIM, ...
                     'Tolfun', 1e-11, ...
                     'TolX', 1e-11, ...
                     'OutputFcn', @callback, ...
                     'Display', 'off');

  % multistart such that ftarget is reached with reasonable prob.
  for ilaunch = 1:100;  % relaunch optimizer up to 100 times
    % set initial conditions
    if mod(ilaunch-1, floor(1 + 3 * rand(1,1))) == 0
      xstart = 8 * rand(DIM, 1) - 4;  % random start solution
      usual_delta = 2;
    else
      xstart = x;  % try to improve found solution
      usual_delta = 0.1 * 0.1^rand(1,1);
    end
    % try fminsearch from Matlab, modified to take usual_delta as arg
    x = fminsearch_mod(FUN, xstart, usual_delta, options);
    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
```

liable black-box search algorithm for low dimensional search spaces. In contrary, for search space dimension larger than five it cannot be thoroughly recommended.

Acknowledgments

The first 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.

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

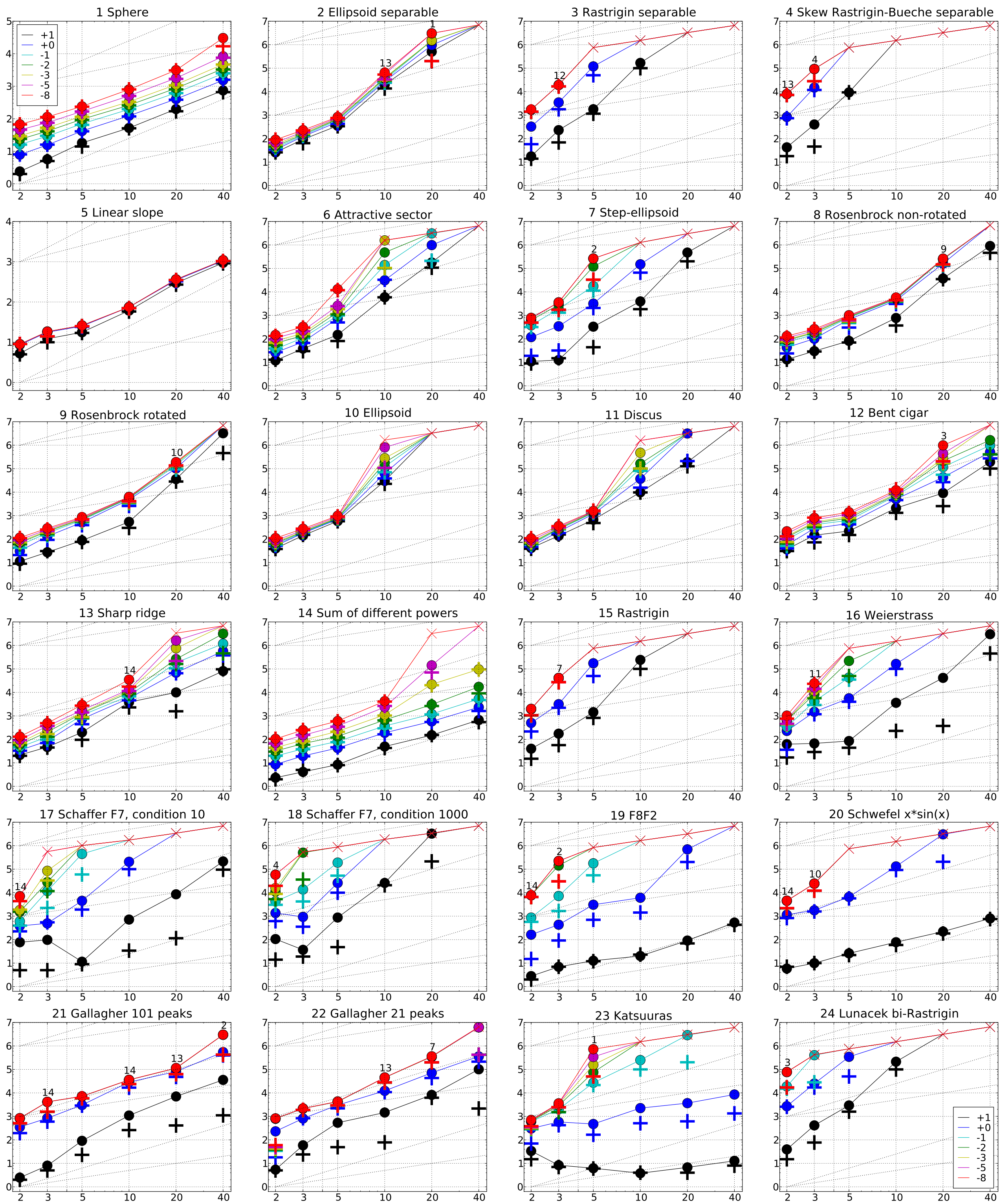

Figure 2: Expected Running Time (ERT, ●) to reach $f_{\rm opt} + \Delta f$ and median number of function evaluations of successful trials (+), shown for $\Delta f = 10, 1, 10^{-1}, 10^{-2}, 10^{-3}, 10^{-5}, 10^{-8}$ (the exponent is given in the legend of f_1 and f_{24}) versus dimension in log-log presentation. The $\mathrm{ERT}(\Delta f)$ equals to $\#\mathrm{FEs}(\Delta f)$ divided by the number of successful trials, where a trial is successful if $f_{\rm opt} + \Delta f$ was surpassed during the trial. The $\#\mathrm{FEs}(\Delta f)$ 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 ($\times$) 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.

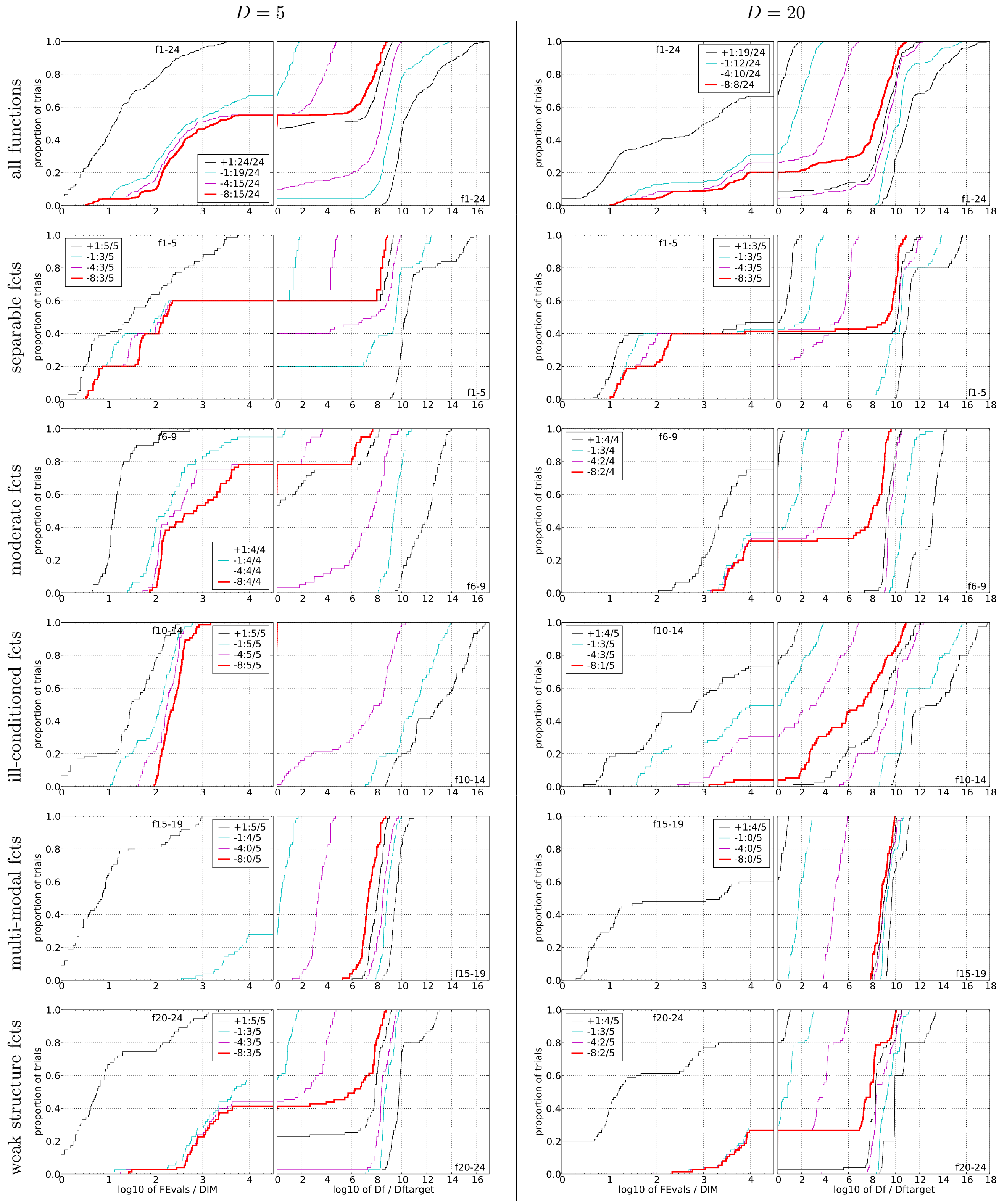

Figure 3: Empirical cumulative distribution functions (ECDFs), plotting the fraction of trials versus running time (left) or Δf. Left subplots: ECDF of the running time (number of function evaluations), divided by search space dimension D, to fall below $f_{\rm opt} + \Delta f$ with $\Delta f = 10^k$, 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.