Benchmarking the NEWUOA on the BBOB-2009 Noisy Testbed

Example Paper [*]

The BBOBies

ABSTRACT

The NEWUOA which belongs to the class of Derivative-Free Optimization (DFO) algorithms is benchmarked on the BBOB-2010 noisy testbed. A multistart strategy is applied with a maximum number of function evaluations of 10^4 times the search space dimension.

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, Derivative-free optimization

1. ALGORITHM PRESENTATION

The NEWUOA (New Unconstrained Optimization Algorithm) [4] is a Derivative-Free Optimization (DFO) algorithm using the trust region paradigm. NEWUOA computes a quadratic interpolation of the objective function in the current trust region and performs a truncated conjugate gradient minimization of the surrogate model in the trust region. It then updates either the current best point or the radius of the trust region, based on the a posteriori interpolation error.

The time complexity of the algorithm is $\mathcal{O}(m^2 n)$ in the worst case but in practice closer to $\mathcal{O}(mn)$, where m is the number of interpolation points used for the determination of the quadratic model and n is the dimension of the search space. The number of interpolation points m is a parameter of the algorithm and needs to be chosen in the range $[n+2, \frac{(n+1)(n+2)}{2}]$. Other parameters of the algorithm are the initial and final radii of the trust region, respectively governing the initial 'granularity' and the precision of the search. A simple stochastic independent restart procedure (as advised in [2]) was added to improve the probability of the algorithm reaching a given target function value.

2. EXPERIMENTAL PROCEDURE

The implementation used for our experiments is the one provided by Matthieu Guibert[1] which delivers Powell's original Fortran source code of the algorithm. This Fortran code has been integrated with the BBOB experimental paradigm. In this paper, the maximum number of points $m = \frac{(n+1)(n+2)}{2}$ has been used. Though the scaling of the algorithm time complexity is close to $\mathcal{O}(n^4)$, preliminary experiments showed the full model to perform better on noisy functions than a smaller model.

The initial radius ρ_{beg} of the search region has been set to 10, the range of the search space. Preliminary experiments shows very few dependencies on this parameter, given it is not too small (ie. by many orders of magnitude) for the problem considered. A final radius $\rho_{\mathrm{end}} = 10^{-16}$ was chosen close to the limit being the machine precision to prevent numerical errors.

The starting point x_0 is chosen uniformly in $[-5, 5]^n$ where n denotes the dimension. The multistart strategy was used with at most 100 restarts to reduce the duration of an experiment. For the same reason, a run is limited to at most $10^4 \times n$ function evaluations. The algorithm used is presented in Figure 1. No parameter tuning was done, the CrE [2] is computed to zero.

3. RESULTS

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

The algorithm solves some of the moderate noise function f_{101}, f_{102}, f_{103}, f_{104}, f_{106}. Furthermore, f_{105}, f_{107}, f_{109}, f_{112}, f_{113}, f_{115}, f_{125}, f_{127}, f_{128}, f_{130} are solved only for dimensions 2 or 3. Noise greatly affects such trust region method, especially the uniform noise model.

4. CPU TIMING EXPERIMENT

[*]Submission deadline: March 25th.

[1]`http://www.inrialpes.fr/bipop/people/guilbert/newuoa/newuoa.html`

GECCO'10, July 7–11, 2010, Portland Oregon, United States of America.

Figure 1: Multistart NEWUOA

```
#include <stdlib.h>
#include <math.h>
#include <stdio.h>
#include "bbobStructures.h"

/* Call to the Fortran function */
extern void newuoa_(unsigned int* n, int* m, double* x0, double* rhobeg,
                    double* rhoend, int* verbose, int* maxfun,
                    double* W, double* ftarget);

/* The Multistart NEWUOA */
void newuoa(unsigned int dim, unsigned int maxfunevals, double ftarget)
{
    int m, iprint = 0, curmaxfun;
    double * x = malloc(sizeof(double) * dim);
    unsigned int iter = 0, i;
    double rhobeg = 10, rhoend = 1e-16;
    /* internal variable of NEWUOA */
    double * w = malloc(1000000 * sizeof(double));

    m = 2 * dim + 1;

    curmaxfun = maxfunevals - fgeneric_evaluations();
    while (curmaxfun > 0 && fgeneric_best() > ftarget && iter < 100)
    {
        /* Generate a starting point */
        for (i = 0; i < dim; i++)
             x[i] = 10. * ((double)rand() / RAND_MAX) - 5.;
        /* Call NEWUOA */
        newuoa_(&dim, &npt, x, &rhobeg, &rhoend, &iprint, &curmaxfun, w, &ftarget);
        /* Update */
        curmaxfun = maxfunevals - fgeneric_evaluations();
        iter++;
    }
    free(x);
    free(w);
}
```

For the timing experiment, the proposed algorithm was run on f_8 and restarted until at least 30 seconds have passed (according to [2]). The experiments were conducted with an Intel Core 2 6700 processor (2.66GHz) on Linux 2.6.24.7. The results are 9.0, 15, 38, 240, 2400 and 32000 $\times 10^{-6}$ seconds per function evaluation for the full model in dimension 2, 3, 5, 10, 20, 40 and 80 respectively.

5. CONCLUSION

The NEWUOA which is a trust region method was tested with restarts on a noisy testbed. Method based on interpolation are expected to fail on noisy functions. Results of this algorithm do not disagree with this.

6. REFERENCES

[1] S. Finck, N. Hansen, R. Ros, and A. Auger. Real-parameter black-box optimization benchmarking 2010: Presentation of the noisy functions. Technical Report 2009/21, Research Center PPE, 2010.

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

[3] N. Hansen, S. Finck, R. Ros, and A. Auger. Real-parameter black-box optimization benchmarking 2009: Noisy functions definitions. Technical Report RR-6869, INRIA, 2009. Updated February 2010.

[4] M. J. D. Powell. The NEWUOA software for unconstrained optimization without derivatives. *Large Scale Nonlinear Optimization*, pages 255–297, 2006.

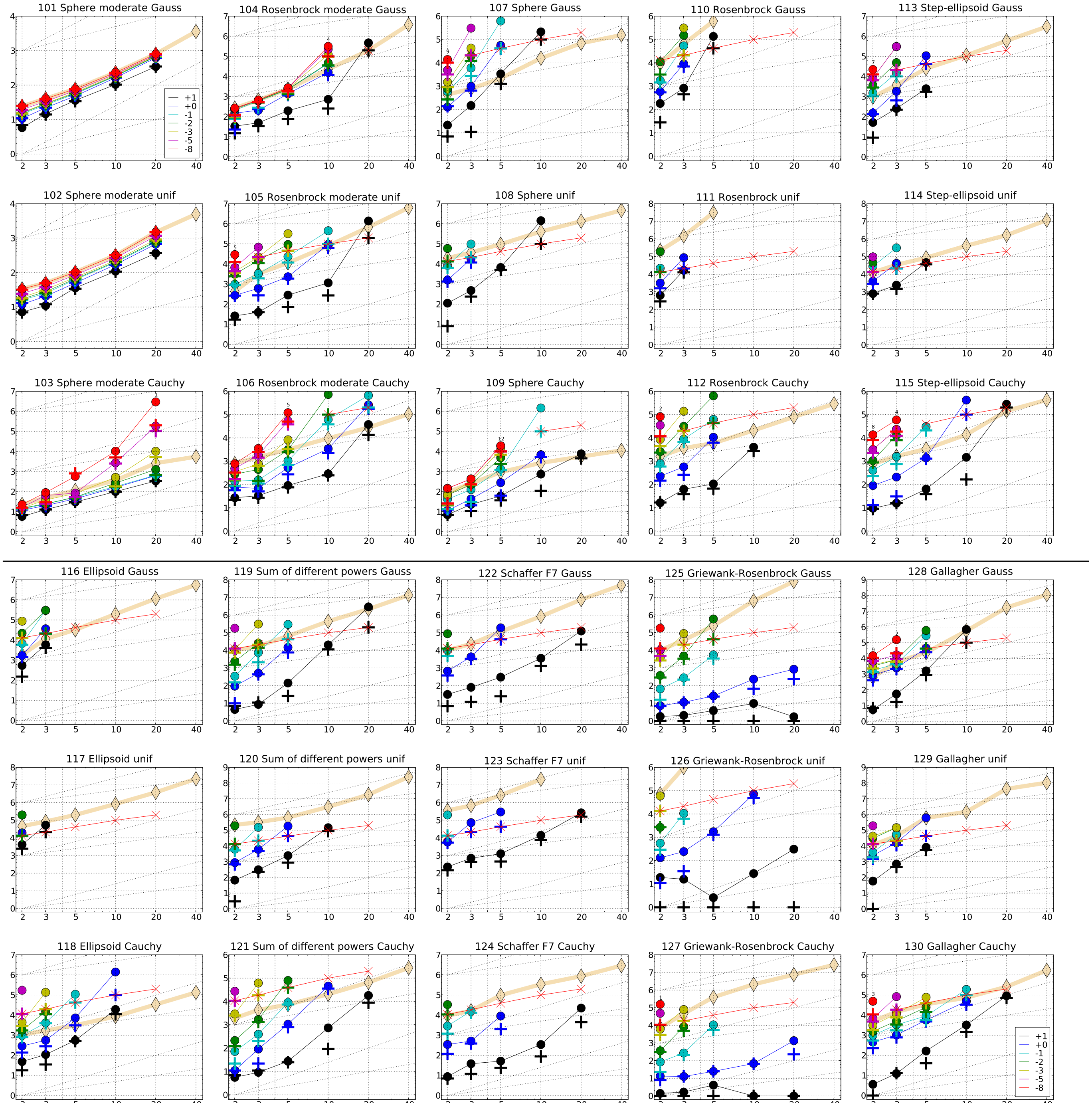

Figure 2: Expected Running Time (ERT, ●) to reach $f_{\rm opt} + \Delta f$ and median number of f-evaluations from successful trials (+), for $\Delta f = 10^{\{+1,0,-1,-2,-3,-5,-8\}}$ (the exponent is given in the legend of f_{101} and f_{130}) versus dimension in log-log presentation. For each function 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 $f_{\rm opt} + \Delta f$ was surpassed. The $\#\mathrm{FEs}(\Delta f)$ are the total number (sum) of f-evaluations while $f_{\rm opt} + \Delta f$ was not surpassed in the trial, from all (successful and unsuccessful) trials, and $f_{\rm opt}$ is the optimal function value. Crosses ($\times$) 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.

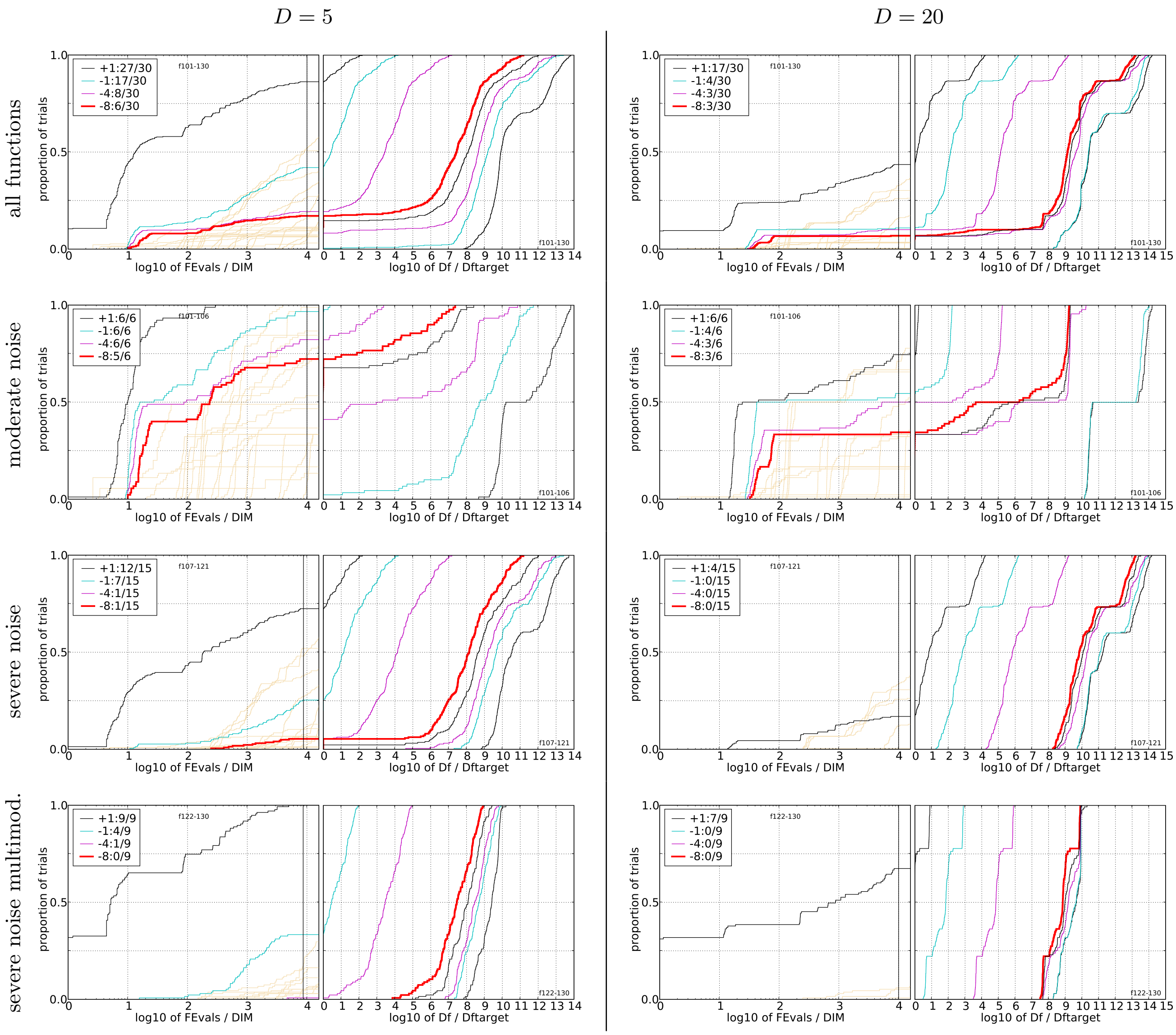

Figure 3: Empirical cumulative distribution functions (ECDFs), plotting the fraction of trials versus running time (left subplots) or versus Δf (right subplots). The thick red line represents the best achieved results. Left subplots: ECDF of the running time (number of function evaluations), divided by search space dimension D, to fall below $f_{\rm opt} + \Delta f$ 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). 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.

	f101 in 5-D, N=15, mFE=89					**f101 in 20-D**, N=15, mFE=955				
Δf	#	ERT	10%	90%	RT_{succ}	#	ERT	10%	90%	RT_{succ}
10	15	3.6e1	3.1e1	4.5e1	3.6e1	15	3.4e2	3.0e2	3.9e2	3.4e2
1	15	4.8e1	4.2e1	5.2e1	4.8e1	15	6.1e2	5.5e2	6.8e2	6.1e2
1e−1	15	5.6e1	5.0e1	6.3e1	5.6e1	15	6.6e2	5.6e2	7.6e2	6.6e2
1e−3	15	6.2e1	5.7e1	6.7e1	6.2e1	15	7.0e2	6.0e2	7.9e2	7.0e2
1e−5	15	6.9e1	6.3e1	7.7e1	6.9e1	15	7.4e2	6.3e2	8.2e2	7.4e2
1e−8	15	7.7e1	6.9e1	8.5e1	7.7e1	15	8.0e2	6.9e2	8.9e2	8.0e2

	f102 in 5-D, N=15, mFE=125					**f102 in 20-D**, N=15, mFE=1641				
Δf	#	ERT	10%	90%	RT_{succ}	#	ERT	10%	90%	RT_{succ}
10	15	3.6e1	2.6e1	4.6e1	3.6e1	15	3.7e2	3.2e2	4.1e2	3.7e2
1	15	5.0e1	4.0e1	6.1e1	5.0e1	15	6.7e2	6.1e2	7.4e2	6.7e2
1e−1	15	6.2e1	5.1e1	7.7e1	6.2e1	15	7.6e2	6.5e2	8.4e2	7.6e2
1e−3	15	7.2e1	5.9e1	8.4e1	7.2e1	15	9.2e2	7.7e2	1.1e3	9.2e2
1e−5	15	8.6e1	7.2e1	1.0e2	8.6e1	15	1.2e3	1.0e3	1.3e3	1.2e3
1e−8	15	1.0e2	8.8e1	1.2e2	1.0e2	15	1.5e3	1.4e3	1.6e3	1.5e3

	f103 in 5-D, N=15, mFE=1280					**f103 in 20-D**, N=15, mFE=200000				
Δf	#	ERT	10%	90%	RT_{succ}	#	ERT	10%	90%	RT_{succ}
10	15	3.1e1	2.5e1	3.9e1	3.1e1	15	3.4e2	3.1e2	3.7e2	3.4e2
1	15	4.5e1	3.5e1	5.3e1	4.5e1	15	6.3e2	5.4e2	7.5e2	6.3e2
1e−1	15	5.3e1	4.7e1	6.5e1	5.3e1	15	6.6e2	5.4e2	8.2e2	6.6e2
1e−3	15	8.1e1	4.8e1	6.7e1	8.1e1	15	1.0e4	5.5e2	2.3e4	1.0e4
1e−5	15	8.5e1	5.0e1	7.2e1	8.5e1	10	1.8e5	3.6e4	4.4e5	8.3e4
1e−8	15	5.8e2	5.0e1	1.2e3	5.8e2	1	2.9e6	3.5e5	6.5e6	1.5e5

	f104 in 5-D, N=15, mFE=10816					**f104 in 20-D**, N=15, mFE=200000				
Δf	#	ERT	10%	90%	RT_{succ}	#	ERT	10%	90%	RT_{succ}
10	15	2.0e2	6.4e1	9.4e2	2.0e2	5	4.7e5	3.1e4	1.1e6	7.3e4
1	15	1.4e3	1.8e2	2.8e3	1.4e3	0	*13e+0*	*89e−1*	*16e+0*	1.3e5
1e−1	15	2.2e3	2.7e2	3.8e3	2.2e3	.	.	.	.	.
1e−3	15	2.5e3	7.0e2	3.9e3	2.5e3	.	.	.	.	.
1e−5	15	2.7e3	7.4e2	4.5e3	2.7e3	.	.	.	.	.
1e−8	15	2.7e3	7.6e2	4.5e3	2.7e3	.	.	.	.	.

	f105 in 5-D, N=15, mFE=46065					**f105 in 20-D**, N=15, mFE=200000				
Δf	#	ERT	10%	90%	RT_{succ}	#	ERT	10%	90%	RT_{succ}
10	15	2.8e2	5.4e1	9.6e2	2.8e2	2	1.4e6	1.6e5	3.2e6	9.0e4
1	15	2.3e3	7.1e1	5.6e3	2.3e3	0	*14e+0*	*45e−1*	*17e+0*	8.9e4
1e−1	12	2.3e4	1.1e3	5.7e4	1.2e4	.	.	.	.	.
1e−3	2	3.2e5	3.7e4	7.5e5	2.6e4	.	.	.	.	.
1e−5	0	*22e−3*	*72e−5*	*19e−2*	1.8e4	.	.	.	.	.
1e−8	.	.	.	.	.	.	.	.	.	.

	f106 in 5-D, N=15, mFE=50000					**f106 in 20-D**, N=15, mFE=200000				
Δf	#	ERT	10%	90%	RT_{succ}	#	ERT	10%	90%	RT_{succ}
10	15	9.2e1	5.5e1	1.5e2	9.2e1	15	3.7e4	2.5e3	1.2e5	3.7e4
1	15	5.3e2	1.2e2	1.0e3	5.3e2	8	2.6e5	1.3e4	6.1e5	8.6e4
1e−1	15	1.1e3	2.2e2	2.9e3	1.1e3	4	6.6e5	8.8e4	1.5e6	1.1e5
1e−3	15	8.3e3	4.2e2	2.8e4	8.3e3	0	*97e−2*	*30e−3*	*77e−1*	1.4e5
1e−5	9	5.3e4	3.0e3	1.3e5	2.0e4	.	.	.	.	.
1e−8	5	1.2e5	3.0e3	2.8e5	2.0e4	.	.	.	.	.

	f107 in 5-D, N=15, mFE=42187					**f107 in 20-D**, N=15, mFE=200000				
Δf	#	ERT	10%	90%	RT_{succ}	#	ERT	10%	90%	RT_{succ}
10	15	3.4e3	2.5e1	1.1e4	3.4e3	0	*68e+0*	*46e+0*	*10e+1*	1.0e5
1	8	5.6e4	5.1e3	1.3e5	1.9e4	.	.	.	.	.
1e−1	1	6.2e5	7.5e4	1.5e6	3.3e4	.	.	.	.	.
1e−3	0	*96e−2*	*29e−2*	*28e−1*	2.5e4	.	.	.	.	.
1e−5	.	.	.	.	.	.	.	.	.	.
1e−8	.	.	.	.	.	.	.	.	.	.

	f108 in 5-D, N=15, mFE=42641					**f108 in 20-D**, N=15, mFE=200000				
Δf	#	ERT	10%	90%	RT_{succ}	#	ERT	10%	90%	RT_{succ}
10	15	6.7e3	1.0e0	1.2e4	6.7e3	0	*11e+1*	*76e+0*	*16e+1*	1.0e5
1	0	*43e−1*	*14e−1*	*85e−1*	2.2e4	.	.	.	.	.
1e−1	.	.	.	.	.	.	.	.	.	.
1e−3	.	.	.	.	.	.	.	.	.	.
1e−5	.	.	.	.	.	.	.	.	.	.
1e−8	.	.	.	.	.	.	.	.	.	.

	f109 in 5-D, N=15, mFE=40192					**f109 in 20-D**, N=15, mFE=200000				
Δf	#	ERT	10%	90%	RT_{succ}	#	ERT	10%	90%	RT_{succ}
10	15	6.1e1	2.6e1	4.6e1	6.1e1	15	7.7e3	3.3e2	2.1e4	7.7e3
1	15	2.8e2	4.8e1	8.1e2	2.8e2	0	*29e−1*	*20e−1*	*48e−1*	1.4e5
1e−1	15	1.3e3	6.3e1	2.8e3	1.3e3	.	.	.	.	.
1e−3	14	1.2e4	8.3e2	2.7e4	9.2e3	.	.	.	.	.
1e−5	12	1.9e4	2.0e3	4.7e4	9.2e3	.	.	.	.	.
1e−8	12	1.9e4	2.0e3	5.0e4	9.2e3	.	.	.	.	.

	f110 in 5-D, N=15, mFE=42567					**f110 in 20-D**, N=15, mFE=200000				
Δf	#	ERT	10%	90%	RT_{succ}	#	ERT	10%	90%	RT_{succ}
10	4	1.4e5	2.1e4	3.2e5	2.0e4	0	*65e+3*	*36e+3*	*10e+4*	8.9e4
1	0	*15e+0*	*39e−1*	*32e+0*	2.5e4	.	.	.	.	.
1e−1	.	.	.	.	.	.	.	.	.	.
1e−3	.	.	.	.	.	.	.	.	.	.
1e−5	.	.	.	.	.	.	.	.	.	.
1e−8	.	.	.	.	.	.	.	.	.	.

	f111 in 5-D, N=15, mFE=42231					**f111 in 20-D**, N=15, mFE=200000				
Δf	#	ERT	10%	90%	RT_{succ}	#	ERT	10%	90%	RT_{succ}
10	0	*16e+1*	*24e+0*	*92e+1*	1.6e4	0	*11e+4*	*47e+3*	*15e+4*	8.9e4
1	.	.	.	.	.	.	.	.	.	.
1e−1	.	.	.	.	.	.	.	.	.	.
1e−3	.	.	.	.	.	.	.	.	.	.
1e−5	.	.	.	.	.	.	.	.	.	.
1e−8	.	.	.	.	.	.	.	.	.	.

	f112 in 5-D, N=15, mFE=43660					**f112 in 20-D**, N=15, mFE=200000				
Δf	#	ERT	10%	90%	RT_{succ}	#	ERT	10%	90%	RT_{succ}
10	15	1.1e2	5.7e1	1.3e2	1.1e2	0	*19e+0*	*16e+0*	*21e+0*	7.1e4
1	14	1.1e4	5.0e2	2.7e4	7.6e3	.	.	.	.	.
1e−1	8	6.2e4	6.2e3	1.3e5	2.4e4	.	.	.	.	.
1e−3	0	*93e−3*	*17e−3*	*98e−2*	2.8e4	.	.	.	.	.
1e−5	.	.	.	.	.	.	.	.	.	.
1e−8	.	.	.	.	.	.	.	.	.	.

	f113 in 5-D, N=15, mFE=42375					**f113 in 20-D**, N=15, mFE=200000				
Δf	#	ERT	10%	90%	RT_{succ}	#	ERT	10%	90%	RT_{succ}
10	15	2.5e3	4.1e1	5.9e3	2.5e3	0	*38e+1*	*19e+1*	*60e+1*	1.3e5
1	5	1.1e5	8.9e3	2.5e5	2.3e4	.	.	.	.	.
1e−1	0	*14e−1*	*73e−2*	*19e−1*	2.2e4	.	.	.	.	.
1e−3	.	.	.	.	.	.	.	.	.	.
1e−5	.	.	.	.	.	.	.	.	.	.
1e−8	.	.	.	.	.	.	.	.	.	.

	f114 in 5-D, N=15, mFE=42513					**f114 in 20-D**, N=15, mFE=200000				
Δf	#	ERT	10%	90%	RT_{succ}	#	ERT	10%	90%	RT_{succ}
10	9	4.6e4	2.2e3	9.9e4	1.8e4	0	*53e+1*	*36e+1*	*63e+1*	1.3e5
1	0	*82e−1*	*31e−1*	*22e+0*	1.4e4	.	.	.	.	.
1e−1	.	.	.	.	.	.	.	.	.	.
1e−3	.	.	.	.	.	.	.	.	.	.
1e−5	.	.	.	.	.	.	.	.	.	.
1e−8	.	.	.	.	.	.	.	.	.	.

	f115 in 5-D, N=15, mFE=40298					**f115 in 20-D**, N=15, mFE=200000				
Δf	#	ERT	10%	90%	RT_{succ}	#	ERT	10%	90%	RT_{succ}
10	15	6.4e1	2.9e1	5.5e1	6.4e1	7	2.8e5	1.8e4	6.4e5	4.9e4
1	15	1.4e3	6.6e1	2.5e3	1.4e3	0	*10e+0*	*60e−1*	*15e+0*	7.1e4
1e−1	11	3.0e4	7.6e3	6.9e4	1.6e4	.	.	.	.	.
1e−3	0	*57e−3*	*18e−3*	*31e−2*	2.2e4	.	.	.	.	.
1e−5	.	.	.	.	.	.	.	.	.	.
1e−8	.	.	.	.	.	.	.	.	.	.

	f116 in 5-D, N=15, mFE=42432					**f116 in 20-D**, N=15, mFE=200000				
Δf	#	ERT	10%	90%	RT_{succ}	#	ERT	10%	90%	RT_{succ}
10	0	*10e+1*	*40e+0*	*17e+1*	1.8e4	0	*23e+3*	*12e+3*	*30e+3*	1.0e5
1	.	.	.	.	.	.	.	.	.	.
1e−1	.	.	.	.	.	.	.	.	.	.
1e−3	.	.	.	.	.	.	.	.	.	.
1e−5	.	.	.	.	.	.	.	.	.	.
1e−8	.	.	.	.	.	.	.	.	.	.

	f117 in 5-D, N=15, mFE=42254					**f117 in 20-D**, N=15, mFE=200000				
Δf	#	ERT	10%	90%	RT_{succ}	#	ERT	10%	90%	RT_{succ}
10	0	*28e+1*	*60e+0*	*10e+2*	2.0e4	0	*32e+3*	*21e+3*	*39e+3*	1.1e5
1	.	.	.	.	.	.	.	.	.	.
1e−1	.	.	.	.	.	.	.	.	.	.
1e−3	.	.	.	.	.	.	.	.	.	.
1e−5	.	.	.	.	.	.	.	.	.	.
1e−8	.	.	.	.	.	.	.	.	.	.

	f118 in 5-D, N=15, mFE=42991					**f118 in 20-D**, N=15, mFE=200000				
Δf	#	ERT	10%	90%	RT_{succ}	#	ERT	10%	90%	RT_{succ}
10	15	5.5e2	7.7e1	1.3e3	5.5e2	0	*44e+0*	*29e+0*	*61e+0*	5.6e4
1	15	7.2e3	1.4e3	1.8e4	7.2e3	.	.	.	.	.
1e−1	5	1.1e5	2.0e4	2.5e5	2.4e4	.	.	.	.	.
1e−3	0	*22e−2*	*50e−3*	*50e−2*	2.5e4	.	.	.	.	.
1e−5	.	.	.	.	.	.	.	.	.	.
1e−8	.	.	.	.	.	.	.	.	.	.

	f119 in 5-D, N=15, mFE=42274					**f119 in 20-D**, N=15, mFE=200000				
Δf	#	ERT	10%	90%	RT_{succ}	#	ERT	10%	90%	RT_{succ}
10	15	1.4e2	2.2e1	4.5e2	1.4e2	1	2.9e6	3.0e5	6.9e6	9.9e4
1	14	1.5e4	4.7e2	3.4e4	1.2e4	0	*18e+0*	*12e+0*	*20e+0*	1.3e5
1e−1	2	3.0e5	3.6e4	6.8e5	2.3e4	.	.	.	.	.
1e−3	0	*31e−2*	*82e−3*	*81e−2*	2.0e4	.	.	.	.	.
1e−5	.	.	.	.	.	.	.	.	.	.
1e−8	.	.	.	.	.	.	.	.	.	.

	f120 in 5-D, N=15, mFE=42788					**f120 in 20-D**, N=15, mFE=200000				
Δf	#	ERT	10%	90%	RT_{succ}	#	ERT	10%	90%	RT_{succ}
10	15	2.4e3	2.3e1	5.1e3	2.4e3	0	*34e+0*	*27e+0*	*44e+0*	7.9e4
1	3	1.8e5	6.8e3	4.3e5	1.2e4	.	.	.	.	.
1e−1	0	*22e−1*	*80e−2*	*32e−1*	1.6e4	.	.	.	.	.
1e−3	.	.	.	.	.	.	.	.	.	.
1e−5	.	.	.	.	.	.	.	.	.	.
1e−8	.	.	.	.	.	.	.	.	.	.

Table 1: Shown are, for functions f_{101}-f_{120} and 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.

	f_{121} in 5-D, N=15, mFE=39962					**f_{121} in 20-D**, N=15, mFE=200000				
Δf	#	ERT	10%	90%	RT_{succ}	#	ERT	10%	90%	RT_{succ}
10	15	2.7e1	2.2e1	3.7e1	2.7e1	15	1.8e4	2.9e2	6.4e4	1.8e4
1	15	1.1e3	4.5e1	3.5e3	1.1e3	0	*53e−1*	*26e−1*	*79e−1*	1.4e5
1e−1	15	9.0e3	7.5e1	2.0e4	9.0e3	.	.	.	.	.
1e−3	0	*13e−3*	*50e−4*	*42e−3*	2.8e4	.	.	.	.	.
1e−5	.	.	.	.	.	.	.	.	.	.
1e−8	.	.	.	.	.	.	.	.	.	.

	f_{122} in 5-D, N=15, mFE=42104					**f_{122} in 20-D**, N=15, mFE=200000				
Δf	#	ERT	10%	90%	RT_{succ}	#	ERT	10%	90%	RT_{succ}
10	15	3.0e2	1.0e0	1.7e3	3.0e2	10	1.3e5	4.5e3	4.0e5	2.6e4
1	3	1.9e5	2.5e4	4.1e5	2.2e4	0	*76e−1*	*60e−1*	*11e+0*	1.3e5
1e−1	0	*17e−1*	*77e−2*	*22e−1*	3.5e4	.	.	.	.	.
1e−3	.	.	.	.	.	.	.	.	.	.
1e−5	.	.	.	.	.	.	.	.	.	.
1e−8	.	.	.	.	.	.	.	.	.	.

	f_{123} in 5-D, N=15, mFE=42805					**f_{123} in 20-D**, N=15, mFE=200000				
Δf	#	ERT	10%	90%	RT_{succ}	#	ERT	10%	90%	RT_{succ}
10	15	1.3e3	2.5e1	2.6e3	1.3e3	8	2.6e5	1.8e4	5.6e5	8.8e4
1	2	3.0e5	3.4e4	6.7e5	1.8e4	0	*99e−1*	*83e−1*	*19e+0*	1.3e5
1e−1	0	*35e−1*	*91e−2*	*46e−1*	1.6e4	.	.	.	.	.
1e−3	.	.	.	.	.	.	.	.	.	.
1e−5	.	.	.	.	.	.	.	.	.	.
1e−8	.	.	.	.	.	.	.	.	.	.

	f_{124} in 5-D, N=15, mFE=39598					**f_{124} in 20-D**, N=15, mFE=200000				
Δf	#	ERT	10%	90%	RT_{succ}	#	ERT	10%	90%	RT_{succ}
10	15	5.2e1	2.2e1	4.4e1	5.2e1	15	2.2e4	2.4e2	1.0e5	2.2e4
1	14	9.0e3	6.5e1	3.1e4	6.2e3	0	*66e−1*	*53e−1*	*80e−1*	8.9e4
1e−1	0	*47e−2*	*22e−2*	*83e−2*	1.8e4	.	.	.	.	.
1e−3	.	.	.	.	.	.	.	.	.	.
1e−5	.	.	.	.	.	.	.	.	.	.
1e−8	.	.	.	.	.	.	.	.	.	.

	f_{125} in 5-D, N=15, mFE=42183					**f_{125} in 20-D**, N=15, mFE=200000				
Δf	#	ERT	10%	90%	RT_{succ}	#	ERT	10%	90%	RT_{succ}
10	15	3.9e0	1.0e0	2.2e1	3.9e0	15	1.7e0	1.0e0	1.0e0	1.7e0
1	15	2.6e1	2.2e1	3.6e1	2.6e1	15	8.6e2	2.3e2	4.5e3	8.6e2
1e−1	15	5.6e3	5.1e2	1.3e4	5.6e3	0	*44e−2*	*33e−2*	*48e−2*	7.9e4
1e−3	0	*41e−3*	*13e−3*	*59e−3*	1.6e4	.	.	.	.	.
1e−5	.	.	.	.	.	.	.	.	.	.
1e−8	.	.	.	.	.	.	.	.	.	.

	f_{126} in 5-D, N=15, mFE=43182					**f_{126} in 20-D**, N=15, mFE=200000				
Δf	#	ERT	10%	90%	RT_{succ}	#	ERT	10%	90%	RT_{succ}
10	15	2.6e0	1.0e0	1.0e0	2.6e0	15	3.1e2	1.0e0	2.8e2	3.1e2
1	15	1.7e3	3.2e1	4.3e3	1.7e3	0	*16e−1*	*11e−1*	*24e−1*	1.3e5
1e−1	0	*25e−2*	*16e−2*	*34e−2*	1.8e4	.	.	.	.	.
1e−3	.	.	.	.	.	.	.	.	.	.
1e−5	.	.	.	.	.	.	.	.	.	.
1e−8	.	.	.	.	.	.	.	.	.	.

	f_{127} in 5-D, N=15, mFE=38685					**f_{127} in 20-D**, N=15, mFE=200000				
Δf	#	ERT	10%	90%	RT_{succ}	#	ERT	10%	90%	RT_{succ}
10	15	4.1e0	1.0e0	2.2e1	4.1e0	15	1.0e0	1.0e0	1.0e0	1.0e0
1	15	2.5e1	2.2e1	2.8e1	2.5e1	15	1.4e3	2.3e2	4.5e3	1.4e3
1e−1	14	1.1e4	2.4e3	2.4e4	7.9e3	0	*44e−2*	*40e−2*	*47e−2*	1.1e5
1e−3	0	*59e−3*	*31e−3*	*84e−3*	2.2e4	.	.	.	.	.
1e−5	.	.	.	.	.	.	.	.	.	.
1e−8	.	.	.	.	.	.	.	.	.	.

	f_{128} in 5-D, N=15, mFE=42174					**f_{128} in 20-D**, N=15, mFE=200000				
Δf	#	ERT	10%	90%	RT_{succ}	#	ERT	10%	90%	RT_{succ}
10	15	1.6e3	3.3e1	4.2e3	1.6e3	0	*71e+0*	*66e+0*	*72e+0*	6.3e4
1	9	4.5e4	2.6e3	1.1e5	1.7e4	.	.	.	.	.
1e−1	2	2.8e5	1.2e4	6.4e5	7.4e3	.	.	.	.	.
1e−3	0	*16e−2*	*18e−3*	*20e−1*	1.6e4	.	.	.	.	.
1e−5	.	.	.	.	.	.	.	.	.	.
1e−8	.	.	.	.	.	.	.	.	.	.

	f_{129} in 5-D, N=15, mFE=42757					**f_{129} in 20-D**, N=15, mFE=200000				
Δf	#	ERT	10%	90%	RT_{succ}	#	ERT	10%	90%	RT_{succ}
10	15	8.2e3	2.2e3	1.6e4	8.2e3	0	*75e+0*	*71e+0*	*78e+0*	1.1e5
1	1	6.2e5	6.9e4	1.4e6	2.7e4	.	.	.	.	.
1e−1	0	*55e−1*	*20e−1*	*78e−1*	2.2e4	.	.	.	.	.
1e−3	.	.	.	.	.	.	.	.	.	.
1e−5	.	.	.	.	.	.	.	.	.	.
1e−8	.	.	.	.	.	.	.	.	.	.

	f_{130} in 5-D, N=15, mFE=39955					**f_{130} in 20-D**, N=15, mFE=200000				
Δf	#	ERT	10%	90%	RT_{succ}	#	ERT	10%	90%	RT_{succ}
10	15	1.6e2	2.9e1	4.2e2	1.6e2	13	9.1e4	1.3e4	2.3e5	6.0e4
1	15	5.7e3	8.1e2	1.0e4	5.7e3	0	*70e−1*	*20e−1*	*10e+0*	1.3e5
1e−1	14	9.1e3	1.3e3	1.8e4	6.3e3	.	.	.	.	.
1e−3	6	7.7e4	1.3e4	1.8e5	1.8e4	.	.	.	.	.
1e−5	0	*14e−4*	*20e−5*	*56e−3*	2.2e4	.	.	.	.	.
1e−8	.	.	.	.	.	.	.	.	.	.

Table 2: Shown are, for functions f_{121}-f_{130} and 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.

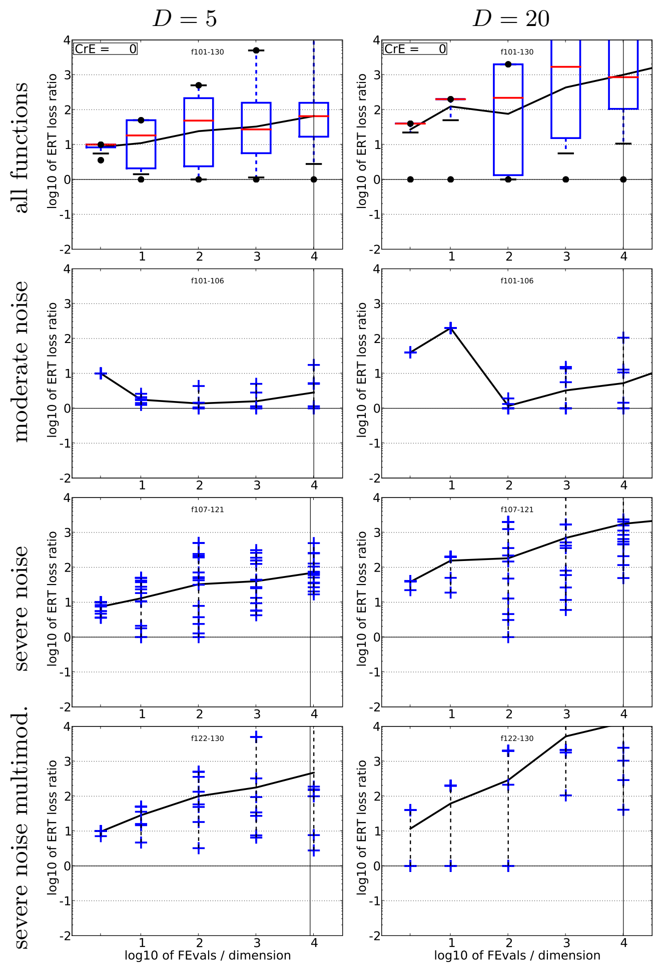

Figure 4: ERT loss ratio versus given budget FEvals. The target value f_t for ERT (see Figure 2) is the smallest (best) recorded function value such that $\mathrm{ERT}(f_t) \le \mathrm{FEvals}$ for the presented algorithm. Shown is FEvals divided by the respective best $\mathrm{ERT}(f_t)$ from BBOB-2009 for functions f_{101}–f_{130} 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.

Table 3: ERT loss ratio (see Figure 4) compared to the respective best result from BBOB-2009 for budgets given in the first column. The last row $\mathrm{RL_{US}/D}$ 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).

	f101–f130 in 5-D, maxFE/D=10000					
#FEs/D	best	10%	25%	**med**	75%	90%
2	3.6	5.1	8.3	10	10	10
10	1.0	1.3	2.1	17	50	50
100	1.0	1.0	2.4	46	2.1e2	5.0e2
1e3	1.0	1.1	5.7	27	1.6e2	2.7e3
1e4	1.0	2.0	17	63	1.6e2	2.5e4
$\mathrm{RL_{US}/D}$	8e3	8e3	8e3	8e3	9e3	9e3

	f101–f130 in 20-D, maxFE/D=10000					
#FEs/D	best	10%	25%	**med**	75%	90%
2	1.0	12	40	40	40	40
10	1.0	34	2.0e2	2.0e2	2.0e2	2.0e2
100	1.0	1.0	1.3	2.2e2	2.0e3	2.0e3
1e3	1.0	3.3	15	1.1e3	2.0e4	2.0e4
1e4	1.0	6.0	1.1e2	7.5e2	2.0e5	2.0e5
1e5	1.0	28	4.7e2	2.0e3	5.6e3	2.0e6
$\mathrm{RL_{US}/D}$	1e4	1e4	1e4	1e4	1e4	1e4