SPSA on the BBOB 2009 Noise-free Testbed

Steffen Finck
Vorarlberg University of Applied Sciences
Hochschulstrasse 1
Dornbirn, Austria
steffen.finck@fhv.at

ABSTRACT

This paper benchmarks the Simultaneous Perturbation Stochastic Algorithm (SPSA) [4] on the BBOB 2009 noise-free testbed. SPSA is a widely used optimization algorithm with its main application in noisy optimization. But it is also of interest how the algorithm behaves in an noise-free environment. The paper presents briefly the algorithm and used parameters for the testbed.

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, stochastic optimization

1. INTRODUCTION

The SPSA algorithm is a very common and widely used optimization algorithm [5] and primarily designed for noisy optimization. Nevertheless, the performance in an noise-free environment is of interest. Further to the knowledge of the author, there is no paper where SPSA is extensively benchmarked and compared with other common (not necessarily stochastic) optimization algorithm. In this paper the basic variant with a simple multistart procedure is presented. The main feature of SPSA is the use of just 2 function evaluations to determine the gradient, independent of the search space dimension DIM. As shown in [4, 6] this is advantageous (especially for large DIM) compared with common stochastic approximation algorithm which use $2 \times$ DIM function evaluations to approximate the gradient. The here presented algorithm is coupled with a simple multistart procedure to effectively use the given number of maximal function evaluations, similar to Fig. 3 in [2].

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

2. ALGORITHM PRESENTATION

In Fig. 1 the main algorithm is presented.

The gain `ak` is used for the update of the current search point, while the gain `ck` is used for the test step of the gradient approximation. The determination of their initial values is shown in Fig. 2. To improve the performance of SPSA, `lambda` gradient approximations are averaged within one iteration before the update of the current search point. Thus, the here presented SPSA uses $2 \times$ `lambda` function evaluations per iteration. The parameter `lambda` is increased during the multistart procedure.

To effectively use the allowed number of function evaluations a simple multistart procedure was implemented. It is shown in Fig. 3. To prevent infinite runs the procedure terminates if the maximal number of restarts is reached.

3. EXPERIMENTAL PROCEDURE

The gain rates were set to their recommended values `alpha` $= 0.602$ and `gamma` $= 0.101$, instead of the respective optimal values. All other parameters were set as recommended in [6]. Since the recommendation for `c0` is to set approximately to the standard deviation at the initial point, a fixed value of `c0` $= 0.001$ is used in the experiments. The experiments were conducted on a Cluster with 2.44 GHz CPUs (machine_type x86_64) under Octave 3.0.2.

4. RESULTS

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

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]). The results were $1.2; 1.2; 1.2; 1.2; 1.3$ and 1.3×10^{-4} seconds per function evaluation in dimension $2; 3; 5; 10; 20$ and 40, respectively. The dependency of CPU time on the search space dimensionality is small.

6. CONCLUSION

This paper reports the result for the basic SPSA on the BBOB 2009 noise-free testbed.

Figure 1: SPSA in Matlab

```
% simple spsa function
function [x,termvalue] = alg(FUN, x, parameter, maxGenerations, ftarget,...
                              DIM, maxfunevals)

    % intialze counters
    k = 1;

    % initialize algorithm parameter
    a0 = parameter(1);
    alpha = parameter(2);
    c0 = parameter(3);
    gamma = parameter(4);
    A = parameter(5);
    lambda = parameter(6);

    while 1

        % gain sequences ak and ck
        ak = a0 * (A + k)^(-alpha);
        ck = c0 * k^(-gamma);

        % gradient approximation with averaging of several approximations
        delta = 2*round(rand(DIM,lambda))-1;
        X = repmat(x,1,lambda);
        yplus = FUN(X + ck.*delta);
        yminus = FUN(X - ck.*delta);
        Gk = mean(repmat((yplus-yminus),DIM,1)./(2*ck.*delta),2);

        % update objectVector
        x = x - ak*Gk;

        % termination criterions
        fit = FUN(x);
        % stop if target or maxfunevals is reached
        if fit <= ftarget ||  feval(FUN, 'evaluations') >= maxfunevals
            termvalue = 1;
            break;
        end
        % stop if maxGenerations or fit is larger 1e30 (probably divergent
        % run)
        if k > maxGenerations  || fit > 1e30
            termvalue = 0;
            break;
        end
        % stop if x has nan or inf entries
        if max(isnan(x)) == 1 || max(isinf(x)) == 1
            termvalue = 0;
            break;
        end

        % increase k
        k = k + 1;

    end % of while loop

end % of function
```

Δf	f_1 **in 5-D**, N=15, mFE=503					f_1 **in 20-D**, N=15, mFE=2015				
	#	ERT	10%	90%	$\mathrm{RT}_{\mathrm{succ}}$	#	ERT	10%	90%	$\mathrm{RT}_{\mathrm{succ}}$
10	13	2.9e2	2.4e2	3.5e2	2.6e2	12	9.4e2	6.7e2	1.2e3	6.8e2
1	6	1.0e3	8.7e2	1.1e3	3.8e2	11	1.4e3	1.0e3	1.7e3	9.4e2
1e−1	5	1.3e3	1.2e3	1.4e3	3.8e2	10	1.8e3	1.4e3	2.1e3	1.2e3
1e−3	2	3.6e3	3.5e3	3.8e3	4.2e2	9	2.3e3	2.0e3	2.6e3	1.4e3
1e−5	1	7.5e3	7.5e3	7.5e3	4.9e2	8	3.0e3	2.7e3	3.3e3	1.7e3
1e−8	0	*15e−1*	*13e−6*	*14e+0*	5.0e2	4	6.8e3	6.3e3	7.2e3	2.0e3

Δf	f_3 **in 5-D**, N=15, mFE=503					f_3 **in 20-D**, N=15, mFE=2015				
	#	ERT	10%	90%	$\mathrm{RT}_{\mathrm{succ}}$	#	ERT	10%	90%	$\mathrm{RT}_{\mathrm{succ}}$
10	0	*18e+1*	*52e+0*	*60e+1*	4.5e2	0	*43e+1*	*24e+1*	*72e+1*	1.8e3
1	.	.	.	.	.	.	.	.	.	.
1e−1	.	.	.	.	.	.	.	.	.	.
1e−3	.	.	.	.	.	.	.	.	.	.
1e−5	.	.	.	.	.	.	.	.	.	.
1e−8	.	.	.	.	.	.	.	.	.	.

Δf	f_5 **in 5-D**, N=15, mFE=503					f_5 **in 20-D**, N=15, mFE=2015				
	#	ERT	10%	90%	$\mathrm{RT}_{\mathrm{succ}}$	#	ERT	10%	90%	$\mathrm{RT}_{\mathrm{succ}}$
10	10	5.2e2	4.3e2	6.0e2	3.2e2	13	1.2e3	1.0e3	1.5e3	1.0e3
1	2	3.4e3	3.1e3	3.8e3	5.0e2	8	2.7e3	2.3e3	3.1e3	1.1e3
1e−1	2	3.5e3	3.2e3	3.8e3	5.0e2	8	2.7e3	2.3e3	3.1e3	1.1e3
1e−3	2	3.5e3	3.3e3	3.8e3	5.0e2	8	2.8e3	2.3e3	3.1e3	1.2e3
1e−5	2	3.5e3	3.2e3	3.8e3	5.0e2	8	2.8e3	2.3e3	3.1e3	1.2e3
1e−8	2	3.5e3	3.2e3	3.8e3	5.0e2	8	2.8e3	2.4e3	3.1e3	1.2e3

Δf	f_7 **in 5-D**, N=15, mFE=533					f_7 **in 20-D**, N=15, mFE=2015				
	#	ERT	10%	90%	$\mathrm{RT}_{\mathrm{succ}}$	#	ERT	10%	90%	$\mathrm{RT}_{\mathrm{succ}}$
10	2	3.5e3	3.1e3	3.9e3	5.2e2	0	*17e+2*	*61e+1*	*28e+2*	7.9e1
1	0	*64e+0*	*64e−1*	*32e+1*	7.9e1	.	.	.	.	.
1e−1	.	.	.	.	.	.	.	.	.	.
1e−3	.	.	.	.	.	.	.	.	.	.
1e−5	.	.	.	.	.	.	.	.	.	.
1e−8	.	.	.	.	.	.	.	.	.	.

Δf	f_9 **in 5-D**, N=15, mFE=503					f_9 **in 20-D**, N=15, mFE=2015				
	#	ERT	10%	90%	$\mathrm{RT}_{\mathrm{succ}}$	#	ERT	10%	90%	$\mathrm{RT}_{\mathrm{succ}}$
10	1	7.5e3	7.5e3	7.5e3	5.0e2	0	*43e+1*	*28e+1*	*13e+2*	2.0e3
1	0	*54e+1*	*13e+0*	*78e+2*	5.0e2	.	.	.	.	.
1e−1	.	.	.	.	.	.	.	.	.	.
1e−3	.	.	.	.	.	.	.	.	.	.
1e−5	.	.	.	.	.	.	.	.	.	.
1e−8	.	.	.	.	.	.	.	.	.	.

Δf	f_{11} **in 5-D**, N=15, mFE=503					f_{11} **in 20-D**, N=15, mFE=2015				
	#	ERT	10%	90%	$\mathrm{RT}_{\mathrm{succ}}$	#	ERT	10%	90%	$\mathrm{RT}_{\mathrm{succ}}$
10	1	7.5e3	7.5e3	7.5e3	5.0e2	0	*24e+1*	*18e+1*	*39e+2*	1.6e3
1	0	*87e+0*	*14e+0*	*12e+4*	3.2e2	.	.	.	.	.
1e−1	.	.	.	.	.	.	.	.	.	.
1e−3	.	.	.	.	.	.	.	.	.	.
1e−5	.	.	.	.	.	.	.	.	.	.
1e−8	.	.	.	.	.	.	.	.	.	.

Δf	f_{13} **in 5-D**, N=15, mFE=503					f_{13} **in 20-D**, N=15, mFE=2015				
	#	ERT	10%	90%	$\mathrm{RT}_{\mathrm{succ}}$	#	ERT	10%	90%	$\mathrm{RT}_{\mathrm{succ}}$
10	0	*10e+1*	*23e+0*	*16e+1*	4.5e2	0	*30e+1*	*11e+1*	*13e+2*	1.6e3
1	.	.	.	.	.	.	.	.	.	.
1e−1	.	.	.	.	.	.	.	.	.	.
1e−3	.	.	.	.	.	.	.	.	.	.
1e−5	.	.	.	.	.	.	.	.	.	.
1e−8	.	.	.	.	.	.	.	.	.	.

Δf	f_{15} **in 5-D**, N=15, mFE=503					f_{15} **in 20-D**, N=15, mFE=2015				
	#	ERT	10%	90%	$\mathrm{RT}_{\mathrm{succ}}$	#	ERT	10%	90%	$\mathrm{RT}_{\mathrm{succ}}$
10	0	*94e+0*	*44e+0*	*36e+1*	4.0e2	0	*41e+1*	*34e+1*	*60e+1*	1.8e3
1	.	.	.	.	.	.	.	.	.	.
1e−1	.	.	.	.	.	.	.	.	.	.
1e−3	.	.	.	.	.	.	.	.	.	.
1e−5	.	.	.	.	.	.	.	.	.	.
1e−8	.	.	.	.	.	.	.	.	.	.

Δf	f_{17} **in 5-D**, N=15, mFE=503					f_{17} **in 20-D**, N=15, mFE=2015				
	#	ERT	10%	90%	$\mathrm{RT}_{\mathrm{succ}}$	#	ERT	10%	90%	$\mathrm{RT}_{\mathrm{succ}}$
10	9	4.8e2	3.6e2	5.9e2	2.6e2	5	5.2e3	4.7e3	5.6e3	1.5e3
1	0	*93e−1*	*37e−1*	*18e+0*	4.0e2	0	*10e+0*	*76e−1*	*25e+0*	1.3e3
1e−1	.	.	.	.	.	.	.	.	.	.
1e−3	.	.	.	.	.	.	.	.	.	.
1e−5	.	.	.	.	.	.	.	.	.	.
1e−8	.	.	.	.	.	.	.	.	.	.

Δf	f_{19} **in 5-D**, N=15, mFE=503					f_{19} **in 20-D**, N=15, mFE=2015				
	#	ERT	10%	90%	$\mathrm{RT}_{\mathrm{succ}}$	#	ERT	10%	90%	$\mathrm{RT}_{\mathrm{succ}}$
10	5	1.0e3	8.1e2	1.2e3	2.2e2	1	2.9e4	2.8e4	3.0e4	2.0e3
1	0	*13e+0*	*50e−1*	*23e+0*	7.9e1	0	*26e+0*	*12e+0*	*36e+0*	3.1e1
1e−1	.	.	.	.	.	.	.	.	.	.
1e−3	.	.	.	.	.	.	.	.	.	.
1e−5	.	.	.	.	.	.	.	.	.	.
1e−8	.	.	.	.	.	.	.	.	.	.

Δf	f_{21} **in 5-D**, N=15, mFE=503					f_{21} **in 20-D**, N=15, mFE=2015				
	#	ERT	10%	90%	$\mathrm{RT}_{\mathrm{succ}}$	#	ERT	10%	90%	$\mathrm{RT}_{\mathrm{succ}}$
10	10	4.2e2	3.3e2	5.1e2	2.6e2	0	*51e+0*	*32e+0*	*81e+0*	3.5e2
1	1	7.4e3	7.2e3	7.5e3	5.0e2	.	.	.	.	.
1e−1	0	*74e−1*	*11e−1*	*36e+0*	5.0e2	.	.	.	.	.
1e−3	.	.	.	.	.	.	.	.	.	.
1e−5	.	.	.	.	.	.	.	.	.	.
1e−8	.	.	.	.	.	.	.	.	.	.

Δf	f_{23} **in 5-D**, N=15, mFE=503					f_{23} **in 20-D**, N=15, mFE=2015				
	#	ERT	10%	90%	$\mathrm{RT}_{\mathrm{succ}}$	#	ERT	10%	90%	$\mathrm{RT}_{\mathrm{succ}}$
10	15	1.0e2	8.5e1	1.2e2	1.0e2	12	5.3e2	2.0e2	8.7e2	3.6e2
1	0	*46e−1*	*23e−1*	*74e−1*	2.8e2	0	*68e−1*	*45e−1*	*13e+0*	3.9e1
1e−1	.	.	.	.	.	.	.	.	.	.
1e−3	.	.	.	.	.	.	.	.	.	.
1e−5	.	.	.	.	.	.	.	.	.	.
1e−8	.	.	.	.	.	.	.	.	.	.

Δf	f_2 **in 5-D**, N=15, mFE=503					f_2 **in 20-D**, N=15, mFE=2015				
	#	ERT	10%	90%	$\mathrm{RT}_{\mathrm{succ}}$	#	ERT	10%	90%	$\mathrm{RT}_{\mathrm{succ}}$
10	0	*92e+3*	*72e+2*	*19e+5*	5.0e2	0	*13e+4*	*59e+3*	*59e+4*	2.0e3
1	.	.	.	.	.	.	.	.	.	.
1e−1	.	.	.	.	.	.	.	.	.	.
1e−3	.	.	.	.	.	.	.	.	.	.
1e−5	.	.	.	.	.	.	.	.	.	.
1e−8	.	.	.	.	.	.	.	.	.	.

Δf	f_4 **in 5-D**, N=15, mFE=503					f_4 **in 20-D**, N=15, mFE=2015				
	#	ERT	10%	90%	$\mathrm{RT}_{\mathrm{succ}}$	#	ERT	10%	90%	$\mathrm{RT}_{\mathrm{succ}}$
10	0	*17e+1*	*65e+0*	*40e+1*	4.5e2	0	*64e+1*	*37e+1*	*10e+2*	1.6e3
1	.	.	.	.	.	.	.	.	.	.
1e−1	.	.	.	.	.	.	.	.	.	.
1e−3	.	.	.	.	.	.	.	.	.	.
1e−5	.	.	.	.	.	.	.	.	.	.
1e−8	.	.	.	.	.	.	.	.	.	.

Δf	f_6 **in 5-D**, N=15, mFE=503					f_6 **in 20-D**, N=15, mFE=2015				
	#	ERT	10%	90%	$\mathrm{RT}_{\mathrm{succ}}$	#	ERT	10%	90%	$\mathrm{RT}_{\mathrm{succ}}$
10	1	7.4e3	7.2e3	7.5e3	5.0e2	0	*30e+1*	*13e+1*	*34e+3*	2.0e3
1	0	*78e+1*	*21e+0*	*37e+3*	5.0e2	.	.	.	.	.
1e−1	.	.	.	.	.	.	.	.	.	.
1e−3	.	.	.	.	.	.	.	.	.	.
1e−5	.	.	.	.	.	.	.	.	.	.
1e−8	.	.	.	.	.	.	.	.	.	.

Δf	f_8 **in 5-D**, N=15, mFE=503					f_8 **in 20-D**, N=15, mFE=2015				
	#	ERT	10%	90%	$\mathrm{RT}_{\mathrm{succ}}$	#	ERT	10%	90%	$\mathrm{RT}_{\mathrm{succ}}$
10	0	*14e+3*	*67e+1*	*29e+3*	5.0e2	0	*14e+2*	*40e+1*	*43e+2*	2.0e3
1	.	.	.	.	.	.	.	.	.	.
1e−1	.	.	.	.	.	.	.	.	.	.
1e−3	.	.	.	.	.	.	.	.	.	.
1e−5	.	.	.	.	.	.	.	.	.	.
1e−8	.	.	.	.	.	.	.	.	.	.

Δf	f_{10} **in 5-D**, N=15, mFE=503					f_{10} **in 20-D**, N=15, mFE=2015				
	#	ERT	10%	90%	$\mathrm{RT}_{\mathrm{succ}}$	#	ERT	10%	90%	$\mathrm{RT}_{\mathrm{succ}}$
10	0	*23e+4*	*26e+3*	*21e+5*	5.0e2	0	*12e+4*	*56e+3*	*64e+4*	2.0e3
1	.	.	.	.	.	.	.	.	.	.
1e−1	.	.	.	.	.	.	.	.	.	.
1e−3	.	.	.	.	.	.	.	.	.	.
1e−5	.	.	.	.	.	.	.	.	.	.
1e−8	.	.	.	.	.	.	.	.	.	.

Δf	f_{12} **in 5-D**, N=15, mFE=503					f_{12} **in 20-D**, N=15, mFE=2015				
	#	ERT	10%	90%	$\mathrm{RT}_{\mathrm{succ}}$	#	ERT	10%	90%	$\mathrm{RT}_{\mathrm{succ}}$
10	0	*99e+5*	*40e+1*	*11e+7*	5.0e2	2	1.5e4	1.5e4	1.5e4	2.0e3
1	.	.	.	.	.	0	*14e+6*	*43e−1*	*23e+7*	2.0e3
1e−1	.	.	.	.	.	.	.	.	.	.
1e−3	.	.	.	.	.	.	.	.	.	.
1e−5	.	.	.	.	.	.	.	.	.	.
1e−8	.	.	.	.	.	.	.	.	.	.

Δf	f_{14} **in 5-D**, N=15, mFE=503					f_{14} **in 20-D**, N=15, mFE=2015				
	#	ERT	10%	90%	$\mathrm{RT}_{\mathrm{succ}}$	#	ERT	10%	90%	$\mathrm{RT}_{\mathrm{succ}}$
10	10	4.3e2	3.3e2	5.2e2	3.9e2	13	1.3e3	1.0e3	1.5e3	1.2e3
1	2	3.5e3	3.3e3	3.8e3	5.0e2	4	6.8e3	6.3e3	7.3e3	1.8e3
1e−1	1	7.2e3	6.8e3	7.5e3	5.0e2	3	9.3e3	8.8e3	9.8e3	1.8e3
1e−3	0	*80e−1*	*97e−2*	*32e+0*	5.0e2	0	*56e−1*	*32e−3*	*21e+0*	2.0e3
1e−5	.	.	.	.	.	.	.	.	.	.
1e−8	.	.	.	.	.	.	.	.	.	.

Δf	f_{16} **in 5-D**, N=15, mFE=503					f_{16} **in 20-D**, N=15, mFE=2015				
	#	ERT	10%	90%	$\mathrm{RT}_{\mathrm{succ}}$	#	ERT	10%	90%	$\mathrm{RT}_{\mathrm{succ}}$
10	1	7.5e3	7.4e3	7.5e3	5.0e2	0	*46e+0*	*30e+0*	*66e+0*	2.0e2
1	0	*22e+0*	*14e+0*	*46e+0*	2.2e2	.	.	.	.	.
1e−1	.	.	.	.	.	.	.	.	.	.
1e−3	.	.	.	.	.	.	.	.	.	.
1e−5	.	.	.	.	.	.	.	.	.	.
1e−8	.	.	.	.	.	.	.	.	.	.

Δf	f_{18} **in 5-D**, N=15, mFE=503					f_{18} **in 20-D**, N=15, mFE=2015				
	#	ERT	10%	90%	$\mathrm{RT}_{\mathrm{succ}}$	#	ERT	10%	90%	$\mathrm{RT}_{\mathrm{succ}}$
10	0	*31e+0*	*16e+0*	*16e+1*	3.2e2	0	*44e+0*	*33e+0*	*68e+0*	1.4e3
1	.	.	.	.	.	.	.	.	.	.
1e−1	.	.	.	.	.	.	.	.	.	.
1e−3	.	.	.	.	.	.	.	.	.	.
1e−5	.	.	.	.	.	.	.	.	.	.
1e−8	.	.	.	.	.	.	.	.	.	.

Δf	f_{20} **in 5-D**, N=15, mFE=503					f_{20} **in 20-D**, N=15, mFE=2015				
	#	ERT	10%	90%	$\mathrm{RT}_{\mathrm{succ}}$	#	ERT	10%	90%	$\mathrm{RT}_{\mathrm{succ}}$
10	2	3.4e3	3.0e3	3.6e3	5.0e2	9	2.3e3	2.0e3	2.7e3	1.6e3
1	0	*48e+1*	*36e−1*	*12e+3*	5.0e2	0	*52e−1*	*35e−1*	*38e+1*	2.0e3
1e−1	.	.	.	.	.	.	.	.	.	.
1e−3	.	.	.	.	.	.	.	.	.	.
1e−5	.	.	.	.	.	.	.	.	.	.
1e−8	.	.	.	.	.	.	.	.	.	.

Δf	f_{22} **in 5-D**, N=15, mFE=503					f_{22} **in 20-D**, N=15, mFE=2015				
	#	ERT	10%	90%	$\mathrm{RT}_{\mathrm{succ}}$	#	ERT	10%	90%	$\mathrm{RT}_{\mathrm{succ}}$
10	5	1.2e3	1.0e3	1.4e3	5.0e2	1	2.9e4	2.8e4	3.0e4	2.0e3
1	0	*18e+0*	*21e−1*	*67e+0*	5.0e2	1	2.9e4	2.8e4	3.0e4	2.0e3
1e−1	.	.	.	.	.	0	*72e+0*	*11e+0*	*86e+0*	1.2e2
1e−3	.	.	.	.	.	.	.	.	.	.
1e−5	.	.	.	.	.	.	.	.	.	.
1e−8	.	.	.	.	.	.	.	.	.	.

Δf	f_{24} **in 5-D**, N=15, mFE=503					f_{24} **in 20-D**, N=15, mFE=2015				
	#	ERT	10%	90%	$\mathrm{RT}_{\mathrm{succ}}$	#	ERT	10%	90%	$\mathrm{RT}_{\mathrm{succ}}$
10	0	*80e+0*	*54e+0*	*99e+0*	2.5e2	0	*46e+1*	*36e+1*	*55e+1*	1.1e3
1	.	.	.	.	.	.	.	.	.	.
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 $f_{\mathrm{opt}}+\Delta f$ (ERT, see Figure 4); 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 ($\mathrm{RT}_{\mathrm{succ}}$). If $f_{\mathrm{opt}}+\Delta f$ 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 4 for the names of functions.

Figure 2: Determination of the initial values for the gains

```
function [a0,c0] = DetermineParameter(A,alpha,x,step,FUN,DIM)

    % generate matrix with trial vectors
    X = repmat(x,1,10);

    % c0
    c0 = 1e-3;

    % a0
    % generation of the simultaneous perturbation vector
    delta = 2*round(rand(DIM,10))-1;

    % function evaluation
    yplus = FUN(X + c0.*delta);
    yminus = FUN(X - c0.*delta);

    % gradient approximation
    gApprox = mean(repmat((yplus-yminus),DIM,1)./(2*c0.*delta),2);

    % mean of the magnitude of gradient element
    gMeanElement = abs(mean(gApprox));

    % determine parameter a
    a0 = step*(1+A)^alpha/gMeanElement;

end % of function
```

Acknowledgments

The author would like to acknowledge the great work of the BBOB team with particular kudos the Anne Auger, Nikolaus Hansen and Raymond Ros. This work was supported by the Austrian Science Fund (FWF) under grant P19069-N18.

7. 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. C. Spall. Multivariate Stochastic Approximation Using a Simultaneous Perturbation Gradient Approximation. *IEEE Transactions on Automatic Control*, 37(3):332–341, March 1992.

[5] J. C. Spall. An Overview of the Simultaneous Perturbation Method for Efficient Optimization. *Johns Hopkins APL Technical Digest*, 19:482–492, 1998.

[6] J. C. Spall. *Introduction to Stochastic Search and Optimization*. John Wiley & Sons, Hoboken, NJ, 2003.

Figure 3: Multistart procedure for the SPSA

```
function x = spsa(FUN, DIM, ftarget, maxfunevals)

    % make sure to terminate
    if isinf(maxfunevals) || maxfunevals > 1e5*DIM
        kmax = 1e5*DIM;
    else
        kmax = maxfunevals;
    end

    % constant parameter
    A = 0.1*kmax;   % A approx 10% of max generations
    gamma = 0.101;  % reduction rate for ck (as recommended)
    alpha = 0.602;  % reduction rate for ak (as recommended)

    % multistart such that ftarget is reached with reasonable prob.
    for ilaunch = 1:100  % relaunch optimizer up to 100 times

        % restarts
        if ilaunch == 1                    % initial scenario
           xstart = 8 * rand(DIM, 1) - 4;  % random start solution
           step = 0.1;                     % parameter to determine a0
           lambda = 10;                    % number of gradient approximations
           [a0,c0] = DetermineParameter(A,alpha,xstart,step,FUN,DIM);

        else

            choice = round(3*rand) + 1;

            % if the xstart is changing, parameter a0 has to be newly
            % calcualeted

            switch choice

                case 1     % new point
                    xstart = 8 * rand(DIM, 1) - 4;
                    [a0,c0] = DetermineParameter(A,alpha,xstart,step,FUN,DIM);

                case 2     % improve old point
                    xstart = x;
                    [a0,c0] = DetermineParameter(A,alpha,xstart,step,FUN,DIM);

                case 3     % half the step size
                    step = step/2;
                    [a0,c0] = DetermineParameter(A,alpha,xstart,step,FUN,DIM);

                case 4     % increase lambda
                    lambda = ceil(lambda * sqrt(2));

            end % switch case

        end

        % try spsa
        parameter = [a0,alpha,c0,gamma,A,lambda];
        [x,termvalue] = alg(FUN,xstart,parameter,kmax,ftarget,DIM,maxfunevals);

        if termvalue == 1
            break;
        end

    end

end % of function
```

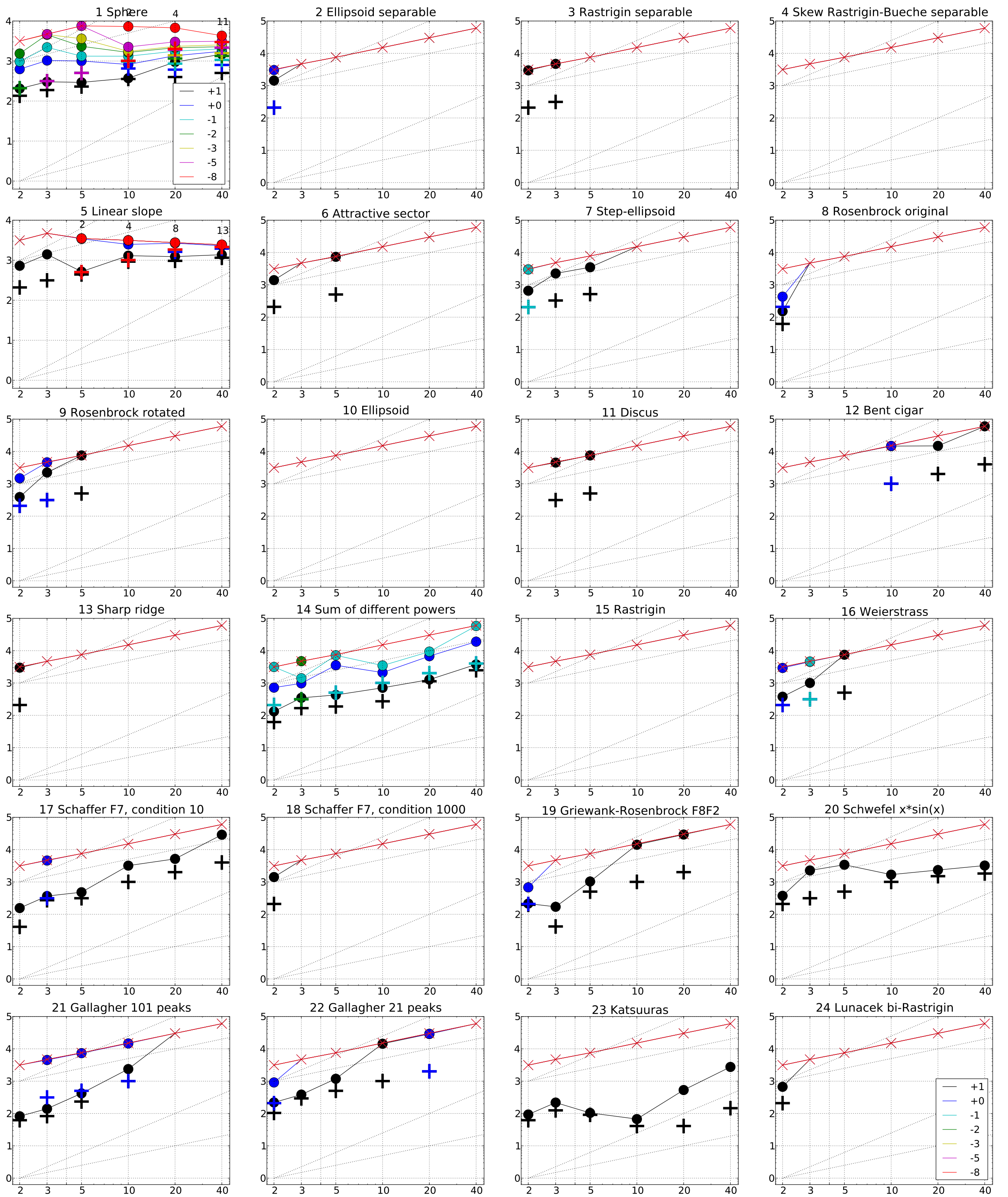

Figure 4: Expected Running Time (ERT, ●) to reach $f_{\rm opt}+\Delta f$ and median number of function evaluations of successful trials (+), shown for $\Delta f = 10, 1, 10^{-1}, 10^{-2}, 10^{-3}, 10^{-5}, 10^{-8}$ (the exponent is given in the legend of f_1 and f_{24}) versus dimension in log-log presentation. 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 (×) 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.

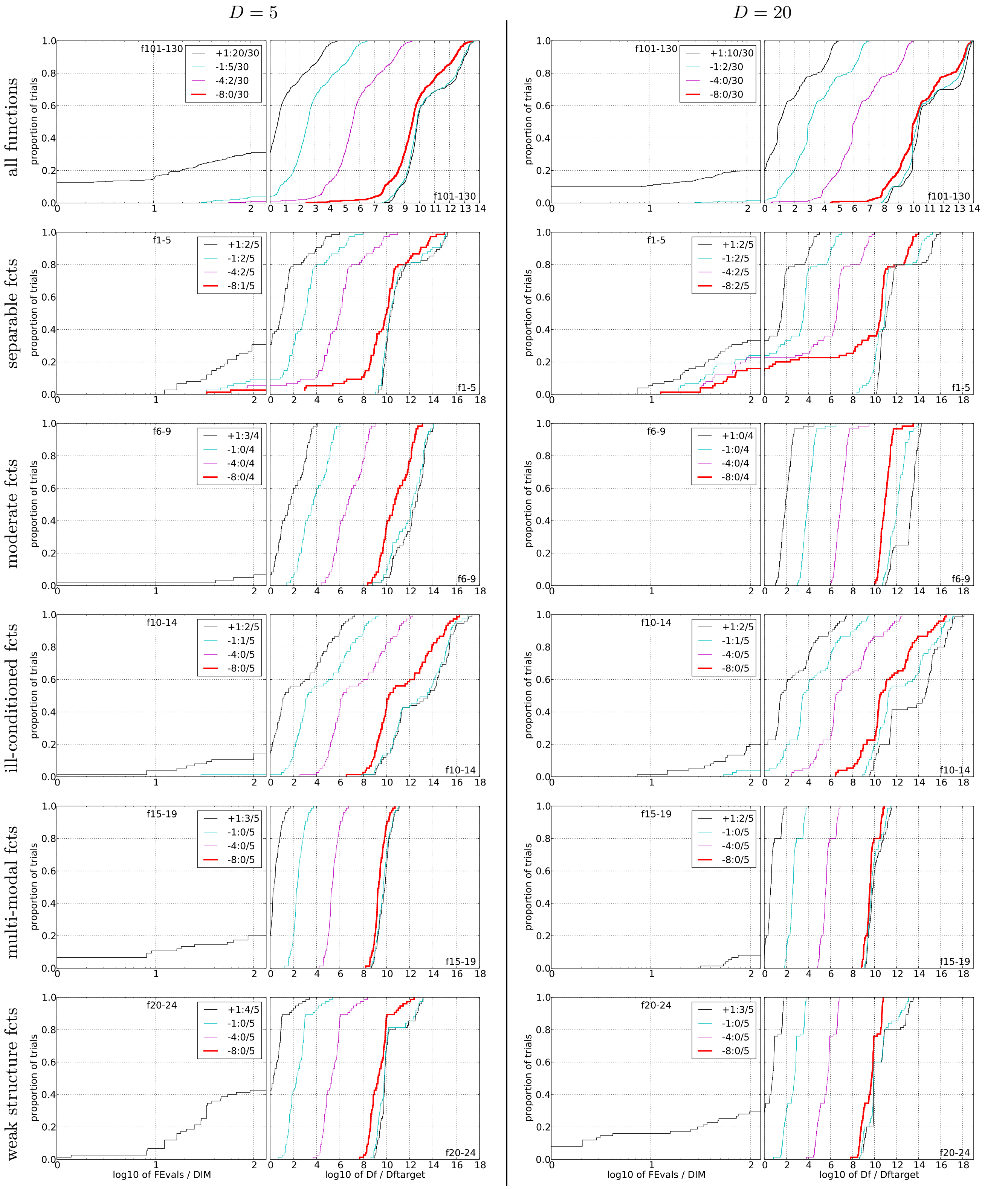

Figure 5: Empirical cumulative distribution functions (ECDFs), plotting the fraction of trials versus running time (left subplots) or versus Δf (right subplots). The thick red line represents the best achieved results. Left subplots: ECDF of the running time (number of function evaluations), divided by search space dimension D, to fall below $f_{\mathrm{opt}} + \Delta f$ with $\Delta f = 10^k$, where k is the first value in the legend. Right subplots: ECDF of the best achieved Δf divided by 10^k (upper left lines in continuation of the left subplot), and best achieved Δf divided by 10^{-8} for running times of $D, 10\,D, 100\,D \ldots$ function evaluations (from right to left cycling black-cyan-magenta). 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.