SPSA with Hessian 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) with Hessian approximation [4] on the BBOB 2009 noise-free testbed. The algorithm is an extension to the widely used SPSA algorithm.

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 presented algorithm is an extension to the basic SPSA algorithm. To simultaneous iteration of the Hessian should increase the perfromance of the algorithm.

2. ALGORITHM PRESENTATION

In Fig. 1 the main algorithm is presented.

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. The maximal number of restarts is 100 and each run performs maximal $1e5 \times$ DIM iterations. 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 2 and 3 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 $8.0; 8.2; 8.2; 8.5; 16$ and 22×10^{-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 not negligible.

6. CONCLUSION

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

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. Feedback and Weighting Mechanisms for Improving Jacobian Estimates in the Adaptive Simultaneous Perturbation Algorithm. *IEEE Transactions on Automatic Control*, 54, 2009.

Figure 1: SPSA with Hessian approximation in Matlab

```
% SPSA2 with Feedback and Weighting Mechanism for BBOB Workshop

function SPSA2(FUN, DIM, ftarget, maxfunevals)

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

        if ilaunch == 1                    % initial scenario
           xstart = 8 * rand(DIM, 1) - 4;
           lambda = 1;
        else

            choice = round(2*rand) + 1;

            switch choice

                case 1     % new point
                    xstart = 8 * rand(DIM, 1) - 4;

                case 2     % improve old point
                    if max(abs(x)) < 5
                        xstart = x;
                    else
                        xstart = 8 * rand(DIM, 1) - 4;
                    end

                case 3     % increase lambda
                    lambda = ceil(lambda * sqrt(2));

            end % switch case

        end

        % try spsa
        [x,termvalue] = alg(FUN,xstart, DIM,ftarget,maxfunevals,lambda);

        if termvalue == 1
            break;
        end

    end

end % of function

function [x,termvalue] = alg(FUN,x, DIM, ftarget, maxfunevals,lambda)

    % initialize parameter
    alpha = 0.602;
    gamma = 0.101;
    if isinf(maxfunevals)
        kmax = 1e5*DIM;
    else
        kmax = maxfunevals/4/lambda;
    end

    A = kmax*0.1;

    % initialize counters
    k = 0; % iteration counter

    % initialize hessian matrix and sum of loss measurements
    HkBar = eye(DIM);
    HkBarBar = zeros(DIM);
    Gk = zeros(DIM,1);
    dGk = zeros(DIM,1);
    Psik = zeros(DIM);
    sumck = 0;
    HkHat = zeros(DIM);

    % determine initial parameter
    % a0
    a0 = 1;
    X = repmat(x,1,10);
    % c0
    dummy = feval(FUN,X);
    c0 = max([5*std(dummy,1),1e-5]);
    c0Bar = 2*c0;

    while k < kmax

        % gain sequences
        ck = c0*(k+1)^(-gamma);
        ckBar = c0Bar*(k+1)^(-gamma);
        sumck = sumck + ck^2*ckBar^2;
        ak = a0*(k + 1 + A)^(-alpha);

        % gradient and hessian approximation
        % generation of the simultaneous perturbation vector
        for i = 1:lambda

            delta = 2*round(rand(DIM,1))-1;    % for gradient recursion
            deltaH = 2*round(rand(DIM,1))-1;   % for hessian recursion

            % function evaluation
            yplus = FUN(x + ck.*delta);
            yminus = FUN(x - ck.*delta);
            yplusH = FUN(x + ck.*delta + ckBar.*deltaH);
            yminusH = FUN(x - ck.*delta + ckBar.*deltaH);

            % gradient approximation
            Gk = (i-1)/i*Gk + 1/i*(yplus-yminus)./(2*ck*delta);

            % gradient approximation for hessian matrix
            dGk = (i-1)/i*dGk + 1/i*((yplusH-yplus)./(ckBar.*deltaH) - ...
                  (yminusH-yminus)./(ckBar.*deltaH));

            hhat = dGk./(2*ck)*(delta.^(-1))';
            HkHat = (i-1)/i*HkHat + 1/(2*i)*(hhat + hhat');

            % feedback term
            Dk = delta*(1./delta)' - eye(DIM);
            DkBar = deltaH*(1./deltaH)' - eye(DIM);
            psik = DkBar'*HkBarBar*Dk + DkBar'*HkBarBar+HkBarBar*Dk;
            Psik = (i-1)/i*Psik + 1/(2*i)*(psik + psik');
        end

        % weights
        wk = 1/(k+1)^(0.501);   % for noise free settimgs

        % hessian matrix recursion
        HkBar = (1-wk)*HkBar + wk*(HkHat - Psik);

        % hessian matrix for update (must be positiv definite)
        try
            HkBarBar = sqrtm(HkBar*HkBar + 1e-5*exp(-k+1)*eye(DIM));
        catch
            HkBarBar = diag(diag(HkBar + 1e-3*exp(-k+1)*eye(DIM)));
        end

        if cond(HkBarBar) > 1e10 || max(max(abs(imag(HkBarBar)))) >= 0
            HkBarBar = diag(diag(HkBar + 1e-3*exp(-k+1)*eye(DIM)));
        end

        % update of the search point
        xnew = x - ak*(HkBarBar\Gk);

        % blocking
        if max(abs(xnew - x)) < 10
            x = xnew;
        end

        % termination criteria
        fit = feval(FUN,x);
        if max(isnan(x)) == 1 || max(isinf(x)) == 1 || fit > 1e30
            termvalue = 0;
            break;
        end

        if feval(FUN, 'fbest') < ftarget || ...
           feval(FUN, 'evaluations') >= maxfunevals
            termvalue = 1;
            break;
        end

        k = k + 1;

    end % of iteration

end % of function
```

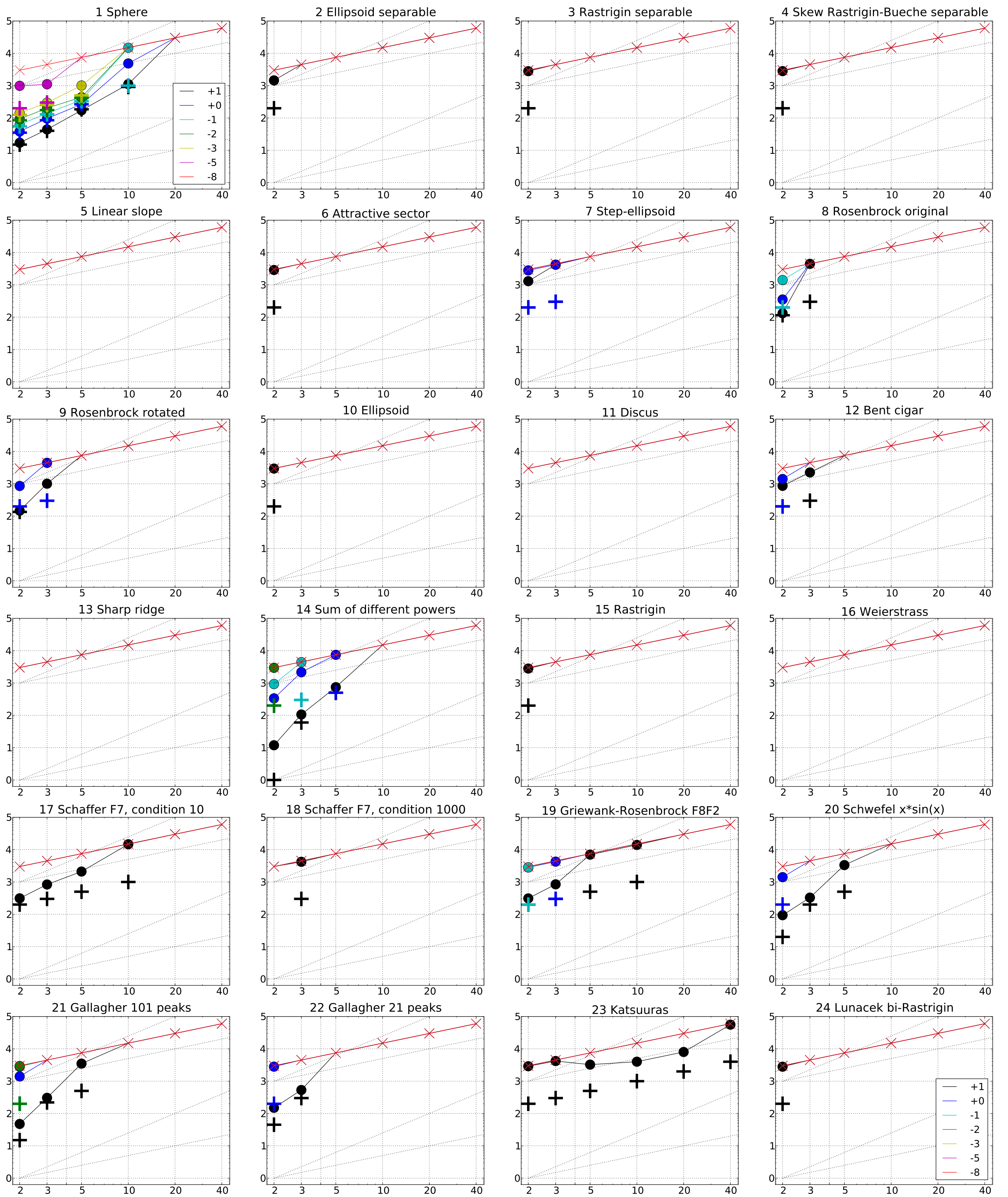

Figure 2: Expected Running Time (ERT, ●) **to reach** $f_{\mathrm{opt}} + \Delta f$ **and median number of function evaluations of successful trials** (+), **shown for** $\Delta f = 10, 1, 10^{-1}, 10^{-2}, 10^{-3}, 10^{-5}, 10^{-8}$ **(the exponent is given in the legend of** f_1 **and** f_{24}**) versus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

	***f*1 in 5-D**, N=15, mFE=500					***f*1 in 20-D**, N=15, mFE=2000				
Δf	#	ERT	10%	90%	$\mathrm{RT_{succ}}$	#	ERT	10%	90%	$\mathrm{RT_{succ}}$
10	15	1.7e2	1.5e2	1.9e2	1.7e2	0	*21e+1*	*13e+1*	*27e+1*	2.8e1
1	15	2.7e2	2.5e2	2.8e2	2.7e2	.	.	.	.	.
1e−1	14	3.6e2	3.4e2	3.9e2	3.4e2	.	.	.	.	.
1e−3	7	1.0e3	9.8e2	1.0e3	4.7e2	.	.	.	.	.
1e−5	0	*11e-4*	*52e-6*	*89e-4*	4.5e2	.	.	.	.	.
1e−8	.	.	.	.	.	.	.	.	.	.

	***f*2 in 5-D**, N=15, mFE=500					***f*2 in 20-D**, N=15, mFE=2000				
Δf	#	ERT	10%	90%	$\mathrm{RT_{succ}}$	#	ERT	10%	90%	$\mathrm{RT_{succ}}$
10	0	*40e+4*	*22e+3*	*18e+5*	4.0e2	0	*16e+6*	*36e+5*	*66e+6*	4.4e1
1	.	.	.	.	.	.	.	.	.	.
1e−1	.	.	.	.	.	.	.	.	.	.
1e−3	.	.	.	.	.	.	.	.	.	.
1e−5	.	.	.	.	.	.	.	.	.	.
1e−8	.	.	.	.	.	.	.	.	.	.

	***f*3 in 5-D**, N=15, mFE=500					***f*3 in 20-D**, N=15, mFE=2000				
Δf	#	ERT	10%	90%	$\mathrm{RT_{succ}}$	#	ERT	10%	90%	$\mathrm{RT_{succ}}$
10	0	*23e+1*	*82e+0*	*80e+1*	1.1e2	0	*20e+2*	*90e+1*	*57e+2*	3.5e1
1	.	.	.	.	.	.	.	.	.	.
1e−1	.	.	.	.	.	.	.	.	.	.
1e−3	.	.	.	.	.	.	.	.	.	.
1e−5	.	.	.	.	.	.	.	.	.	.
1e−8	.	.	.	.	.	.	.	.	.	.

	***f*4 in 5-D**, N=15, mFE=500					***f*4 in 20-D**, N=15, mFE=2000				
Δf	#	ERT	10%	90%	$\mathrm{RT_{succ}}$	#	ERT	10%	90%	$\mathrm{RT_{succ}}$
10	0	*29e+1*	*17e+1*	*11e+2*	5.6e1	0	*21e+2*	*11e+2*	*63e+2*	1.7e1
1	.	.	.	.	.	.	.	.	.	.
1e−1	.	.	.	.	.	.	.	.	.	.
1e−3	.	.	.	.	.	.	.	.	.	.
1e−5	.	.	.	.	.	.	.	.	.	.
1e−8	.	.	.	.	.	.	.	.	.	.

	***f*5 in 5-D**, N=15, mFE=500					***f*5 in 20-D**, N=15, mFE=2000				
Δf	#	ERT	10%	90%	$\mathrm{RT_{succ}}$	#	ERT	10%	90%	$\mathrm{RT_{succ}}$
10	0	*11e+1*	*73e+0*	*14e+1*	3.9e1	0	*33e+1*	*24e+1*	*43e+1*	6.3e2
1	.	.	.	.	.	.	.	.	.	.
1e−1	.	.	.	.	.	.	.	.	.	.
1e−3	.	.	.	.	.	.	.	.	.	.
1e−5	.	.	.	.	.	.	.	.	.	.
1e−8	.	.	.	.	.	.	.	.	.	.

	***f*6 in 5-D**, N=15, mFE=500					***f*6 in 20-D**, N=15, mFE=2000				
Δf	#	ERT	10%	90%	$\mathrm{RT_{succ}}$	#	ERT	10%	90%	$\mathrm{RT_{succ}}$
10	0	*16e+4*	*23e+3*	*45e+4*	5.6e1	0	*86e+4*	*35e+4*	*14e+5*	3.9e1
1	.	.	.	.	.	.	.	.	.	.
1e−1	.	.	.	.	.	.	.	.	.	.
1e−3	.	.	.	.	.	.	.	.	.	.
1e−5	.	.	.	.	.	.	.	.	.	.
1e−8	.	.	.	.	.	.	.	.	.	.

	***f*7 in 5-D**, N=15, mFE=500					***f*7 in 20-D**, N=15, mFE=2000				
Δf	#	ERT	10%	90%	$\mathrm{RT_{succ}}$	#	ERT	10%	90%	$\mathrm{RT_{succ}}$
10	0	*32e+1*	*76e+0*	*15e+2*	1.0e0	0	*22e+2*	*14e+2*	*46e+2*	1.0e0
1	.	.	.	.	.	.	.	.	.	.
1e−1	.	.	.	.	.	.	.	.	.	.
1e−3	.	.	.	.	.	.	.	.	.	.
1e−5	.	.	.	.	.	.	.	.	.	.
1e−8	.	.	.	.	.	.	.	.	.	.

	***f*8 in 5-D**, N=15, mFE=500					***f*8 in 20-D**, N=15, mFE=2000				
Δf	#	ERT	10%	90%	$\mathrm{RT_{succ}}$	#	ERT	10%	90%	$\mathrm{RT_{succ}}$
10	0	*16e+3*	*22e+2*	*51e+3*	4.0e2	0	*41e+4*	*17e+4*	*66e+4*	2.2e1
1	.	.	.	.	.	.	.	.	.	.
1e−1	.	.	.	.	.	.	.	.	.	.
1e−3	.	.	.	.	.	.	.	.	.	.
1e−5	.	.	.	.	.	.	.	.	.	.
1e−8	.	.	.	.	.	.	.	.	.	.

	***f*9 in 5-D**, N=15, mFE=500					***f*9 in 20-D**, N=15, mFE=2000				
Δf	#	ERT	10%	90%	$\mathrm{RT_{succ}}$	#	ERT	10%	90%	$\mathrm{RT_{succ}}$
10	0	*23e+2*	*52e+1*	*11e+3*	1.6e2	0	*97e+3*	*35e+3*	*24e+4*	3.5e1
1	.	.	.	.	.	.	.	.	.	.
1e−1	.	.	.	.	.	.	.	.	.	.
1e−3	.	.	.	.	.	.	.	.	.	.
1e−5	.	.	.	.	.	.	.	.	.	.
1e−8	.	.	.	.	.	.	.	.	.	.

	***f*10 in 5-D**, N=15, mFE=500					***f*10 in 20-D**, N=15, mFE=2000				
Δf	#	ERT	10%	90%	$\mathrm{RT_{succ}}$	#	ERT	10%	90%	$\mathrm{RT_{succ}}$
10	0	*52e+4*	*49e+3*	*23e+5*	2.5e2	0	*90e+5*	*50e+5*	*15e+6*	2.5e1
1	.	.	.	.	.	.	.	.	.	.
1e−1	.	.	.	.	.	.	.	.	.	.
1e−3	.	.	.	.	.	.	.	.	.	.
1e−5	.	.	.	.	.	.	.	.	.	.
1e−8	.	.	.	.	.	.	.	.	.	.

	***f*11 in 5-D**, N=15, mFE=500					***f*11 in 20-D**, N=15, mFE=2000				
Δf	#	ERT	10%	90%	$\mathrm{RT_{succ}}$	#	ERT	10%	90%	$\mathrm{RT_{succ}}$
10	0	*74e+1*	*64e+0*	*48e+4*	3.2e2	0	*33e+4*	*20e+3*	*61e+5*	7.9e2
1	.	.	.	.	.	.	.	.	.	.
1e−1	.	.	.	.	.	.	.	.	.	.
1e−3	.	.	.	.	.	.	.	.	.	.
1e−5	.	.	.	.	.	.	.	.	.	.
1e−8	.	.	.	.	.	.	.	.	.	.

	***f*12 in 5-D**, N=15, mFE=500					***f*12 in 20-D**, N=15, mFE=2000				
Δf	#	ERT	10%	90%	$\mathrm{RT_{succ}}$	#	ERT	10%	90%	$\mathrm{RT_{succ}}$
10	0	*21e+6*	*29e+3*	*30e+7*	1.2e2	0	*12e+8*	*27e+7*	*43e+8*	3.5e1
1	.	.	.	.	.	.	.	.	.	.
1e−1	.	.	.	.	.	.	.	.	.	.
1e−3	.	.	.	.	.	.	.	.	.	.
1e−5	.	.	.	.	.	.	.	.	.	.
1e−8	.	.	.	.	.	.	.	.	.	.

	***f*13 in 5-D**, N=15, mFE=500					***f*13 in 20-D**, N=15, mFE=2000				
Δf	#	ERT	10%	90%	$\mathrm{RT_{succ}}$	#	ERT	10%	90%	$\mathrm{RT_{succ}}$
10	0	*72e+1*	*30e+1*	*16e+2*	3.5e1	0	*30e+2*	*23e+2*	*37e+2*	1.2e1
1	.	.	.	.	.	.	.	.	.	.
1e−1	.	.	.	.	.	.	.	.	.	.
1e−3	.	.	.	.	.	.	.	.	.	.
1e−5	.	.	.	.	.	.	.	.	.	.
1e−8	.	.	.	.	.	.	.	.	.	.

	***f*14 in 5-D**, N=15, mFE=500					***f*14 in 20-D**, N=15, mFE=2000				
Δf	#	ERT	10%	90%	$\mathrm{RT_{succ}}$	#	ERT	10%	90%	$\mathrm{RT_{succ}}$
10	7	7.4e2	6.1e2	8.8e2	3.4e2	0	*13e+1*	*44e+0*	*25e+1*	3.1e1
1	1	7.4e3	7.4e3	7.5e3	5.0e2	.	.	.	.	.
1e−1	0	*11e+0*	*15e-1*	*46e+0*	7.0e1	.	.	.	.	.
1e−3	.	.	.	.	.	.	.	.	.	.
1e−5	.	.	.	.	.	.	.	.	.	.
1e−8	.	.	.	.	.	.	.	.	.	.

	***f*15 in 5-D**, N=15, mFE=500					***f*15 in 20-D**, N=15, mFE=2000				
Δf	#	ERT	10%	90%	$\mathrm{RT_{succ}}$	#	ERT	10%	90%	$\mathrm{RT_{succ}}$
10	0	*22e+1*	*53e+0*	*62e+1*	1.4e2	0	*18e+2*	*12e+2*	*35e+2*	5.0e1
1	.	.	.	.	.	.	.	.	.	.
1e−1	.	.	.	.	.	.	.	.	.	.
1e−3	.	.	.	.	.	.	.	.	.	.
1e−5	.	.	.	.	.	.	.	.	.	.
1e−8	.	.	.	.	.	.	.	.	.	.

	***f*16 in 5-D**, N=15, mFE=500					***f*16 in 20-D**, N=15, mFE=2000				
Δf	#	ERT	10%	90%	$\mathrm{RT_{succ}}$	#	ERT	10%	90%	$\mathrm{RT_{succ}}$
10	0	*12e+1*	*29e+0*	*16e+1*	2.2e2	0	*86e+0*	*64e+0*	*11e+1*	3.5e2
1	.	.	.	.	.	.	.	.	.	.
1e−1	.	.	.	.	.	.	.	.	.	.
1e−3	.	.	.	.	.	.	.	.	.	.
1e−5	.	.	.	.	.	.	.	.	.	.
1e−8	.	.	.	.	.	.	.	.	.	.

	***f*17 in 5-D**, N=15, mFE=500					***f*17 in 20-D**, N=15, mFE=2000				
Δf	#	ERT	10%	90%	$\mathrm{RT_{succ}}$	#	ERT	10%	90%	$\mathrm{RT_{succ}}$
10	3	2.1e3	1.8e3	2.4e3	3.6e2	0	*36e+0*	*20e+0*	*11e+1*	3.9e1
1	0	*13e+0*	*54e-1*	*28e+0*	2.5e2	.	.	.	.	.
1e−1	.	.	.	.	.	.	.	.	.	.
1e−3	.	.	.	.	.	.	.	.	.	.
1e−5	.	.	.	.	.	.	.	.	.	.
1e−8	.	.	.	.	.	.	.	.	.	.

	***f*18 in 5-D**, N=15, mFE=500					***f*18 in 20-D**, N=15, mFE=2000				
Δf	#	ERT	10%	90%	$\mathrm{RT_{succ}}$	#	ERT	10%	90%	$\mathrm{RT_{succ}}$
10	0	*39e+0*	*18e+0*	*16e+1*	2.5e2	0	*10e+1*	*60e+0*	*52e+1*	4.4e1
1	.	.	.	.	.	.	.	.	.	.
1e−1	.	.	.	.	.	.	.	.	.	.
1e−3	.	.	.	.	.	.	.	.	.	.
1e−5	.	.	.	.	.	.	.	.	.	.
1e−8	.	.	.	.	.	.	.	.	.	.

	***f*19 in 5-D**, N=15, mFE=500					***f*19 in 20-D**, N=15, mFE=2000				
Δf	#	ERT	10%	90%	$\mathrm{RT_{succ}}$	#	ERT	10%	90%	$\mathrm{RT_{succ}}$
10	1	7.0e3	6.5e3	7.5e3	5.0e2	0	*30e+0*	*25e+0*	*51e+0*	1.6e2
1	0	*18e+0*	*12e+0*	*58e+0*	1.6e2	.	.	.	.	.
1e−1	.	.	.	.	.	.	.	.	.	.
1e−3	.	.	.	.	.	.	.	.	.	.
1e−5	.	.	.	.	.	.	.	.	.	.
1e−8	.	.	.	.	.	.	.	.	.	.

	***f*20 in 5-D**, N=15, mFE=500					***f*20 in 20-D**, N=15, mFE=2000				
Δf	#	ERT	10%	90%	$\mathrm{RT_{succ}}$	#	ERT	10%	90%	$\mathrm{RT_{succ}}$
10	2	3.3e3	3.0e3	3.8e3	2.7e2	0	*12e+4*	*71e+3*	*18e+4*	2.5e1
1	0	*93e+1*	*30e-1*	*31e+3*	3.2e2	.	.	.	.	.
1e−1	.	.	.	.	.	.	.	.	.	.
1e−3	.	.	.	.	.	.	.	.	.	.
1e−5	.	.	.	.	.	.	.	.	.	.
1e−8	.	.	.	.	.	.	.	.	.	.

	***f*21 in 5-D**, N=15, mFE=500					***f*21 in 20-D**, N=15, mFE=2000				
Δf	#	ERT	10%	90%	$\mathrm{RT_{succ}}$	#	ERT	10%	90%	$\mathrm{RT_{succ}}$
10	2	3.5e3	3.2e3	3.8e3	4.1e2	0	*84e+0*	*76e+0*	*85e+0*	1.5e1
1	0	*44e+0*	*76e-1*	*62e+0*	5.0e1	.	.	.	.	.
1e−1	.	.	.	.	.	.	.	.	.	.
1e−3	.	.	.	.	.	.	.	.	.	.
1e−5	.	.	.	.	.	.	.	.	.	.
1e−8	.	.	.	.	.	.	.	.	.	.

	***f*22 in 5-D**, N=15, mFE=500					***f*22 in 20-D**, N=15, mFE=2000				
Δf	#	ERT	10%	90%	$\mathrm{RT_{succ}}$	#	ERT	10%	90%	$\mathrm{RT_{succ}}$
10	0	*64e+0*	*19e+0*	*77e+0*	5.0e1	0	*85e+0*	*83e+0*	*86e+0*	1.7e1
1	.	.	.	.	.	.	.	.	.	.
1e−1	.	.	.	.	.	.	.	.	.	.
1e−3	.	.	.	.	.	.	.	.	.	.
1e−5	.	.	.	.	.	.	.	.	.	.
1e−8	.	.	.	.	.	.	.	.	.	.

	***f*23 in 5-D**, N=15, mFE=500					***f*23 in 20-D**, N=15, mFE=2000				
Δf	#	ERT	10%	90%	$\mathrm{RT_{succ}}$	#	ERT	10%	90%	$\mathrm{RT_{succ}}$
10	2	3.3e3	2.8e3	3.8e3	2.5e2	3	8.0e3	6.7e3	9.3e3	1.3e3
1	0	*19e+0*	*93e-1*	*32e+0*	1.6e2	0	*12e+0*	*89e-1*	*25e+0*	1.1e2
1e−1	.	.	.	.	.	.	.	.	.	.
1e−3	.	.	.	.	.	.	.	.	.	.
1e−5	.	.	.	.	.	.	.	.	.	.
1e−8	.	.	.	.	.	.	.	.	.	.

	***f*24 in 5-D**, N=15, mFE=500					***f*24 in 20-D**, N=15, mFE=2000				
Δf	#	ERT	10%	90%	$\mathrm{RT_{succ}}$	#	ERT	10%	90%	$\mathrm{RT_{succ}}$
10	0	*93e+0*	*62e+0*	*12e+1*	1.8e2	0	*59e+1*	*46e+1*	*71e+1*	4.4e1
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 $f_{\rm opt}+\Delta f$ (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.

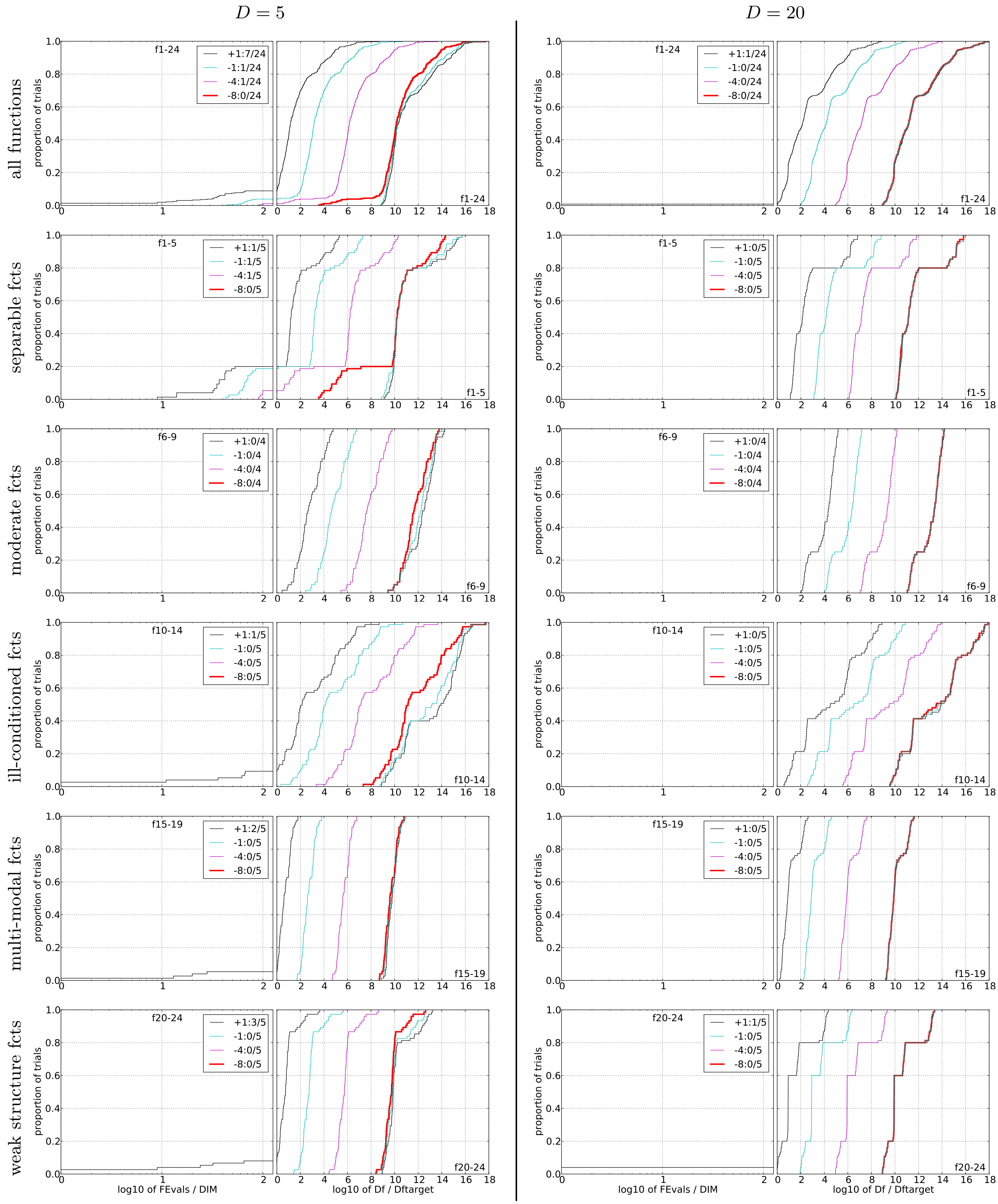

Figure 3: Empirical cumulative distribution functions (ECDFs), plotting the fraction of trials versus running time (left subplots) or versus Δf (right subplots). The thick red line represents the best achieved results. Left subplots: ECDF of the running time (number of function evaluations), divided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