A Stigmergy-Based Algorithm for Black-Box Optimization: Noisy Function Testbed

Peter Korošec
Jožef Stefan Institute
Jamova cesta 39
SI-1000 Ljubljana, Slovenia
peter.korosec@ijs.si

Jurij Šilc
Jožef Stefan Institute
Jamova cesta 39
SI-1000 Ljubljana, Slovenia
jurij.silc@ijs.si

ABSTRACT

In this paper, we present a stigmergy-based algorithm for solving optimization problems with continuous variables, labeled Differential Ant-Stigmergy Algorithm (DASA). The performance of the DASA is evaluated on the set of benchmark problems provided for Black-Box Optimization Benchmarking (BBOB) 2009, a GECCO Workshop for Real-Parameter Optimization. Benchmarking for noisy function testbed is presented.

Categories and Subject Descriptors

G.1.6 [**Numerical Analysis**]: OptimizationGlobal Optimization, Unconstrained Optimization; F.2.1 [**Analysis of Algorithms and Problem Complexity**]: Numerical Algorithms and Problems

General Terms

Algorithms

Keywords

Benchmarking, Black-box optimization, Stigmergy

1. INTRODUCTION

Numerical optimization problems are an important field of research and have a wide number of applications, from the mathematical optimization of functions to the real-world engineering problems. Many engineering problems involve choosing the best configuration of a set of parameters to achieve a specified objective. Numerical optimization refers to the case when these parameters take continuous values, as opposed to combinatorial optimization, which deals with discrete values.

In this paper we consider the following numerical minimization problem:

$$\min(f(\vec{x})), \vec{B}_l \leq \vec{x} \leq \vec{B}_u$$

where $\vec{x} = (x_1, x_2, \ldots, x_D)$ is the variable vector in $\mathbb{R}^D$, $f(\vec{x})$ denotes the cost function to minimize and $\vec{B}_l = (B_{l1}, B_{l2}, \ldots, B_{lD})$, $\vec{B}_u = (B_{u1}, B_{u2}, \ldots, B_{uD})$ represent, respectively, the lower and the upper bound of the variables, such that $x_i \in [B_{li}, B_{ui}]$.

Usually, numerical optimization problems are black-box optimizations, in which the cost function's form as well as its derivatives are unknown. Normally, this occurs when the cost function is computed using a complex simulation about which the optimization algorithm has no information. Executing a black-box simulation in order to evaluate a candidate solution is usually very expensive and can take up to several minutes or even hours. This is particularly problematic because optimization algorithms for black-box problems are necessarily blind search algorithms that must repeatedly sample points in a solution space, evaluate them by running the simulation, and apply various heuristics in order to choose the next points to sample.

In the past two decades, many bio-inspired optimization algorithms have been proposed to solve this kind of optimization problem, e.g., real-parameter genetic algorithm [10], evolution strategies [3], differential evolution [9], particle swarm optimization [7], immunological algorithm [2], ant-colony optimization [1], etc. These algorithms have been used to solve problems in several research fields due to the fact that do not require previous considerations regarding the problem to be optimized and offers a high degree of parallelism.

Although ant-based optimization has been proven to be one of the best metaheuristics in some combinatorial optimization problems, the application to the numerical optimizations appears more challenging, since the pheromone laying method is not straightforward. There are several possibilities to use ants for numerical optimization. We can use simplified direct simulation of real ants' behavior or we can extend method to explore continuous spaces. This extension can be done by the suitable discretization of a search space or by probabilistic sampling. In this way a fine-grained discrete form of continuous domain is created. With it we are able to represent this problem as a graph, which enables the use of ant-based approach for solving numerical optimization problems.

The remainder of this paper is organized as follows: Section 2 introduces the optimization algorithm. Sections 3 and 4 present the experimental procedure and black-box optimization benchmarking for noisy function testbed, respectively. CPU timing experiment is presented in Section 5. Finally, Section 6 concludes the paper.

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

2. STIGMERGY-BASED ALGORITHM

2.1 Parameter Differences

Let x'_i be the current value of the i-th parameter. During the searching for the optimal parameter value, the new value, x_i, is assigned to the i-th parameter as follows:

$$x_i = x'_i + \delta_i. \tag{1}$$

Here, δ_i is the so-called *parameter difference* and is chosen from the set

$$\Delta_i = \Delta_i^- \cup \{0\} \cup \Delta_i^+,$$

where

$$\Delta_i^- = \{\delta_{i,k}^- | \; \delta_{i,k}^- = -b^{k+B_{li}-1}, k = 1, 2, \ldots, d_i\}$$

and

$$\Delta_i^+ = \{\delta_{i,k}^+ | \; \delta_{i,k}^+ = b^{k+B_{li}-1}, k = 1, 2, \ldots, d_i\}.$$

Here, $d_i = B_{ui} - B_{li} + 1$. Therefore, for each parameter x_i, the parameter difference, δ_i, has a range from $b^{B_{li}}$ to $b^{B_{ui}}$, where b is the so-called *discrete base*, $B_{li} = \lfloor \lg_b(\epsilon_i) \rfloor$, and $B_{ui} = \lfloor \lg_b(\max(x_i) - \min(x_i)) \rfloor$. With the parameter ϵ_i, the maximum precision of the parameter x_i is set. The precision is limited by the computer's floating-point arithmetics. To enable a more flexible movement over the search space, the weight ω is added to Eq. 1:

$$x_i = x'_i + \omega\delta_i \tag{2}$$

where $\omega = \mathrm{RandomInteger}(1, b-1)$.

2.2 Graph Representation

From all the sets Δ_i, $1 \le i \le D$, where D represents the number of parameters, the so-called *differential graph* $\mathcal{G} = (V, E)$ with a set of vertices, V, and a set of edges, E, between the vertices is constructed. Each set Δ_i is represented by the set of vertices, $V_i = \{v_{i,1}, v_{i,2}, \ldots, v_{i,2d_i+1}\}$, and $V = \bigcup_{i=1}^{D} V_i$. Then we have that

$$\Delta_i = \{\delta_{i,d_i}^-, \ldots, \delta_{i,d_i-j+1}^-, \ldots, \delta_{i,1}^-, 0, \delta_{i,1}^+, \ldots, \delta_{i,j}^+, \ldots, \delta_{i,d_i}^+\}$$

corresponds to

$$V_i = \{v_{i,1}, \ldots, v_{i,j}, \ldots, v_{i,d_i+1}, \ldots, v_{i,d_i+1+j}, \ldots, v_{i,2d_i+1}\},$$

where

$$\begin{aligned} v_{i,j} &\xrightarrow{\delta} \delta_{i,d_i-j+1}^-, \\ v_{i,d_i+1} &\xrightarrow{\delta} 0, \\ v_{i,d_i+1+j} &\xrightarrow{\delta} \delta_{i,j}^+ \end{aligned}$$

and $j = 1, 2, \ldots, d_i$. Each vertex of the set V_i is connected to all the vertices that belong to the set V_{i+1}. Therefore, this is a directed graph, where each path $\vec{p}$ from the starting vertex, $v_1 \in V_1$, to any of the ending vertices, $v_D \in V_D$, is of equal length and can be defined with v_i as $\nu = (v_1 v_2 \ldots v_i \ldots v_D)$, where $v_i \in V_i$, $1 \le i \le D$.

2.3 Algorithm Implementation

The optimization consists of an iterative improvement of the temporary best solution, $\vec{x}^{\,\mathrm{tb}}$, by constructing an appropriate path $\vec{p}$. New solutions are produced by applying $\vec{p}$ to $\vec{x}^{\,\mathrm{tb}}$ (Eq. 2).

First a solution $\vec{x}^{\,\mathrm{tb}}$ is randomly chosen by uniform sampling and evaluated. Then a search graph is created and an initial amount of pheromone is deposited on search graph according to the Cauchy probability density function

$$C(z) = \frac{1}{s\pi(1 + (\frac{z-l_i}{s})^2)},$$

where l_i is the location offset for the i-th parameter and

$$s = s_{\mathrm{global}} - s_{\mathrm{local}}$$

is the scale factor. For an initial pheromone distribution the Cauchy distribution with $s_{\mathrm{global}} = 10$, $s_{\mathrm{local}} = 0$, and $l_i = 0, i = 1, 2, \ldots, D$ is used and each parameter vertices are equidistantly arranged between $z = [-4, 4]$.

There are m ants in a colony, all of which begin simultaneously from the starting vertex. Ants use a probability rule to determine which vertex will be chosen next. The rule is based on a simple ACO. More specifically, ant a in step i moves from a vertex in set V_{i-1} to vertex $v_{i,j} \in V_i$ with a probability given by:

$$\mathrm{prob}(a, v_{i,j}) = \frac{\tau(v_{i,j})}{\sum_{1 \le k \le 2d_i+1} \tau(v_{i,k})},$$

where $\tau(v_{i,k})$ is the amount of pheromone in vertex $v_{i,k}$.

The ants repeat this action until they reach the ending vertex. For each ant i, path $\vec{p_i}$ is constructed. If for some predetermined number of tries (in our case m^2 for all ants) we get $\vec{p_i} = \mathbf{0}$ the search process is reset by randomly choosing new $\vec{x}^{\,\mathrm{tb}}$ and pheromone re-initialization. Otherwise, a new solution $\vec{x_i}$ is constructed.

After all ants have created solutions, they are being evaluated with a calculation of $y_i = f(\vec{x_i})$. The information about the best among them is stored as currently best information ($\vec{x}^{\,\mathrm{cb}}$, $\vec{p}^{\,\mathrm{cb}}$, and $y_i^{\,\mathrm{cb}}$).

The current best solution, $\vec{x}^{\,\mathrm{cb}}$ is compared to the temporary best solution $\vec{x}^{\,\mathrm{tb}}$. If $y^{\,\mathrm{cb}}$ is better than $y^{\,\mathrm{tb}}$, then temporally best information is replaced with currently best information. In this case s_{global} is increased according to the global scale increase factor, s_+:

$$s_{\mathrm{global}} \leftarrow (1 + s_+) s_{\mathrm{global}},$$

s_{local} is set to

$$s_{\mathrm{local}} = \frac{1}{2} s_{\mathrm{global}}$$

and pheromone amount is redistributed according to the associated path $\vec{p}^{\,\mathrm{cb}}$, where $l_i = z(p_i^{\,\mathrm{cb}})$, so that the peak of Cauchy distribution is over with path selected vertex. Furthermore, if new $y^{\,\mathrm{tb}}$ is better then global best $y^{\,\mathrm{b}} = f(x^{\,\mathrm{b}})$, then globally best information is replaced with temporally best information. So, global best solution is stored. If no better solution is found s_{global} is decreased according to the global scale decrease factor, s_-:

$$s_{\mathrm{global}} \leftarrow (1 - s_-) s_{\mathrm{global}}.$$

Pheromone evaporation is defined by some predetermined percentage ρ. The probability density function $C(z)$ is changed in the following way:

$$l_i \leftarrow (1 - \rho) l_i$$

and

$$s_{\mathrm{local}} \leftarrow (1 - \rho) s_{\mathrm{local}}.$$

Here we must emphasize that $\rho > s_-$, because otherwise we might get negative scale factor.

The whole procedure is then repeated until some ending condition is met.

The pseudocode of the Differential Ant-Stigmergy Algorithm (DASA) is presented as follows:

Algorithm 1 The DASA

1: $\vec{x}^{\text{ tb}}$ = `Rnd_Solution()`
2: $y^{\text{ b}} = f(\vec{x}^{\text{ tb}})$
3: $y^{\text{ tb}} = \inf$
4: $\mathcal{G}$ = `Graph_Initialization`$(\vec{x}^{\text{ tb}}, \vec{\epsilon})$
5: `Pheromone_Initialization`$(\mathcal{G})$
6: **while** not ending condition met **do**
7: $\quad k = 0$
8: $\quad$ **for all** m ants **do**
9: $\quad\quad$ **repeat**
10: $\quad\quad\quad \vec{p_i}$ = `Find_Path`$(\mathcal{G})$
11: $\quad\quad\quad k = k + 1$
12: $\quad\quad\quad$ **if** $k > m^2$ **then**
13: $\quad\quad\quad\quad \vec{x}^{\text{ tb}}$ = `Rnd_Solution()`
14: $\quad\quad\quad\quad$ `Pheromone_Initialization`$(\mathcal{G})$
15: $\quad\quad\quad\quad$ **goto** line 7
16: $\quad\quad\quad$ **end if**
17: $\quad\quad$ **until** $(\vec{p_i} = \mathbf{0})$
18: $\quad\quad \omega$ = `Random_Integer`$(1, b-1)$
19: $\quad\quad \vec{x_i} = \vec{x}^{\text{ tb}} + \omega\delta(\vec{p})$
20: $\quad$ **end for**
21: $\quad y^{\text{ cb}} = \inf$
22: $\quad$ **for all** m ants **do**
23: $\quad\quad y = f(\vec{x_i})$
24: $\quad\quad$ **if** $y < y^{\text{ cb}}$ **then**
25: $\quad\quad\quad y^{\text{ cb}} = y$
26: $\quad\quad\quad \vec{p}^{\text{ cb}} = \vec{p_i}$
27: $\quad\quad\quad \vec{x}^{\text{ cb}} = \vec{x_i}$
28: $\quad\quad$ **end if**
29: $\quad$ **end for**
30: $\quad$ **if** $y^{\text{ cb}} < y^{\text{ tb}}$ **then**
31: $\quad\quad y^{\text{ tb}} = y^{\text{ cb}}$
32: $\quad\quad \vec{x}^{\text{ tb}} = \vec{x}^{\text{ cb}}$
33: $\quad\quad s$ = `Update_Scales`$(s_{\text{global}}, s_{\text{local}})$
34: $\quad\quad$ `Pheromone_Redistribution`$(\vec{p}^{\text{ cb}}, s)$
35: $\quad\quad$ **if** $y^{\text{ tb}} < y^{\text{ b}}$ **then**
36: $\quad\quad\quad y^{\text{ b}} = y^{\text{ tb}}$
37: $\quad\quad\quad \vec{x}^{\text{ b}} = \vec{x}^{\text{ tb}}$
38: $\quad\quad$ **end if**
39: $\quad$ **else**
40: $\quad\quad$ `Update_Scale`(s_{global})
41: $\quad$ **end if**
42: $\quad$ `Pheromone_Evaporation`$(\mathcal{G}, \rho)$
43: **end while**

3. EXPERIMENTAL PROCEDURE

The DASA includes six parameters: the number of ants, m, the pheromone dispersion factor, ρ, the global scale-increasing factor, s_+, the global scale-decreasing factor, s_-, the maximum parameter precision, ε, and the discrete base, b. The single setting for parameters' was used for all functions, i.e., the crafting effort was zero. We set $m = 30$ and $\rho = 0.2$, while other parameters' settings are standard [8]: $s_+ = 0.01$, $s_- = 0.02$, $\varepsilon = 1.0 \times 10^{-15}$, and $b = 10$.

Maximal number of restarts was set to 1000 and maximal number of FEs was set to $D \times 10^6$.

4. RESULTS

Results from experiments according to [5] on the benchmarks functions given in [4, 6] are presented in Figures 1 and 2 and in Tables 1 and 2.

5. CPU TIMING EXPERIMENT

For the the timing experiment the DASA was run on f_{108}. The computer platform used to perform the experiments was based on Intel Core i7 processors at 3.5 GHz, 6 GB of RAM, and the Microsoft Windows Vista x64 operating system. The DASA was implemented in Borland Delphi and for the function testbed the original implementation of functions in C were used in a form of dynamic link library. The results were 2.7; 3.0; 3.8; 5.2; 7.5 and 12.8 $\times 10^{-6}$ seconds per function evaluation in dimension 2; 3; 5; 10; 20 and 40, respectively. We can see a proportional dependency of CPU time on the search space dimensionality.

6. CONCLUSIONS

In this paper a stigmergy-based algorithm called Differential Ant-Stigmergy Algorithm (DASA) for solving optimization problems with continuous variables was presented. The performance of the DASA was evaluated on the noisy function testbed provided for Black-Box Optimization Benchmarking (BBOB).

The DASA performs acceptable on functions with moderate noise ($f_{101}, \ldots, f_{106}$) with exception of f_{101} and f_{102}, which are solved with no problems. While on the rest of the functions with severe noise ($f_{107}, \ldots, f_{130}$) performs poorly.

7. REFERENCES

[1] G. Bilchev and I. C. Parmee. The ant colony metaphor for searching continuous design spaces. In T. C. Fogarty, editor, *Evolutionary Computing*, volume 993 of *Lecture Notes in Computer Science*, pages 25–39, Sheeld, UK, April 3-4 1995.

[2] V. Cutello, G. Narzisi, G. Nicosia, and M. Pavone. An immunological algorithm for global numerical optimization. In E.-G. Talbi, P. Liardet, P. Collet, E. Lutton, and M. Schoenauer, editors, *Proceedings of the 7th International Conference on Artificial Evolution, Evolution Artificielle, EA 2005*, volume 3871 of *Lecture Notes in Computer Science*, pages 284–295, Lille, France, 2006. Springer-Verlag.

[3] K. Deb, A. Anand, and D. Joshi. A computationally efficient evolutionary algorithm for real-parameter optimization. *Evolutionary Computation*, 10(4):371–395, December 2002.

[4] S. Finck, H. Hansen, R. Ros, and A. Auger. Real-parameter black-box optimization benchmarking 2009: Presentation of the Noisy Functions., Technical Report 2009/21, Research Center PPE, 2009.

[5] N. Hansen, A. Auger, S. Finck, and R. Ros. Real-parameter black-box optimization benchmarking

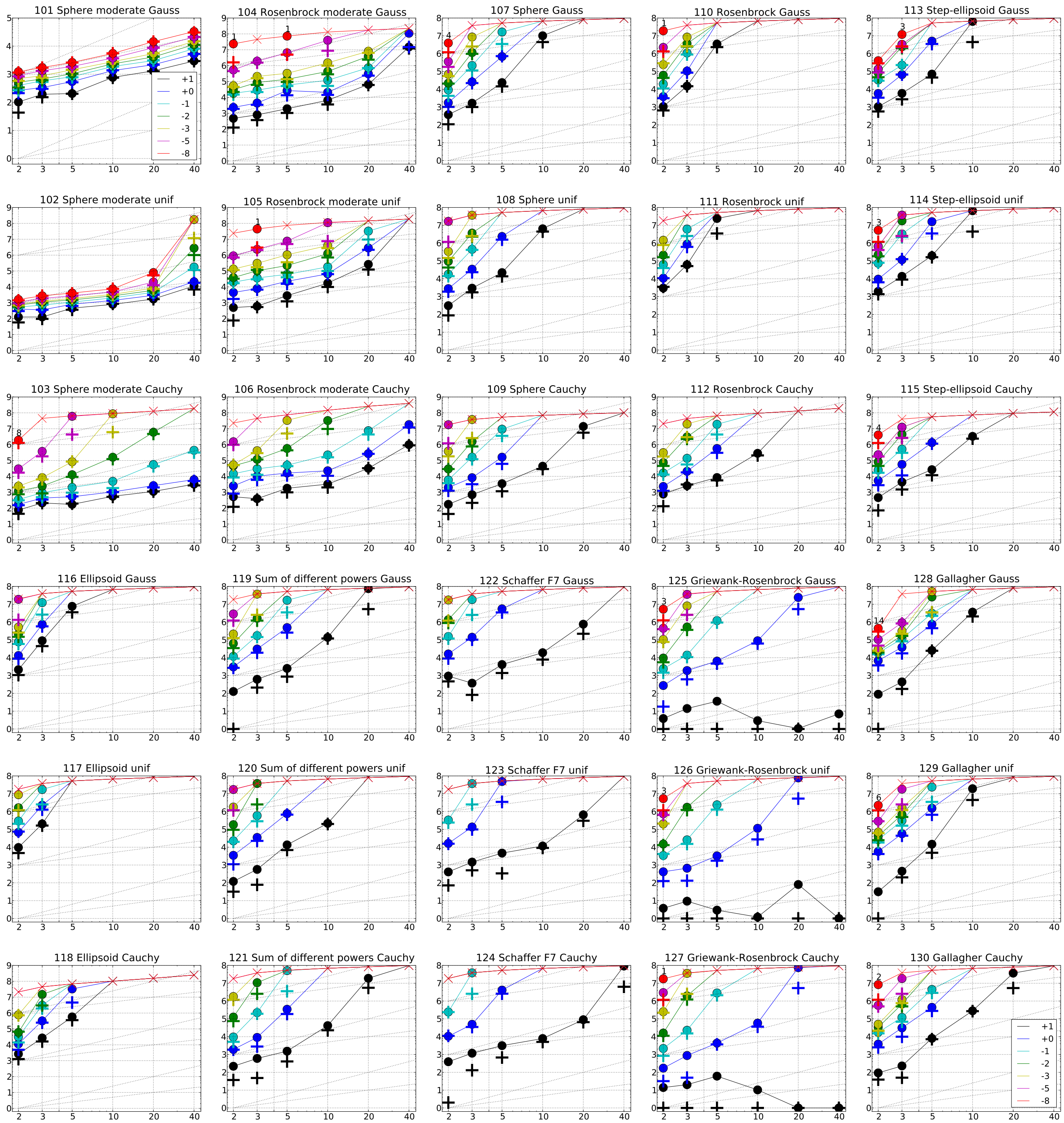

Figure 1: Expected Running Time (ERT, ●) to reach $f_{\mathrm{opt}} + \Delta f$ and median number of function evaluations of successful trials (+), shown for $\Delta f = 10, 1, 10^{-1}, 10^{-2}, 10^{-3}, 10^{-5}, 10^{-8}$ (the exponent is given in the legend of f_{101} and f_{130}) versus dimension in 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_{101} **in 5-D**, N=15, mFE=3783					f_{101} **in 20-D**, N=15, mFE=15966						f_{102} **in 5-D**, N=15, mFE=5883					f_{102} **in 20-D**, N=15, mFE=280403				
Δf	#	ERT	10%	90%	RT$_{succ}$	#	ERT	10%	90%	RT$_{succ}$	Δf	#	ERT	10%	90%	RT$_{succ}$	#	ERT	10%	90%	RT$_{succ}$
10	15	2.1e2	1.6e2	2.5e2	2.1e2	15	1.3e3	1.2e3	1.4e3	1.3e3	10	15	4.6e2	3.7e2	5.4e2	4.6e2	15	1.7e3	1.6e3	1.9e3	1.7e3
1	15	5.5e2	4.8e2	6.2e2	5.5e2	15	2.2e3	2.1e3	2.3e3	2.2e3	1	15	8.1e2	6.6e2	9.5e2	8.1e2	15	2.9e3	2.7e3	3.2e3	2.9e3
1e−1	15	8.1e2	7.4e2	8.8e2	8.1e2	15	3.1e3	2.9e3	3.2e3	3.1e3	1e−1	15	1.1e3	9.6e2	1.3e3	1.1e3	15	4.6e3	4.2e3	4.9e3	4.6e3
1e−3	15	1.4e3	1.4e3	1.5e3	1.4e3	15	5.4e3	5.3e3	5.6e3	5.4e3	1e−3	15	1.9e3	1.7e3	2.0e3	1.9e3	15	8.6e3	7.3e3	9.8e3	8.6e3
1e−5	15	1.9e3	1.8e3	2.0e3	1.9e3	15	8.8e3	8.7e3	8.9e3	8.8e3	1e−5	15	2.8e3	2.5e3	3.0e3	2.8e3	15	2.0e4	1.7e4	2.4e4	2.0e4
1e−8	15	2.6e3	2.5e3	2.8e3	2.6e3	15	1.4e4	1.4e4	1.5e4	1.4e4	1e−8	15	4.1e3	3.8e3	4.5e3	4.1e3	15	8.0e4	5.7e4	1.1e5	8.0e4

	f_{103} **in 5-D**, N=15, mFE=4352862					f_{103} **in 20-D**, N=15, mFE=8678352						f_{104} **in 5-D**, N=15, mFE=5000023					f_{104} **in 20-D**, N=15, mFE=12334272				
Δf	#	ERT	10%	90%	RT$_{succ}$	#	ERT	10%	90%	RT$_{succ}$	Δf	#	ERT	10%	90%	RT$_{succ}$	#	ERT	10%	90%	RT$_{succ}$
10	15	1.7e2	1.5e2	1.9e2	1.7e2	15	1.1e3	1.1e3	1.2e3	1.1e3	10	15	1.9e3	1.3e3	2.7e3	1.9e3	15	6.6e4	4.6e4	8.9e4	6.6e4
1	15	5.2e2	4.6e2	5.9e2	5.2e2	15	2.6e3	2.0e3	3.3e3	2.6e3	1	15	2.7e4	1.4e4	4.0e4	2.7e4	15	3.1e5	2.3e5	3.8e5	3.1e5
1e−1	15	2.0e3	9.5e2	3.2e3	2.0e3	15	5.5e4	3.7e4	7.6e4	5.5e4	1e−1	15	5.9e4	4.1e4	7.9e4	5.9e4	15	7.4e5	5.6e5	9.2e5	7.4e5
1e−3	15	8.4e4	5.5e4	1.1e5	8.4e4	0	*75e−4*	*43e−4*	*12e−3*	4.5e6	1e−3	15	3.3e5	1.9e5	4.9e5	3.3e5	12	8.0e6	6.3e6	9.5e6	6.5e6
1e−5	1	6.1e7	5.6e7	6.5e7	4.3e6	.	.	.	.	.	1e−5	8	6.5e6	5.2e6	7.7e6	3.7e6	0	*24e−5*	*27e−6*	*49e−4*	6.3e6
1e−8	0	*59e−6*	*15e−6*	*14e−5*	1.8e6	.	.	.	.	.	1e−8	1	7.2e7	7.0e7	7.5e7	5.0e6	.	.	.	.	.

	f_{105} **in 5-D**, N=15, mFE=5000029					f_{105} **in 20-D**, N=15, mFE=9938682						f_{106} **in 5-D**, N=15, mFE=5000026					f_{106} **in 20-D**, N=15, mFE=17886222				
Δf	#	ERT	10%	90%	RT$_{succ}$	#	ERT	10%	90%	RT$_{succ}$	Δf	#	ERT	10%	90%	RT$_{succ}$	#	ERT	10%	90%	RT$_{succ}$
10	15	2.8e3	2.0e3	3.6e3	2.8e3	15	2.6e5	1.4e5	3.9e5	2.6e5	10	15	1.8e3	1.2e3	2.5e3	1.8e3	15	3.2e4	2.5e4	3.9e4	3.2e4
1	15	2.3e4	1.5e4	3.1e4	2.3e4	15	2.8e6	2.0e6	3.6e6	2.8e6	1	15	1.6e4	1.2e4	2.1e4	1.6e4	15	2.7e5	2.1e5	3.4e5	2.7e5
1e−1	15	6.2e4	4.0e4	8.5e4	6.2e4	4	3.3e7	3.0e7	3.5e7	9.6e6	1e−1	15	5.0e4	3.8e4	6.4e4	5.0e4	13	7.5e6	5.4e6	9.7e6	6.9e6
1e−3	15	1.1e6	6.4e5	1.5e6	1.1e6	0	*26e−2*	*31e−3*	*78e−2*	5.6e6	1e−3	2	3.4e7	3.0e7	3.8e7	5.0e6	0	*76e−3*	*53e−3*	*14e−2*	7.9e6
1e−5	7	7.6e6	6.3e6	8.9e6	4.0e6	.	.	.	.	.	1e−5	0	*18e−4*	*72e−5*	*36e−4*	1.8e6	.	.	.	.	.
1e−8	0	*13e−6*	*19e−8*	*18e−5*	2.0e6	.	.	.	.	.	1e−8	.	.	.	.	.	.	.	.	.	.

	f_{107} **in 5-D**, N=15, mFE=3497562					f_{107} **in 20-D**, N=15, mFE=5361072						f_{108} **in 5-D**, N=15, mFE=3500682					f_{108} **in 20-D**, N=15, mFE=5360952				
Δf	#	ERT	10%	90%	RT$_{succ}$	#	ERT	10%	90%	RT$_{succ}$	Δf	#	ERT	10%	90%	RT$_{succ}$	#	ERT	10%	90%	RT$_{succ}$
10	15	2.4e4	1.5e4	3.3e4	2.4e4	0	*64e+0*	*57e+0*	*80e+0*	2.2e6	10	15	2.2e4	1.6e4	2.9e4	2.2e4	0	*64e+0*	*46e+0*	*80e+0*	1.6e6
1	15	6.9e5	5.0e5	8.9e5	6.9e5	.	.	.	.	.	1	12	2.4e6	1.9e6	2.8e6	1.9e6	.	.	.	.	.
1e−1	3	1.6e7	1.6e7	1.7e7	3.5e6	.	.	.	.	.	1e−1	0	*65e−2*	*30e−2*	*13e−1*	1.3e6	.	.	.	.	.
1e−3	0	*29e−2*	*96e−3*	*62e−2*	1.6e6	.	.	.	.	.	1e−3	.	.	.	.	.	.	.	.	.	.
1e−5	.	.	.	.	.	.	.	.	.	.	1e−5	.	.	.	.	.	.	.	.	.	.
1e−8	.	.	.	.	.	.	.	.	.	.	1e−8	.	.	.	.	.	.	.	.	.	.

	f_{109} **in 5-D**, N=15, mFE=3630702					f_{109} **in 20-D**, N=15, mFE=5717352						f_{110} **in 5-D**, N=15, mFE=3552702					f_{110} **in 20-D**, N=15, mFE=5364702				
Δf	#	ERT	10%	90%	RT$_{succ}$	#	ERT	10%	90%	RT$_{succ}$	Δf	#	ERT	10%	90%	RT$_{succ}$	#	ERT	10%	90%	RT$_{succ}$
10	15	3.5e3	2.1e3	4.8e3	3.5e3	5	1.4e7	1.3e7	1.5e7	5.7e6	10	9	3.5e6	2.7e6	4.2e6	1.4e6	0	*28e+3*	*12e+3*	*43e+3*	3.2e6
1	15	1.6e5	9.1e4	2.3e5	1.6e5	0	*12e+0*	*80e−1*	*13e+0*	1.8e6	1	0	*92e−1*	*50e−1*	*25e+0*	1.1e6	.	.	.	.	.
1e−1	5	9.3e6	8.3e6	1.0e7	3.5e6	.	.	.	.	.	1e−1	.	.	.	.	.	.	.	.	.	.
1e−3	0	*15e−2*	*44e−3*	*31e−2*	2.0e6	.	.	.	.	.	1e−3	.	.	.	.	.	.	.	.	.	.
1e−5	.	.	.	.	.	.	.	.	.	.	1e−5	.	.	.	.	.	.	.	.	.	.
1e−8	.	.	.	.	.	.	.	.	.	.	1e−8	.	.	.	.	.	.	.	.	.	.

	f_{111} **in 5-D**, N=15, mFE=3498042					f_{111} **in 20-D**, N=15, mFE=5363952						f_{112} **in 5-D**, N=15, mFE=4409022					f_{112} **in 20-D**, N=15, mFE=8856042				
Δf	#	ERT	10%	90%	RT$_{succ}$	#	ERT	10%	90%	RT$_{succ}$	Δf	#	ERT	10%	90%	RT$_{succ}$	#	ERT	10%	90%	RT$_{succ}$
10	2	2.4e7	2.2e7	2.6e7	3.5e6	0	*35e+3*	*24e+3*	*61e+3*	2.5e6	10	15	8.5e3	6.0e3	1.1e4	8.5e3	0	*18e+0*	*11e+0*	*25e+0*	2.8e6
1	0	*21e+0*	*86e−1*	*46e+0*	2.0e6	.	.	.	.	.	1	15	5.7e5	2.7e5	9.3e5	5.7e5	.	.	.	.	.
1e−1	.	.	.	.	.	.	.	.	.	.	1e−1	3	1.9e7	1.7e7	2.1e7	3.9e6	.	.	.	.	.
1e−3	.	.	.	.	.	.	.	.	.	.	1e−3	0	*18e−2*	*86e−3*	*58e−2*	1.8e6	.	.	.	.	.
1e−5	.	.	.	.	.	.	.	.	.	.	1e−5	.	.	.	.	.	.	.	.	.	.
1e−8	.	.	.	.	.	.	.	.	.	.	1e−8	.	.	.	.	.	.	.	.	.	.

	f_{113} **in 5-D**, N=15, mFE=3529992					f_{113} **in 20-D**, N=15, mFE=5368662						f_{114} **in 5-D**, N=15, mFE=3493542					f_{114} **in 20-D**, N=15, mFE=5361972				
Δf	#	ERT	10%	90%	RT$_{succ}$	#	ERT	10%	90%	RT$_{succ}$	Δf	#	ERT	10%	90%	RT$_{succ}$	#	ERT	10%	90%	RT$_{succ}$
10	15	7.1e4	4.5e4	1.0e5	7.1e4	0	*20e+1*	*13e+1*	*27e+1*	2.5e6	10	15	2.0e5	1.3e5	2.8e5	2.0e5	0	*27e+1*	*17e+1*	*33e+1*	2.2e6
1	7	5.1e6	3.9e6	6.0e6	2.2e6	.	.	.	.	.	1	3	1.6e7	1.5e7	1.7e7	3.5e6	.	.	.	.	.
1e−1	0	*11e−1*	*40e−2*	*20e−1*	2.0e6	.	.	.	.	.	1e−1	0	*18e−1*	*73e−2*	*26e−1*	2.0e6	.	.	.	.	.
1e−3	.	.	.	.	.	.	.	.	.	.	1e−3	.	.	.	.	.	.	.	.	.	.
1e−5	.	.	.	.	.	.	.	.	.	.	1e−5	.	.	.	.	.	.	.	.	.	.
1e−8	.	.	.	.	.	.	.	.	.	.	1e−8	.	.	.	.	.	.	.	.	.	.

	f_{115} **in 5-D**, N=15, mFE=3802092					f_{115} **in 20-D**, N=15, mFE=6310572						f_{116} **in 5-D**, N=15, mFE=3587622					f_{116} **in 20-D**, N=15, mFE=5377272				
Δf	#	ERT	10%	90%	RT$_{succ}$	#	ERT	10%	90%	RT$_{succ}$	Δf	#	ERT	10%	90%	RT$_{succ}$	#	ERT	10%	90%	RT$_{succ}$
10	15	2.6e4	1.6e4	3.9e4	2.6e4	0	*41e+0*	*27e+0*	*56e+0*	3.2e6	10	6	7.7e6	6.9e6	8.6e6	3.4e6	0	*95e+2*	*47e+2*	*13e+3*	2.0e6
1	14	1.3e6	1.0e6	1.7e6	1.2e6	.	.	.	.	.	1	0	*12e+0*	*32e−1*	*44e+0*	1.6e6	.	.	.	.	.
1e−1	0	*63e−2*	*27e−2*	*94e−2*	1.3e6	.	.	.	.	.	1e−1	.	.	.	.	.	.	.	.	.	.
1e−3	.	.	.	.	.	.	.	.	.	.	1e−3	.	.	.	.	.	.	.	.	.	.
1e−5	.	.	.	.	.	.	.	.	.	.	1e−5	.	.	.	.	.	.	.	.	.	.
1e−8	.	.	.	.	.	.	.	.	.	.	1e−8	.	.	.	.	.	.	.	.	.	.

	f_{117} **in 5-D**, N=15, mFE=3502932					f_{117} **in 20-D**, N=15, mFE=5361072						f_{118} **in 5-D**, N=15, mFE=4661862					f_{118} **in 20-D**, N=15, mFE=10717242				
Δf	#	ERT	10%	90%	RT$_{succ}$	#	ERT	10%	90%	RT$_{succ}$	Δf	#	ERT	10%	90%	RT$_{succ}$	#	ERT	10%	90%	RT$_{succ}$
10	0	*39e+0*	*19e+0*	*89e+0*	2.2e6	0	*12e+3*	*78e+2*	*16e+3*	2.0e6	10	15	5.7e5	3.6e5	8.1e5	5.7e5	0	*32e+1*	*24e+1*	*42e+1*	7.1e6
1	.	.	.	.	.	.	.	.	.	.	1	2	3.3e7	3.2e7	3.4e7	4.5e6	.	.	.	.	.
1e−1	.	.	.	.	.	.	.	.	.	.	1e−1	0	*27e−1*	*81e−2*	*72e−1*	1.8e6	.	.	.	.	.
1e−3	.	.	.	.	.	.	.	.	.	.	1e−3	.	.	.	.	.	.	.	.	.	.
1e−5	.	.	.	.	.	.	.	.	.	.	1e−5	.	.	.	.	.	.	.	.	.	.
1e−8	.	.	.	.	.	.	.	.	.	.	1e−8	.	.	.	.	.	.	.	.	.	.

	f_{119} **in 5-D**, N=15, mFE=3533802					f_{119} **in 20-D**, N=15, mFE=5377842						f_{120} **in 5-D**, N=15, mFE=3499962					f_{120} **in 20-D**, N=15, mFE=5369052				
Δf	#	ERT	10%	90%	RT$_{succ}$	#	ERT	10%	90%	RT$_{succ}$	Δf	#	ERT	10%	90%	RT$_{succ}$	#	ERT	10%	90%	RT$_{succ}$
10	15	2.5e3	1.7e3	3.4e3	2.5e3	1	7.5e7	7.0e7	8.1e7	5.4e6	10	15	1.4e4	9.4e3	1.8e4	1.4e4	0	*20e+0*	*13e+0*	*22e+0*	2.8e6
1	15	4.9e5	3.3e5	6.7e5	4.9e5	0	*17e+0*	*12e+0*	*19e+0*	2.2e6	1	15	7.7e5	5.5e5	9.9e5	7.7e5	.	.	.	.	.
1e−1	3	1.7e7	1.6e7	1.7e7	3.1e6	.	.	.	.	.	1e−1	0	*44e−2*	*23e−2*	*72e−2*	1.0e6	.	.	.	.	.
1e−3	0	*31e−2*	*48e−3*	*49e−2*	2.0e6	.	.	.	.	.	1e−3	.	.	.	.	.	.	.	.	.	.
1e−5	.	.	.	.	.	.	.	.	.	.	1e−5	.	.	.	.	.	.	.	.	.	.
1e−8	.	.	.	.	.	.	.	.	.	.	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 1); 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 (RT$_{\rm succ}$). 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 1 for the names of functions.

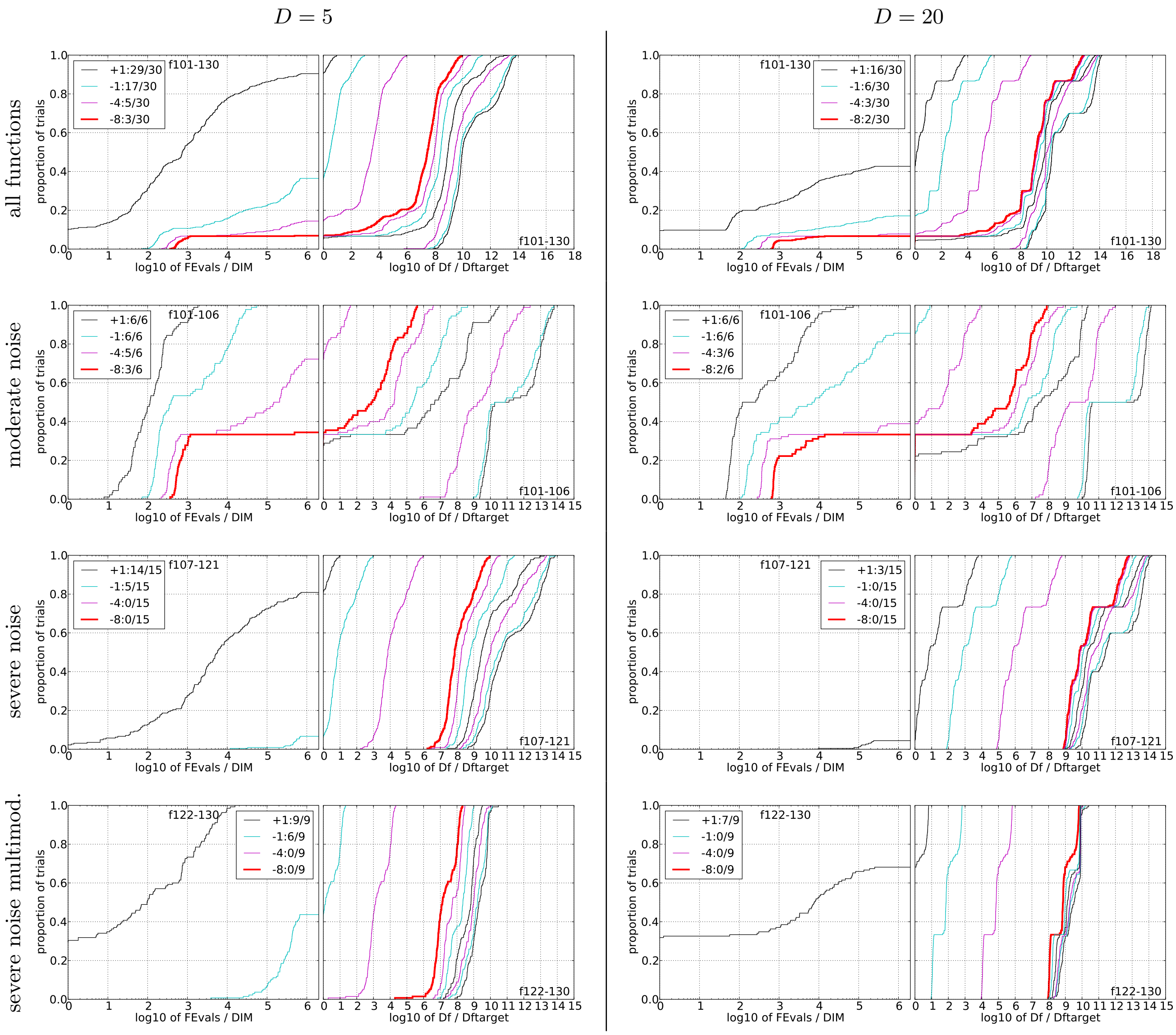

Figure 2: Empirical cumulative distribution functions (ECDFs), plotting the fraction of trials versus running time (left) or Δf. Left subplots: ECDF of the running time (number of function evaluations), divided by search space dimension D, to fall below $f_{\rm opt} + \Delta f$ with $\Delta f = 10^k$, 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: moderate noise functions; third row: severe noise functions; fourth row: severe noise and highly-multimodal functions. 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.

Δf	**f121 in 5-D**, N=15, mFE=3591222					**f121 in 20-D**, N=15, mFE=5588172				
	#	ERT	10%	90%	$\mathrm{RT}_{\mathrm{succ}}$	#	ERT	10%	90%	$\mathrm{RT}_{\mathrm{succ}}$
10	15	1.5e3	8.3e2	2.3e3	1.5e3	4	1.9e7	1.8e7	2.0e7	5.6e6
1	15	3.5e5	2.2e5	4.9e5	3.5e5	0	*11e+0*	*87e−1*	*13e+0*	4.0e6
1e−1	1	5.1e7	4.9e7	5.4e7	3.6e6	.	.	.	.	.
1e−3	0	*34e−2*	*24e−2*	*38e−2*	1.4e6	.	.	.	.	.
1e−5	.	.	.	.	.	.	.	.	.	.
1e−8	.	.	.	.	.	.	.	.	.	.

Δf	**f122 in 5-D**, N=15, mFE=3512862					**f122 in 20-D**, N=15, mFE=5386212				
	#	ERT	10%	90%	$\mathrm{RT}_{\mathrm{succ}}$	#	ERT	10%	90%	$\mathrm{RT}_{\mathrm{succ}}$
10	15	4.1e3	2.2e3	6.2e3	4.1e3	15	7.6e5	3.6e5	1.2e6	7.6e5
1	7	5.5e6	4.6e6	6.4e6	2.1e6	0	*75e−1*	*64e−1*	*95e−1*	2.0e6
1e−1	0	*10e−1*	*83e−2*	*14e−1*	2.0e6	.	.	.	.	.
1e−3	.	.	.	.	.	.	.	.	.	.
1e−5	.	.	.	.	.	.	.	.	.	.
1e−8	.	.	.	.	.	.	.	.	.	.

Δf	**f123 in 5-D**, N=15, mFE=3501042					**f123 in 20-D**, N=15, mFE=5365092				
	#	ERT	10%	90%	$\mathrm{RT}_{\mathrm{succ}}$	#	ERT	10%	90%	$\mathrm{RT}_{\mathrm{succ}}$
10	15	4.7e3	2.5e3	7.4e3	4.7e3	15	6.6e5	3.7e5	9.9e5	6.6e5
1	1	4.9e7	4.5e7	5.2e7	3.5e6	0	*74e−1*	*63e−1*	*95e−1*	1.6e6
1e−1	0	*14e−1*	*11e−1*	*20e−1*	1.8e6	.	.	.	.	.
1e−3	.	.	.	.	.	.	.	.	.	.
1e−5	.	.	.	.	.	.	.	.	.	.
1e−8	.	.	.	.	.	.	.	.	.	.

Δf	**f124 in 5-D**, N=15, mFE=3573852					**f124 in 20-D**, N=15, mFE=5509662				
	#	ERT	10%	90%	$\mathrm{RT}_{\mathrm{succ}}$	#	ERT	10%	90%	$\mathrm{RT}_{\mathrm{succ}}$
10	15	3.1e3	1.5e3	4.8e3	3.1e3	15	9.0e4	6.4e4	1.2e5	9.0e4
1	9	4.1e6	3.4e6	4.8e6	3.1e6	0	*69e−1*	*55e−1*	*75e−1*	3.2e6
1e−1	0	*97e−2*	*54e−2*	*14e−1*	2.0e6	.	.	.	.	.
1e−3	.	.	.	.	.	.	.	.	.	.
1e−5	.	.	.	.	.	.	.	.	.	.
1e−8	.	.	.	.	.	.	.	.	.	.

Δf	**f125 in 5-D**, N=15, mFE=3508992					**f125 in 20-D**, N=15, mFE=5366412				
	#	ERT	10%	90%	$\mathrm{RT}_{\mathrm{succ}}$	#	ERT	10%	90%	$\mathrm{RT}_{\mathrm{succ}}$
10	15	3.6e1	1.4e1	6.1e1	3.6e1	15	1.1e0	1.0e0	1.1e0	1.1e0
1	15	6.5e3	4.7e3	8.2e3	6.5e3	3	2.4e7	2.2e7	2.6e7	5.4e6
1e−1	15	1.2e6	9.1e5	1.5e6	1.2e6	0	*11e−1*	*97e−2*	*12e−1*	3.2e6
1e−3	0	*68e−3*	*44e−3*	*86e−3*	1.6e6	.	.	.	.	.
1e−5	.	.	.	.	.	.	.	.	.	.
1e−8	.	.	.	.	.	.	.	.	.	.

Δf	**f126 in 5-D**, N=15, mFE=3498912					**f126 in 20-D**, N=15, mFE=5361402				
	#	ERT	10%	90%	$\mathrm{RT}_{\mathrm{succ}}$	#	ERT	10%	90%	$\mathrm{RT}_{\mathrm{succ}}$
10	15	2.9e0	1.0e0	4.9e0	2.9e0	15	8.1e1	3.1e0	1.6e2	8.1e1
1	15	3.3e3	1.9e3	4.6e3	3.3e3	1	7.7e7	7.4e7	8.0e7	5.3e6
1e−1	11	2.4e6	1.8e6	2.9e6	1.8e6	0	*12e−1*	*11e−1*	*13e−1*	2.2e6
1e−3	0	*70e−3*	*48e−3*	*12e−2*	1.3e6	.	.	.	.	.
1e−5	.	.	.	.	.	.	.	.	.	.
1e−8	.	.	.	.	.	.	.	.	.	.

Δf	**f127 in 5-D**, N=15, mFE=3498132					**f127 in 20-D**, N=15, mFE=5363772				
	#	ERT	10%	90%	$\mathrm{RT}_{\mathrm{succ}}$	#	ERT	10%	90%	$\mathrm{RT}_{\mathrm{succ}}$
10	15	6.1e1	7.5e0	1.2e2	6.1e1	15	1.0e0	1.0e0	1.0e0	1.0e0
1	15	4.4e3	3.2e3	5.6e3	4.4e3	1	7.5e7	7.0e7	8.0e7	5.4e6
1e−1	11	2.8e6	2.4e6	3.4e6	2.2e6	0	*11e−1*	*10e−1*	*13e−1*	2.5e6
1e−3	0	*86e−3*	*36e−3*	*13e−2*	2.0e6	.	.	.	.	.
1e−5	.	.	.	.	.	.	.	.	.	.
1e−8	.	.	.	.	.	.	.	.	.	.

Δf	**f128 in 5-D**, N=15, mFE=3505572					**f128 in 20-D**, N=15, mFE=5361012				
	#	ERT	10%	90%	$\mathrm{RT}_{\mathrm{succ}}$	#	ERT	10%	90%	$\mathrm{RT}_{\mathrm{succ}}$
10	15	2.6e4	1.9e4	3.3e4	2.6e4	0	*61e+0*	*50e+0*	*64e+0*	1.4e6
1	15	7.5e5	5.1e5	1.0e6	7.5e5	.	.	.	.	.
1e−1	11	3.1e6	2.7e6	3.5e6	2.5e6	.	.	.	.	.
1e−3	1	5.2e7	5.2e7	5.2e7	3.5e6	.	.	.	.	.
1e−5	0	*65e−3*	*22e−4*	*43e−2*	2.0e6	.	.	.	.	.
1e−8	.	.	.	.	.	.	.	.	.	.

Δf	**f129 in 5-D**, N=15, mFE=3505512					**f129 in 20-D**, N=15, mFE=5361072				
	#	ERT	10%	90%	$\mathrm{RT}_{\mathrm{succ}}$	#	ERT	10%	90%	$\mathrm{RT}_{\mathrm{succ}}$
10	15	1.5e4	9.2e3	2.0e4	1.5e4	0	*58e+0*	*44e+0*	*65e+0*	2.2e6
1	13	1.5e6	1.0e6	2.0e6	1.5e6	.	.	.	.	.
1e−1	2	2.4e7	2.1e7	2.6e7	3.5e6	.	.	.	.	.
1e−3	0	*25e−2*	*37e−3*	*13e−1*	1.6e6	.	.	.	.	.
1e−5	.	.	.	.	.	.	.	.	.	.
1e−8	.	.	.	.	.	.	.	.	.	.

Δf	**f130 in 5-D**, N=15, mFE=3583722					**f130 in 20-D**, N=15, mFE=5366772				
	#	ERT	10%	90%	$\mathrm{RT}_{\mathrm{succ}}$	#	ERT	10%	90%	$\mathrm{RT}_{\mathrm{succ}}$
10	15	8.1e3	5.9e3	1.0e4	8.1e3	2	3.7e7	3.4e7	4.0e7	5.4e6
1	15	4.5e5	2.7e5	6.3e5	4.5e5	0	*19e+0*	*97e−1*	*37e+0*	2.5e6
1e−1	9	4.5e6	4.0e6	5.0e6	2.7e6	.	.	.	.	.
1e−3	0	*60e−3*	*23e−3*	*21e−2*	1.8e6	.	.	.	.	.
1e−5	.	.	.	.	.	.	.	.	.	.
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 $f_{\mathrm{opt}} + \Delta f$ (ERT, see Figure 1); 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 $f_{\mathrm{opt}} + \Delta f$ 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 1 for the names of functions.

2009: Experimental setup. Technical Report RR-6828, INRIA, 2009.

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

[7] J. Kennedy and R. C. Eberhart. Particle swarm optimization. In *Proceedings of the IEEE International Conference on Neural Networks*, volume IV, pages 1942–1948, Perth, Australia, December 1995. IEEE Service Center, Piscataway, NJ.

[8] P. Korošec and J. Šilc. The differential ant-stigmergy algorithm applied to dynamic optimization problems. In *Proceedings of the IEEE Congress on Evolutionary Computation*, Trondheim, Norway, May 2009. IEEE, Piscataway, NJ.

[9] R. Storn and K. V. Price. Differential evolution – a fast and efficient heuristic for global optimization over continuous spaces. *Journal of Global Optimization*, 11(4):341–359, 1997.

[10] A. H. Wright. Genetic algorithms for real parameter optimization. In G. J. E. Rawlins, editor, *Foundations of Genetic Algorithms - 1*, pages 205–218, San Mateo, CA, 1991. Morgan Kaufman.