

# Benchmarking Exponential Natural Evolution Strategies on the Noiseless and Noisy Black-box Optimization Testbeds

Tom Schaul

Courant Institute of Mathematical Sciences, New York University  
Broadway 715, New York, USA  
schaul@cims.nyu.edu

## ABSTRACT

Natural Evolution Strategies (NES) are a recent member of the class of real-valued optimization algorithms that are based on adapting search distributions. Exponential NES (xNES) are the most common instantiation of NES, and particularly appropriate for the BBOB 2012 benchmarks, given that many are non-separable, and their relatively small problem dimensions. This report provides the the most extensive empirical results on that algorithm to date, on both the noise-free and noisy BBOB testbeds.

## 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

Evolution Strategies, Natural Gradient, Benchmarking

## 1. INTRODUCTION

Evolution strategies (ES), in contrast to traditional evolutionary algorithms, aim at repeating the type of mutation that led to those good individuals. We can characterize those mutations by an explicitly parameterized *search distribution* from which new candidate samples are drawn, akin to estimation of distribution algorithms (EDA). Covariance matrix adaptation ES (CMA-ES [10]) innovated the field by introducing a parameterization that includes the full covariance matrix, allowing them to solve highly non-separable problems.

A more recent variant, *natural evolution strategies* (NES [16, 6, 14, 15]) aims at a higher level of generality, providing a

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

GECCO'12 Companion, July 7–11, 2012, Philadelphia, PA, USA.  
Copyright 2012 ACM 978-1-4503-1178-6/12/07 ...\$10.00.

procedure to update the search distribution’s parameters for any type of distribution, by ascending the gradient towards higher expected fitness. Further, it has been shown [12, 11] that following the *natural gradient* to adapt the search distribution is highly beneficial, because it appropriately normalizes the update step with respect to its uncertainty and makes the algorithm scale-invariant.

Exponential NES (xNES), the most common instantiation of NES, used a search distribution parameterized by a mean vector and a full covariance matrix, and is thus most similar to CMA-ES (in fact, the precise relation is described in [4] and [5]). Given the relatively small problem dimensions of the BBOB benchmarks, and the fact that many are non-separable, it is also among the most appropriate NES variants for the task.

In this report, we retain the original formulation of xNES (including all parameter settings, except for an added stopping criterion) and describe the empirical performance on all 54 benchmark functions (both noise-free and noisy) of the BBOB 2012 workshop.

## 2. NATURAL EVOLUTION STRATEGIES

Natural evolution strategies (NES) maintain a search distribution  $\pi$  and adapt the distribution parameters  $\theta$  by following the *natural gradient* [1] of expected fitness  $J$ , that is, maximizing

$$J(\theta) = \mathbb{E}_\theta[f(\mathbf{z})] = \int f(\mathbf{z}) \pi(\mathbf{z} | \theta) d\mathbf{z}$$

Just like their close relative CMA-ES [10], NES algorithms are invariant under monotone transformations of the fitness function and linear transformations of the search space. Each iteration the algorithm produces  $n$  samples  $\mathbf{z}_i \sim \pi(\mathbf{z} | \theta)$ ,  $i \in \{1, \dots, n\}$ , i.i.d. from its search distribution, which is parameterized by  $\theta$ . The gradient w.r.t. the parameters  $\theta$  can be rewritten (see [16]) as

$$\nabla_\theta J(\theta) = \nabla_\theta \int f(\mathbf{z}) \pi(\mathbf{z} | \theta) d\mathbf{z} = \mathbb{E}_\theta [f(\mathbf{z}) \nabla_\theta \log \pi(\mathbf{z} | \theta)]$$

from which we obtain a Monte Carlo estimate

$$\nabla_\theta J(\theta) \approx \frac{1}{n} \sum_{i=1}^n f(\mathbf{z}_i) \nabla_\theta \log \pi(\mathbf{z}_i | \theta)$$

of the search gradient. The key step then consists in replacing this gradient by the natural gradient defined as  $\mathbf{F}^{-1} \nabla_\theta J(\theta)$  where  $\mathbf{F} = \mathbb{E} [\nabla_\theta \log \pi(\mathbf{z} | \theta) \nabla_\theta \log \pi(\mathbf{z} | \theta)^\top]$  is the Fisher information matrix. The search distribution is iteratively

updated using natural gradient ascent

$$\theta \leftarrow \theta + \eta \mathbf{F}^{-1} \nabla_{\theta} J(\theta)$$

with learning rate parameter  $\eta$ .

## 2.1 Exponential NES

While the NES formulation is applicable to arbitrary parameterizable search distributions [16, 11], the most common variant employs multinormal search distributions. For that case, two helpful techniques were introduced in [6], namely an exponential parameterization of the covariance matrix, which guarantees positive-definiteness, and a novel method for changing the coordinate system into a “natural” one, which makes the algorithm computationally efficient. The resulting algorithm, NES with a multivariate Gaussian search distribution and using both these techniques is called *xNES*, and the pseudocode is given in Algorithm 1.

**Algorithm 1:** Exponential NES (xNES)

---

```

input:  $f$ ,  $\mu_{\text{init}}$ ,  $\eta_{\sigma}$ ,  $\eta_{\mathbf{B}}$ ,  $u_k$ 
       $\mu \leftarrow \mu_{\text{init}}$ 
initialize  $\sigma \leftarrow 1$ 
       $\mathbf{B} \leftarrow \mathbb{I}$ 

repeat
  for  $k = 1 \dots n$  do
    draw sample  $\mathbf{s}_k \sim \mathcal{N}(0, \mathbb{I})$ 
     $\mathbf{z}_k \leftarrow \mu + \sigma \mathbf{B}^{\top} \mathbf{s}_k$ 
    evaluate the fitness  $f(\mathbf{z}_k)$ 
  end

  sort  $\{(\mathbf{s}_k, \mathbf{z}_k)\}$  with respect to  $f(\mathbf{z}_k)$ 
  and assign utilities  $u_k$  to each sample

  compute gradients
   $\nabla_{\delta} J \leftarrow \sum_{k=1}^n u_k \cdot \mathbf{s}_k$ 
   $\nabla_{\mathbf{M}} J \leftarrow \sum_{k=1}^n u_k \cdot (\mathbf{s}_k \mathbf{s}_k^{\top} - \mathbb{I})$ 
   $\nabla_{\sigma} J \leftarrow \text{tr}(\nabla_{\mathbf{M}} J)/d$ 
   $\nabla_{\mathbf{B}} J \leftarrow \nabla_{\mathbf{M}} J - \nabla_{\sigma} J \cdot \mathbb{I}$ 

  update parameters
   $\mu \leftarrow \mu + \sigma \mathbf{B} \cdot \nabla_{\delta} J$ 
   $\sigma \leftarrow \sigma \cdot \exp(\eta_{\sigma}/2 \cdot \nabla_{\sigma} J)$ 
   $\mathbf{B} \leftarrow \mathbf{B} \cdot \exp(\eta_{\mathbf{B}}/2 \cdot \nabla_{\mathbf{B}} J)$ 

until stopping criterion is met

```

---

**Table 1:** Default parameter values for xNES (including the utility function and adaptation sampling) as a function of problem dimension  $d$ .

parameter	default value
$n$	$4 + \lfloor 3 \log(d) \rfloor$
$\eta_{\sigma} = \eta_{\mathbf{B}}$	$\frac{3(3 + \log(d))}{5d\sqrt{d}}$
$u_k$	$\frac{\max(0, \log(\frac{n}{2} + 1) - \log(k))}{\sum_{j=1}^n \max(0, \log(\frac{n}{2} + 1) - \log(j))} - \frac{1}{n}$

## 3. EXPERIMENTAL SETTINGS

We use identical default hyper-parameter values for all benchmarks (both noisy and noise-free functions), which are taken from [6, 11]. Table 1 summarizes all the hyper-parameters used.

In addition, we make use of the provided target fitness  $f_{\text{opt}}$  to trigger *independent* algorithm restarts<sup>1</sup>, using a simple ad-hoc procedure: If the log-progress during the past  $1000d$  evaluations is too small, i.e., if

$$\log_{10} \left| \frac{f_{\text{opt}} - f_t}{f_{\text{opt}} - f_{t-1000d}} \right| < (r+2)^2 \cdot m^{3/2} \cdot [\log_{10} |f_{\text{opt}} - f_t| + 8]$$

where  $m$  is the remaining budget of evaluations divided by  $1000d$ ,  $f_t$  is the best fitness encountered until evaluation  $t$  and  $r$  is the number of restarts so far. The total budget is  $10^5 d^{3/2}$  evaluations.

Implementations of this and other NES algorithm variants are available in Python through the PyBrain machine learning library [13], as well as in other languages at [www.idsia.ch/~tom/nest.html](http://www.idsia.ch/~tom/nest.html).

## 4. CPU TIMING

A timing experiment was performed to determine the CPU-time per function evaluation, and how it depends on the problem dimension. For each dimension, the algorithm was restarted with a maximum budget of  $10000/d$  evaluations, until at least 30 seconds had passed.

Our xNES implementation (in Python, based on the PyBrain [13] library), running on an Intel Xeon with 2.67GHz, required an average time of 1.1, 0.9, 0.7, 0.7, 0.9, 2.7 milliseconds per function evaluation for dimensions 2, 5, 10, 20, 40, 80 respectively (the function evaluations themselves take about 0.1ms).

## 5. RESULTS

Results of xNES on the noiseless testbed (from experiments according to [7] on the benchmark functions given in [2, 8]) are presented in Figures 1, 3 and 5 and in Tables 2 and 4.

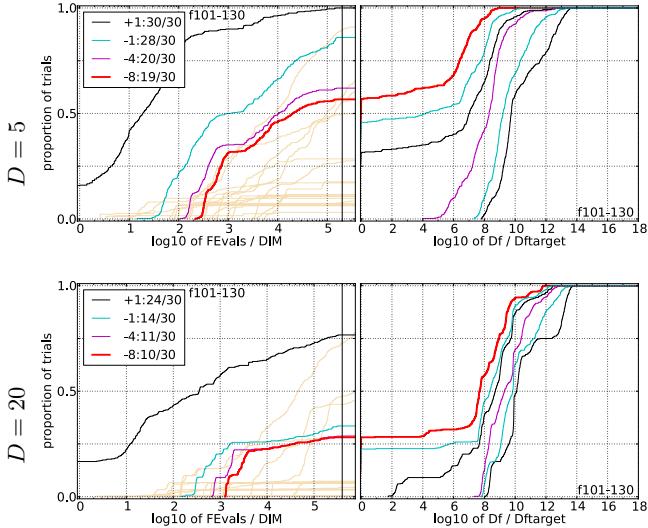
Similarly, results of xNES on the testbed of noisy functions (from experiments according to [7] on the benchmark functions given in [3, 9]) are presented in Figures 2, 4 and 5 and in Tables 3, and 4.

## 6. DISCUSSION

The top rows in Figures 3 and 4 give a good overview picture, showing that across all benchmarks taken together, xNES performs almost as well as the best and better than most of the BBOB 2009 contestants. Beyond this high-level perspective, the results speak for themselves, of course, we will just highlight a few observations.

According to Tables 2 and 3, the only conditions where xNES significantly outperforms *all* algorithms from the BBOB2009 competition on dimension 20 are on functions  $f_{18}$ ,  $f_{115}$  and  $f_{119}$  (during the early phase), as well as on  $f_{118}$  on dimension 5. We observe the worst performance on multimodal functions like  $f_3$ ,  $f_4$  and  $f_{15}$  that other algorithms tackle very

<sup>1</sup>It turns out that this use of  $f_{\text{opt}}$  is technically not permitted by the BBOB guidelines, so strictly speaking a different restart strategy should be employed, for example the one described in [11].



**Figure 4:** Empirical cumulative distribution functions (ECDFs) of the 30 noisy benchmark functions. Plotted is the fraction of trials versus running time (left subplots) or versus  $\Delta f$  (right subplots) (see Figure 3 for details).

easily. Comparing different types of noise, xNES appears to be least sensitive to Cauchy noise and most sensitive to uniform noise (see Figure 2).

From Figure 5 and Table 4, we observe a good loss ratio across the board on all benchmarks, with the best ones on moderate functions, ill-conditioned functions, and for all levels of noise. On the other hand, the algorithm is less competitive on (noisy or noise-free) multimodal benchmarks, which we expect to be directly related to its small default population size.

## Acknowledgements

The author wants to thank the organizers of the BBOB workshop for providing such a well-designed benchmark setup, and especially such high-quality post-processing utilities.

This work was funded in part through AFR postdoc grant number 2915104, of the National Research Fund Luxembourg.

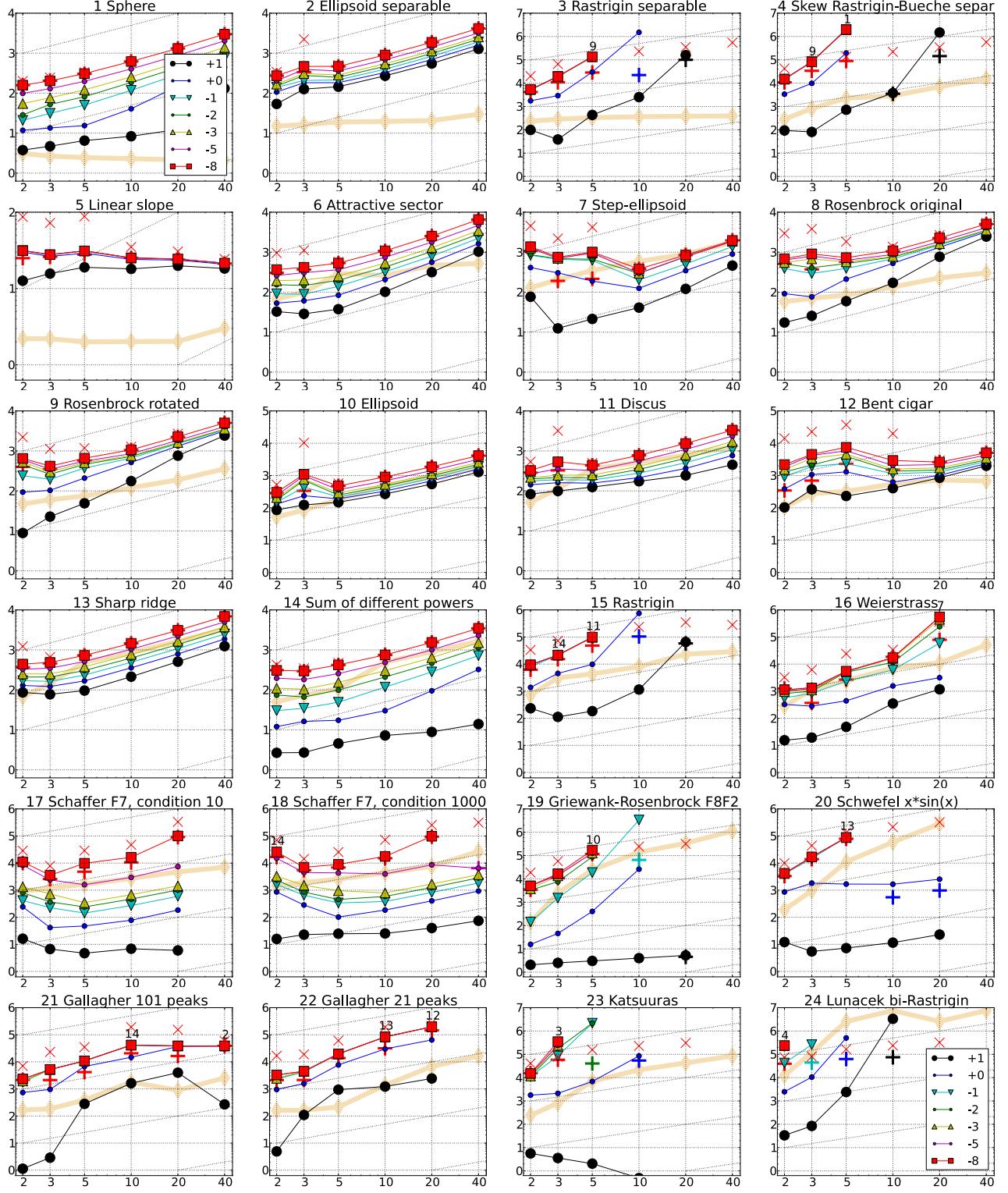
## 7. REFERENCES

- [1] S. I. Amari. Natural Gradient Works Efficiently in Learning. *Neural Computation*, 10:251–276, 1998.
- [2] 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. Updated February 2010.
- [3] 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.
- [4] N. Fukushima, Y. Nagata, S. Kobayashi, and I. Ono. Proposal of distance-weighted exponential natural evolution strategies. In *2011 IEEE Congress of*

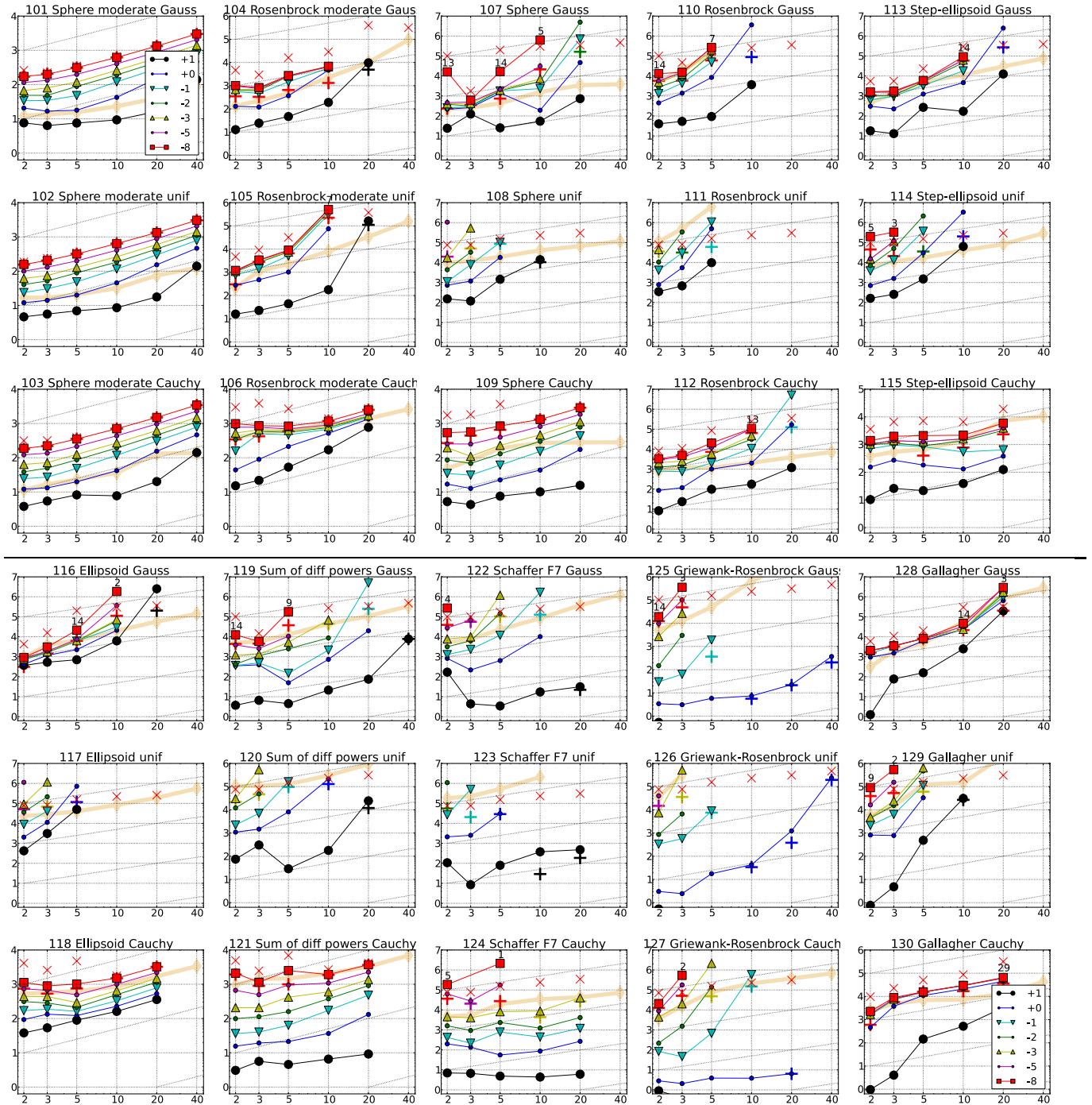
**Table 4:** ERT loss ratio compared to the respective best result from BBOB-2009 for budgets given in the first column (see also Figure 5). The last row RLUs/D gives the number of function evaluations in unsuccessful runs divided by dimension. Shown are the smallest, 10%-ile, 25%-ile, 50%-ile, 75%-ile and 90%-ile value (smaller values are better). The ERT Loss ratio equals to one for the respective best algorithm from BBOB-2009. Typical median values are between ten and hundred.

<i>f1-f24</i> in 5-D, maxFE/D=164731						
#FEs/D	best	10%	25%	med	75%	90%
2	1.7	2.4	4.2	7.0	10	10
10	1.6	2.1	2.6	3.4	4.9	11
100	0.81	1.6	3.4	6.5	12	42
1e3	0.51	1.2	2.4	6.7	26	76
1e4	1.2	1.8	3.2	12	41	1.3e2
1e5	1.2	1.8	3.2	8.7	26	3.1e2
1e6	1.2	1.8	3.2	8.7	26	6.8e2
RLUs/D	1e5	1e5	1e5	1e5	2e5	2e5
<i>f1-f24</i> in 20-D, maxFE/D=370397						
#FEs/D	best	10%	25%	med	75%	90%
2	1.0	2.4	11	31	40	40
10	0.70	1.6	2.2	4.0	5.9	27
100	2.1	3.1	7.5	14	35	2.7e2
1e3	0.86	1.4	5.6	12	1.0e2	2.4e2
1e4	0.95	1.7	2.7	11	95	6.2e2
1e5	0.95	1.9	4.5	17	3.3e2	1.2e3
1e6	0.95	1.9	4.5	25	6.3e2	8.6e3
RLUs/D	3e5	3e5	3e5	3e5	3e5	4e5
<i>f101-f130</i> in 5-D, maxFE/D=200043						
#FEs/D	best	10%	25%	med	75%	90%
2	2.1	4.2	5.6	10	10	10
10	1.0	1.5	2.0	4.6	26	50
100	0.43	1.3	2.7	11	56	5.0e2
1e3	0.61	1.3	2.9	12	46	2.8e3
1e4	1.2	1.5	1.8	7.0	20	2.5e4
1e5	1.2	1.5	3.1	12	20	37
1e6	1.2	1.7	2.3	15	66	2.2e2
RLUs/D	1e5	1e5	1e5	2e5	2e5	2e5
<i>f101-f130</i> in 20-D, maxFE/D=400059						
#FEs/D	best	10%	25%	med	75%	90%
2	1.1	2.1	14	40	40	40
10	0.63	0.71	0.95	3.4	1.2e2	2.0e2
100	0.51	0.69	1.7	4.5	16	2.0e3
1e3	0.08	0.78	2.1	8.6	28	2.0e4
1e4	0.82	1.6	3.5	12	39	2.0e5
1e5	0.82	1.8	11	44	94	1.0e6
1e6	0.82	1.5	11	1.4e2	5.6e2	1.0e7
1e7	0.82	1.8	11	8.5e2	5.4e3	1.0e8
RLUs/D	3e5	3e5	3e5	3e5	3e5	4e5

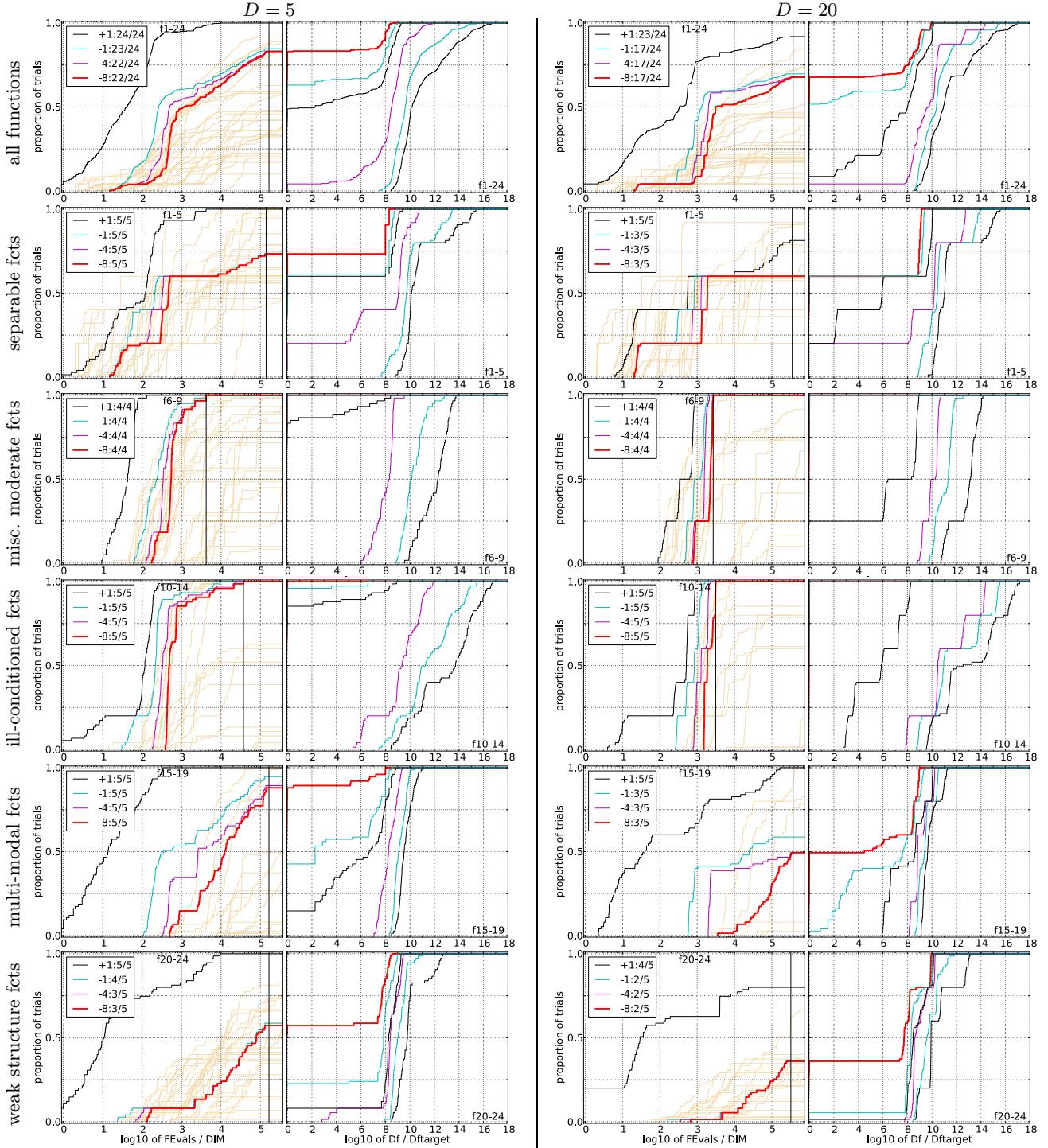
- [5] T. Glasmachers, T. Schaul, and J. Schmidhuber. A Natural Evolution Strategy for Multi-Objective Optimization. In *Parallel Problem Solving from Nature (PPSN)*, 2010.
- [6] T. Glasmachers, T. Schaul, Y. Sun, D. Wierstra, and



**Figure 1:** Expected number of  $f$ -evaluations (ERT, with lines, see legend) to reach  $f_{\text{opt}} + \Delta f$ , median number of  $f$ -evaluations to reach the most difficult target that was reached at least once (+) and maximum number of  $f$ -evaluations in any trial ( $\times$ ), all divided by dimension and plotted as  $\log_{10}$  values versus dimension. Shown are  $\Delta f = 10^{\{1,0,-1,-2,-3,-5,-8\}}$ . Numbers above ERT-symbols indicate the number of successful trials. The light thick line with diamonds indicates the respective best result from BBOB-2009 for  $\Delta f = 10^{-8}$ . Horizontal lines mean linear scaling, slanted grid lines depict quadratic scaling.



**Figure 2:** Expected number of  $f$ -evaluations (ERT, with lines, see legend) to reach  $f_{\text{opt}} + \Delta f$ , median number of  $f$ -evaluations to reach the most difficult target that was reached at least once (+) and maximum number of  $f$ -evaluations in any trial ( $\times$ ), all divided by dimension and plotted as  $\log_{10}$  values versus dimension. Shown are  $\Delta f = 10^{\{1,0,-1,-2,-3,-5,-8\}}$ . Numbers above ERT-symbols indicate the number of successful trials. The light thick line with diamonds indicates the respective best result from BBOB-2009 for  $\Delta f = 10^{-8}$ . Horizontal lines mean linear scaling, slanted grid lines depict quadratic scaling.

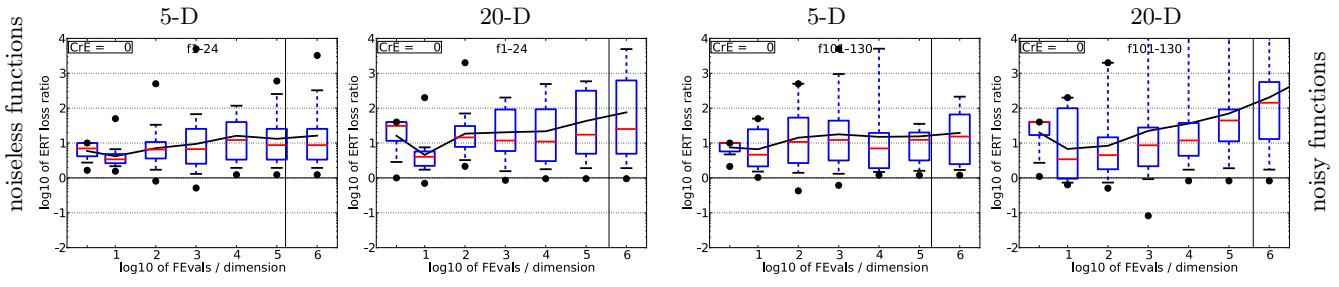


**Figure 3: Empirical cumulative distribution functions (ECDFs), plotting the fraction of trials with an outcome not larger than the respective value on the  $x$ -axis. Left subplots: ECDF of number of function evaluations (FEvals) divided by search space dimension  $D$ , to fall below  $f_{\text{opt}} + \Delta f$  with  $\Delta f = 10^k$ , where  $k$  is the first value in the legend. Right subplots: ECDF of the best achieved  $\Delta f$  divided by  $10^{-8}$  for running times of  $D, 10D, 100D, \dots$  function evaluations (from right to left cycling black-cyan-magenta). The thick red line represents the most difficult target value  $f_{\text{opt}} + 10^{-8}$ . Legends indicate the number of functions that were solved in at least one trial. Light brown lines in the background show ECDFs for  $\Delta f = 10^{-8}$  of all algorithms benchmarked during BBOB-2009.**

5-D							
$\Delta f$	1e+1	1e+0	1e-1	1e-3	1e-5	1e-7	#succ
<b>f<sub>1</sub></b>	11 3.0(3)	12 6.3(4)	12 21(3)	12 50(6)	12 81(8)	12 110(7)	15/15
<b>f<sub>2</sub></b>	83 8.7(2)	87 10(2)	88 12(1)	90 16(1)	92 20(1)	94 23(1)	15/15
<b>f<sub>3</sub></b>	716 3.0(1)	1622 92(114)	1637 414(497)	1646 412(445)	1650 412(488)	1654 411(468)	15/15
<b>f<sub>4</sub></b>	809 4.5(6)	1633 599(531)	1688 5871(6011)	1817 5453(6233)	1886 5255(6181)	1903 5208(5690)	15/15
<b>f<sub>5</sub></b>	10 9.5(4)	10 15(6)	10 15(6)	10 16(6)	10 16(6)	10 16(6)	15/15
<b>f<sub>6</sub></b>	11.4 1.6(1)	214 2.0(0.5)	281 2.5(0.3)	580 2.2(0.3)	1038 1.8(0.1)	1332 1.8(0.1)	15/15
<b>f<sub>7</sub></b>	24 4.5(2)	324 2.9(0.5)	1171 2.7(4)	1572 3.0(6)	1572 3.0(6)	1597 3.0(6)	15/15
<b>f<sub>8</sub></b>	73 4.1(2)	273 3.9(1)	336 5.8(3)	391 7.3(4)	410 7.7(4)	422 8.3(4)	15/15
<b>f<sub>9</sub></b>	35 7.1(2)	127 8.2(3)	214 8.7(4)	300 8.4(5)	335 8.3(4)	369 8.4(4)	15/15
<b>f<sub>10</sub></b>	349 2.1(0.6)	500 1.9(0.3)	574 1.9(0.2)	626 2.3(0.2)	829 2.2(0.2)	880 2.5(0.2)	15/15
<b>f<sub>11</sub></b>	143 4.4(1)	202 3.9(1)	763 1.2(0.3)	1177 1.1(0.2)	1467 1.1(0.1)	1673 1.2(0.1)	15/15
<b>f<sub>12</sub></b>	108 11(3)	268 24(38)	371 32(42)	461 50(106)	1303 22(39)	1494 25(35)	15/15
<b>f<sub>13</sub></b>	132 3.6(0.8)	195 4.3(0.3)	250 4.7(0.3)	1310 1.4(0.1)	1752 1.5(0.1)	2255 1.5(0.1)	15/15
<b>f<sub>14</sub></b>	10 2.3(2)	41 2.1(1)	58 4.2(1)	139 5.5(0.7)	251 5.1(0.7)	476 3.9(0.2)	15/15
<b>f<sub>15</sub></b>	511 1.8(1)	9310 5.3(6)	19369 25(32)	20073 24(25)	20769 24(30)	21359 23(24)	14/15
<b>f<sub>16</sub></b>	120 2.0(2)	612 3.6(2)	2662 4.4(8)	10449 2.5(3)	11644 2.3(3)	12095 2.2(3)	15/15
<b>f<sub>17</sub></b>	5.2 4.5(6)	215 1.1(0.4)	899 0.79(0.2)	3669 0.47(0.1)	6351 1.3(1)	7934 6.1(9)	15/15
<b>f<sub>18</sub></b>	103 1.2(1)	378 1.4(0.5)	3968 0.43(0.1)	9280 0.51(0.6)	10905 2.0(2)	12469 3.6(4)	15/15
<b>f<sub>19</sub></b>	1 15(16)	1 2009(1734)	242 386(502)	1.2e5 5.3(5)	1.2e5 5.3(5)	1.2e5 5.6(5)	15/15
<b>f<sub>20</sub></b>	16 2.3(2)	851 10(12)	38111 11(10)	54470 8.0(7)	54861 8.0(7)	55313 7.9(7)	14/15
<b>f<sub>21</sub></b>	41 34(123)	1157 29(37)	1674 32(44)	1705 32(44)	1729 31(43)	1757 31(42)	14/15
<b>f<sub>22</sub></b>	71 66(209)	386 100(193)	938 104(119)	1008 97(110)	1040 95(107)	1068 92(104)	14/15
<b>f<sub>23</sub></b>	3.0 3.4(3)	518 66(68)	14249 755(820)	31654 $\infty$	33030 $\infty$	34256 $\infty$	15/15
<b>f<sub>24</sub></b>	1622 7.5(10)	2.2e5 12(13)	6.4e6 $\infty$	9.6e6 $\infty$	1.3e7 $\infty$	1.3e7 $\infty$	3/15

20-D							
$\Delta f$	1e+1	1e+0	1e-1	1e-3	1e-5	1e-7	#succ
<b>f<sub>1</sub></b>	43 5.8(3)	43 70(5)	43 137(6)	43 274(5)	43 410(7)	43 546(9)	15/15
<b>f<sub>2</sub></b>	385 29(0.7)	386 36(0.7)	387 43(0.9)	390 58(1)	391 72(0.8)	393 87(1)	15/15
<b>f<sub>3</sub></b>	5066 629(842)	7626 $\infty$	7635 $\infty$	7643 $\infty$	7646 $\infty$	7651 $\infty$	15/15
<b>f<sub>4</sub></b>	4722 6323(6991)	7628 $\infty$	7666 $\infty$	7700 $\infty$	7758 $\infty$	1.4e5 $\infty$	9/15
<b>f<sub>5</sub></b>	41 10(1)	41 12(1)	41 12(1)	41 12(1)	41 12(1)	41 12(1)	15/15
<b>f<sub>6</sub></b>	1296 4.9(0.3)	2343 4.8(0.2)	3413 4.7(0.2)	5220 5.0(0.1)	6728 5.3(0.1)	8409 5.4(0.1)	15/15
<b>f<sub>7</sub></b>	1351 1.8(0.4)	4274 1.6(0.1)	9503 1.1(0.0)	16524 0.94(0.1)	16524 0.94(0.1)	16969 0.96(0.1)	15/15
<b>f<sub>8</sub></b>	2039 7.5(0.8)	3871 6.9(1)	4040 7.5(1)	4219 7.7(1)	4371 8.4(1.0)	4484 10(0.9)	15/15
<b>f<sub>9</sub></b>	1716 8.9(1)	3102 8.5(1)	3277 9.3(2)	3455 9.4(2)	3594 10(1)	3727 11(1)	15/15
<b>f<sub>10</sub></b>	7413 1.5(0.0)	8861 1.6(0.0)	10735 1.6(0.0)	14920 1.5(0.0)	17073 1.7(0.0)	17476 2.0(0.0)	15/15
<b>f<sub>11</sub></b>	1002 4.9(0.3)	2228 3.3(0.2)	6278 1.6(0.1)	9762 1.8(0.0)	12285 1.9(0.0)	14831 1.9(0.0)	15/15
<b>f<sub>12</sub></b>	1042 16(0.5)	1938 11(0.4)	2740 9.3(1)	4140 8.3(0.8)	12407 3.5(0.2)	13827 3.6(0.2)	15/15
<b>f<sub>13</sub></b>	652 16(0.5)	2021 7.8(0.2)	2751 7.9(0.2)	18749 1.8(0.0)	24455 1.8(0.0)	30201 1.9(0.0)	15/15
<b>f<sub>14</sub></b>	75 2.4(1.0)	239 7.9(1)	304 19(0.7)	932 14(0.4)	1648 12(0.3)	15661 1.7(0.0)	15/15
<b>f<sub>15</sub></b>	30378 43(41)	1.5e5 $\infty$	3.1e5 $\infty$	3.2e5 $\infty$	4.5e5 $\infty$	4.6e5 $\infty$	0/15
<b>f<sub>16</sub></b>	1384 17(8)	27265 2.3(2)	77015 15(13)	1.9e5 48(57)	2.0e5 56(57)	2.2e5 50(62)	15/15
<b>f<sub>17</sub></b>	63 1.9(1)	1030 3.6(0.6)	4005 3.0(0.1)	30677 0.94(0.0)	56288 2.7(3)	80472 12(14)	15/15
<b>f<sub>18</sub></b>	621 1.3(0.6)	3972 2.0(0.2)	19561 0.84(0.0)	67569 0.50(0.0)	1.3e5 1.3(2)	1.5e5 11(14)	15/15
<b>f<sub>19</sub></b>	1 105(59)	1 $\infty$	1 $\infty$	3.4e5 $\infty$	6.2e6 $\infty$	6.7e6 $\infty$	0/15
<b>f<sub>20</sub></b>	82 5.6(2)	46150 1.1(0.9)	3.1e6 $\infty$	5.5e6 $\infty$	5.6e6 $\infty$	5.6e6 $\infty$	14/15
<b>f<sub>21</sub></b>	561 142(281)	6541 112(168)	14103 54(75)	14643 53(72)	15567 50(68)	17589 44(60)	15/15
<b>f<sub>22</sub></b>	467 104(171)	5580 231(329)	23491 168(168)	24948 158(167)	26847 147(143)	1.3e5 29(31)	12/15
<b>f<sub>23</sub></b>	3.2 2.0(2)	1614 $\infty$	67457 $\infty$	4.9e5 $\infty$	8.1e5 $\infty$	8.4e5 $\infty$	15/15
<b>f<sub>24</sub></b>	1.3e6 $\infty$	7.5e6 $\infty$	5.2e7 $\infty$	5.2e7 $\infty$	5.2e7 $\infty$	5.2e7 $\infty$	3/15

**Table 2:** Expected running time (ERT in number of function evaluations) divided by the best ERT measured during BBOB-2009 (given in the respective first row) for different  $\Delta f$  values for functions  $f_1-f_{24}$ . The median number of conducted function evaluations is additionally given in *italics*, if  $\text{ERT}(10^{-7}) = \infty$ . #succ is the number of trials that reached the final target  $f_{\text{opt}} + 10^{-8}$ .



**Figure 5:** ERT loss ratio vs. a given budget FFEvals. The target value  $f_t$  used for a given FFEvals is the smallest (best) recorded function value such that  $\text{ERT}(f_t) \leq \text{FFEvals}$  for the presented algorithm. Shown is FFEvals divided by the respective best  $\text{ERT}(f_t)$  from BBOB-2009 for all functions (noiseless  $f_1-f_{24}$ , left columns, and noisy  $f_{101}-f_{130}$ , right columns) in 5-D and 20-D. 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 a single trial in this function subset.

5-D										20-D									
$\Delta f$	1e+1	1e+0	1e-1	1e-3	1e-5	1e-7	#succ	$\Delta f$	1e+1	1e+0	1e-1	1e-3	1e-5	1e-7	#succ				
f <sub>101</sub>	11 3.4(1)	37 2.4(1)	44 5.4(0.9)	62 10(1)	69 14(1)	75 18(0.6)	15/15	f <sub>101</sub>	59 5.5(2)	425 7.1(0.9)	571 10(0.7)	700 17(0.6)	739 24(0.4)	783 30(0.6)	15/15				
f <sub>102</sub>	11 3.1(4)	35 2.8(1)	50 4.9(1)	72 8.8(1)	86 12(1)	99 14(1)	15/15	f <sub>102</sub>	231 1.5(0.7)	399 7.7(0.9)	579 10(0.5)	921 13(0.3)	1157 15(0.2)	1407 17(0.4)	15/15				
f <sub>103</sub>	11 3.7(2)	28 3.6(1)	30 8.2(2)	31 20(3)	35 31(3)	115 13(1)	15/15	f <sub>103</sub>	65 6.2(2)	417 7.4(0.8)	629 10(0.4)	1313 9.4(0.3)	1893 10(0.4)	2464 11(0.3)	14/15				
f <sub>104</sub>	173 1.4(0.7)	773 2.4(0.9)	1287 5.6(10)	1768 7.3(11)	2040 6.4(10)	2284 5.9(9)	15/15	f <sub>104</sub>	23690 8.1(10)	85656 $\infty$	1.7e5 $\infty$	1.8e5 $\infty$	1.9e5 $\infty$	2.0e5 $\infty$	15/15				
f <sub>105</sub>	167 1.3(0.5)	1436 3.6(6)	5174 4.9(7)	10388 3.6(5)	10824 4.0(5)	11202 4.0(5)	15/15	f <sub>105</sub>	1.9e5 17(16)	6.1e5 $\infty$	6.3e5 $\infty$	6.5e5 $\infty$	6.6e5 $\infty$	6.7e5 $\infty$	15/15				
f <sub>106</sub>	92 2.9(1.0)	529 2.0(0.8)	1050 2.2(0.8)	2666 1.2(0.9)	2887 1.2(0.9)	3087 1.3(1)	15/15	f <sub>106</sub>	11480 1.3(0.1)	21668 1.2(0.1)	23746 1.3(0.1)	25470 1.3(0.1)	26492 1.5(0.1)	27360 1.7(0.1)	15/15				
f <sub>107</sub>	40 3.2(3)	228 31(66)	453 20(33)	940 10(16)	1376 7.6(11)	1850 45(8)	15/15	f <sub>107</sub>	8571 1.7(0.4)	13582 69(69)	16226 879(897)	27357 $\infty$	52486 $\infty$	65052 $\infty$	15/15				
f <sub>108</sub>	87 84(11.7)	5144 18(22)	14469 36(32)	30935 $\infty$	58628 $\infty$	80667 $\infty$	15/15	f <sub>108</sub>	58063 $\infty$	97228 $\infty$	2.0e5 $\infty$	4.5e5 $\infty$	6.3e5 $\infty$	9.0e5 $\infty$	15/15				
f <sub>109</sub>	11 3.4(2)	57 2.0(0.8)	216 1.4(0.3)	572 2.0(0.5)	873 2.3(0.2)	946 4.1(0.7)	15/15	f <sub>109</sub>	333 0.94(0.3)	632 5.5(0.7)	1138 7.7(0.4)	2287 10(0.5)	3583 10(0.5)	4952 10(0.3)	15/15				
f <sub>110</sub>	949 0.49(0.2)	33625 1.3(2)	1.2e5 2.1(3)	5.9e5 1.7(2)	6.0e5 2.2(2)	6.1e5 7/15	15/15	f <sub>110</sub>	$\infty$ $\infty$	$\infty$ $\infty$	$\infty$ $\infty$	$\infty$ $\infty$	$\infty$ $\infty$	$\infty$ $\infty$	0/15				
f <sub>111</sub>	6856 7.2(9)	6.1e5 4.1(4)	8.8e6 0.61(0.6)	2.3e7 $\infty$	3.1e7 $\infty$	3.1e7 $\infty$	3/15	f <sub>111</sub>	$\infty$ $\infty$	$\infty$ $\infty$	$\infty$ $\infty$	$\infty$ $\infty$	$\infty$ $\infty$	$\infty$ $\infty$	0/15				
f <sub>112</sub>	107 4.6(1)	1684 3.0(5)	3421 3.0(3)	4502 6.5(8)	5132 12(15)	5596 18(28)	15/15	f <sub>112</sub>	25552 0.93(0.2)	64124 52(64)	69621 1454(1597)	73557 $\infty$	76137 $\infty$	78238 $\infty$	15/15				
f <sub>113</sub>	133 10(6)	1883 3.3(8)	8081 2.1(3)	24128 1.2(2)	24128 1.2(2)	24402 1.2(2)	15/15	f <sub>113</sub>	50123 5.0(5)	3.6e5 137(148)	5.6e5 $\infty$	5.9e5 $\infty$	5.9e5 $\infty$	5.9e5 $\infty$	15/15				
f <sub>114</sub>	767 10(17)	14720 10(10)	56311 33(40)	83272 $\infty$	83272 $\infty$	84949 $\infty$	15/15	f <sub>114</sub>	2.1e5 $\infty$	1.1e6 $\infty$	1.4e6 $\infty$	1.6e6 $\infty$	1.6e6 $\infty$	1.6e6 $\infty$	15/15				
f <sub>115</sub>	64 1.7(1)	485 1.9(0.3)	1829 2.3(5)	2550 2.5(5)	2550 2.6(4)	2970 15/15	f <sub>115</sub>	2405 1.0(0.2)	30268 <b>0.25(0.1)</b> $\downarrow 4$	91749 <b>0.14(0.0)</b> $\downarrow 4$	1.3e5 <b>0.62(0.8)</b> $\downarrow 4$	1.3e5 <b>0.62(0.8)</b> $\downarrow 4$	1.3e5 <b>0.69(0.8)</b> $\downarrow 4$	15/15					
f <sub>116</sub>	5730 0.62(1)	14472 0.78(1)	22311 1.3(2)	26868 1.2(1)	30329 1.2(1)	31661 1.2(1)	15/15	f <sub>116</sub>	5.0e5 101(118)	6.9e5 $\infty$	8.9e5 $\infty$	1.0e6 $\infty$	1.1e6 $\infty$	1.1e6 $\infty$	15/15				
f <sub>117</sub>	26686 9.4(13)	76052 47(51)	1.1e5 $\infty$	1.4e5 $\infty$	1.7e5 $\infty$	1.9e5 $\infty$	15/15	f <sub>117</sub>	1.8e6 $\infty$	2.5e6 $\infty$	2.6e6 $\infty$	2.9e6 $\infty$	3.2e6 $\infty$	3.6e6 $\infty$	15/15				
f <sub>118</sub>	429 1.0(0.4)	1217 <b>0.50(0.1)</b> $\downarrow 4$	1555 <b>0.52(0.1)</b> $\downarrow 4$	1998 <b>0.73(0.1)</b> $\downarrow 2$	2430 1.0(0.2)	2913 1.6(0.2)	15/15	f <sub>118</sub>	6908 1.0(0.0)	11786 0.90(0.0)	17514 0.90(0.0)	26342 1.1(0.1)	30062 1.5(0.1)	32659 1.8(0.0)	15/15				
f <sub>119</sub>	12 1.9(2)	657 0.38(0.4)	1136 0.66(0.2)	10372 2.6(3)	35296 1.5(1)	49747 12(13)	15/15	f <sub>119</sub>	2771 <b>0.54(0.3)</b> $\downarrow 2$	29365 14(13)	35930 2741(2974)	4.1e5 $\infty$	4.1e5 $\infty$	4.1e5 $\infty$	15/15				
f <sub>120</sub>	16 9.0(12)	2900 14(17)	18698 43(42)	72438 $\infty$	3.3e5 $\infty$	5.5e5 $\infty$	15/15	f <sub>120</sub>	36040 14(23)	1.8e5 $\infty$	2.8e5 $\infty$	1.6e6 $\infty$	6.7e6 $\infty$	1.4e7 $\infty$	13/15				
f <sub>121</sub>	8.6 2.7(3)	111 1.9(2)	273 6.4(4)	1583 200(195)	3870 $\infty$	6195 $\infty$	15/15	f <sub>121</sub>	249 0.74(0.2)	769 3.4(0.6)	1426 6.7(0.4)	9304 2.9(0.2)	34434 6.4(0.1)	57404 1.1(0.1)	15/15				
f <sub>122</sub>	10 1.7(2)	1727 1.9(2)	9190 6.4(4)	30087 200(195)	53743 $\infty$	1.1e5 $\infty$	15/15	f <sub>122</sub>	692 0.91(0.8)	52008 $\infty$	1.4e5 $\infty$	7.9e5 $\infty$	2.0e6 $\infty$	5.8e6 $\infty$	15/15				
f <sub>123</sub>	11 36(70)	16066 10(9)	81505 $\infty$	3.4e5 $\infty$	6.7e5 $\infty$	2.2e6 $\infty$	15/15	f <sub>123</sub>	1063 9.0(9)	5.3e5 $\infty$	1.5e6 $\infty$	5.3e6 $\infty$	2.7e7 $\infty$	1.6e8 $\infty$	0/15				
f <sub>124</sub>	10 2.5(2)	202 1.4(0.4)	1040 3.8(10)	20478 2.0(2)	45337 19(19)	95200 112(127)	15/15	f <sub>124</sub>	192 0.63(0.3)	1959 2.7(0.3)	40840 0.58(0.1)	1.3e5 6.4(7)	3.9e5 $\infty$	8.0e5 $\infty$	15/15				
f <sub>125</sub>	1 1.1	1 29(26)	1 9476(14165)	1 $\infty$	2.4e5 $\infty$	2.4e5 $\infty$	15/15	f <sub>125</sub>	1 1.5(1)	1 451(239)	1 $\infty$	2.5e7 $\infty$	8.0e7 $\infty$	8.1e7 $\infty$	4/15				
f <sub>126</sub>	1 1.1	1 88(191)	1 44499(50671)	1 $\infty$	1 $\infty$	1 $\infty$	0/15	f <sub>126</sub>	1 1.2(0.5)	1 24814(63226)	1 $\infty$	1 $\infty$	1 $\infty$	1 $\infty$	0/30				
f <sub>127</sub>	1 1.1	1 19(22)	1 3395(5420)	1 31(35)	3.4e5 $\infty$	3.9e5 $\infty$	4.0e5 15/15	f <sub>127</sub>	1 1.4(1)	1 128(58)	1 $\infty$	4.4e6 $\infty$	7.3e6 $\infty$	7.4e6 $\infty$	15/15				
f <sub>128</sub>	111 7.1(2)	4248 7.3(11)	7808 5.2(6)	12447 3.3(4)	17217 2.4(3)	21162 2.0(2)	15/15	f <sub>128</sub>	1.4e5 28(30)	1.3e7 0.97(0.9)	1.7e7 1.2(1)	1.7e7 2.0(2)	1.7e7 3.4(4)	1.7e7 3.4(4)	9/15				
f <sub>129</sub>	64 38(34)	10710 15(10)	59443 10(9)	2.8e5 11(12)	5.1e5 $\infty$	5.8e5 $\infty$	15/15	f <sub>129</sub>	7.8e6 $\infty$	4.1e7 $\infty$	4.2e7 $\infty$	4.2e7 $\infty$	4.2e7 $\infty$	4.2e7 $\infty$	5/15				
f <sub>130</sub>	55 13(1)	812 67(84)	3034 25(25)	32823 2.4(2)	33889 2.3(2)	34528 2.3(2)	15/15	f <sub>130</sub>	4904 14(20)	93149 10(10)	2.5e5 4.9(5)	2.5e5 4.9(5)	2.6e5 4.9(5)	2.6e5 4.9(5)	7/15				
															29/30				

Table 3: ERT ratios, as in table 2, for functions  $f_{101}$ – $f_{130}$ .

- J. Schmidhuber. Exponential Natural Evolution Strategies. In *Genetic and Evolutionary Computation Conference (GECCO)*, Portland, OR, 2010.
- [7] N. Hansen, A. Auger, S. Finck, and R. Ros. Real-parameter black-box optimization benchmarking 2012: Experimental setup. Technical report, INRIA, 2012.
- [8] 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. Updated February 2010.
- [9] 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.
- [10] N. Hansen and A. Ostermeier. Completely derandomized self-adaptation in evolution strategies. *IEEE Transactions on Evolutionary Computation*, 9:159–195, 2001.
- [11] T. Schaul. *Studies in Continuous Black-box Optimization*. Ph.D. thesis, Technische Universität München, 2011.

- [12] T. Schaul. Natural Evolution Strategies Converge on Sphere Functions. In *Genetic and Evolutionary Computation Conference (GECCO)*, Philadelphia, PA, 2012.
- [13] T. Schaul, J. Bayer, D. Wierstra, Y. Sun, M. Felder, F. Sehnke, T. Rückstieß, and J. Schmidhuber. PyBrain. *Journal of Machine Learning Research*, 11:743–746, 2010.
- [14] Y. Sun, D. Wierstra, T. Schaul, and J. Schmidhuber. Stochastic search using the natural gradient. In *International Conference on Machine Learning (ICML)*, 2009.
- [15] D. Wierstra, T. Schaul, T. Glasmachers, Y. Sun, and J. Schmidhuber. Natural Evolution Strategies. Technical report, 2011.
- [16] D. Wierstra, T. Schaul, J. Peters, and J. Schmidhuber. Natural Evolution Strategies. In *Proceedings of the IEEE Congress on Evolutionary Computation (CEC)*, Hong Kong, China, 2008.