

# Benchmarking Separable 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. *Separable* NES (SNES) are a variant of NES that scale linearly with problem dimension and are particularly appropriate for large, separable problems. This report provides 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 [8]) 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, 4, 14, 15]) aims at a higher level of generality, providing a procedure to update the search distribution’s parameters for

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, July 7–11, 2012, Philadelphia, USA.

Copyright 2012 ACM 978-1-4503-0073-5/10/07 ...\$10.00.

any type of distribution, by ascending the gradient towards higher expected fitness. Further, it has been shown [11, 10] 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.

Separable NES (SNES [13]), an instantiation of NES designed for when the problem dimensionality is too high for using a full covariance matrix parameterization, instead using only the diagonal for the search distribution. It is thus quite similar to sep-CMA-ES [9]. Given the relatively small problem dimensions of the BBOB benchmarks, and the fact that many are non-separable, SNES is not the most appropriate NES variants for this particular task. In this report, we retain the original formulation of SNES (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 [8], 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 Separable NES

While the NES formulation is applicable to arbitrary parameterizable search distributions [16, 10], the most common variant employs multinormal search distributions. For that case, two helpful techniques were introduced in [4], 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*. Building on this work, a separable variant that parameterizes only the diagonal of the search distribution was introduced in [13]. The pseudocode is given in Algorithm 1.

**Algorithm 1:** Separable NES (SNES)

---

```

input:  $f$ ,  $\mu_{\text{init}}$ 
initialize  $\begin{array}{l} \mu \leftarrow \mu_{\text{init}} \\ \sigma \leftarrow 1 \end{array}$ 
repeat
  for  $k = 1 \dots n$  do
    draw sample  $\mathbf{s}_k \sim \mathcal{N}(0, \mathbb{I})$ 
     $\mathbf{z}_k \leftarrow \mu + \sigma \mathbf{s}_k$ 
    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  $\begin{array}{l} \nabla_{\mu} J \leftarrow \sum_{k=1}^n u_k \cdot \mathbf{s}_k \\ \nabla_{\sigma} J \leftarrow \sum_{k=1}^n u_k \cdot (\mathbf{s}_k^2 - 1) \end{array}$ 
  update parameters  $\begin{array}{l} \mu \leftarrow \mu + \eta_{\mu} \cdot \sigma \cdot \nabla_{\mu} J \\ \sigma \leftarrow \sigma \cdot \exp(\eta_{\sigma}/2 \cdot \nabla_{\sigma} J) \end{array}$ 
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
$n$	$4 + \lfloor 3 \log(d) \rfloor$
$\eta_{\sigma} = \eta_B$	$\frac{3 + \log(d)}{5\sqrt{d}}$
$u_k$	$\frac{\max(0, \log(\frac{n}{2} + 1) - \log(k))}{\sum_{j=1}^n \max(0, \log(\frac{n}{2} + 1) - \log(j))} - \frac{1}{n}$

## 3. EXPERIMENTAL SETTINGS

We use identical default hyper-parameter values for all benchmarks (both noisy and noise-free functions), which are taken from [13, 10]. 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 [12], 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 SNES implementation (in Python, stand-alone), running on an Intel Xeon with 2.67GHz, required an average time of 0.15, 0.16, 0.15, 0.15, 0.16, 0.18, 0.23, 0.38 milliseconds per function evaluation for dimensions 2, 5, 10, 20, 40, 80, 160, 320 respectively. Not that within that cost, the majority of computation is taken up by the function evaluations themselves, which last 0.11, 0.11, 0.12, 0.12, 0.12, 0.14, 0.17, 0.28 milliseconds each, for the same range of dimensions respectively.

## 5. RESULTS

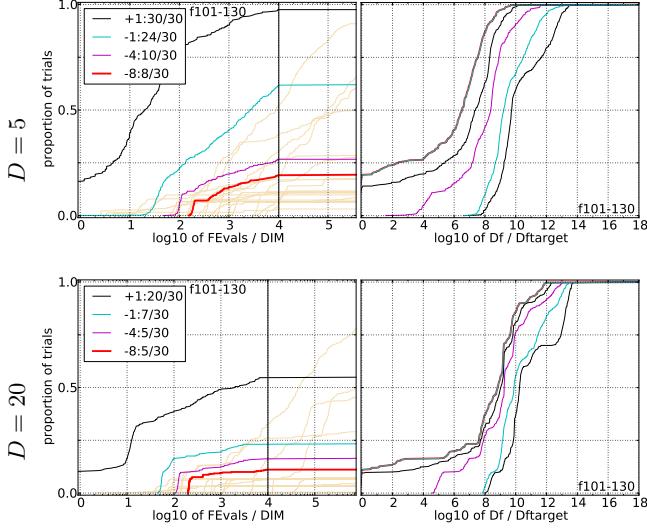
Results of SNES on the noiseless testbed (from experiments according to [5] on the benchmark functions given in [2, 6]) are presented in Figures 1, 3 and 5 and in Tables 2 and 4.

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

## 6. DISCUSSION

Given the composition of the testbeds, with many non-separable problems, it does not come as a surprise that SNES only performs well on a subset of the benchmarks (e.g., functions 1, 2, 3, 5, 21, 22, 101, 102, 103, 107, 109, 128, 130). According to Table 3, the only conditions where SNES significantly outperforms *all* algorithms from the BBOB2009 competition in dimension 20 are on functions  $f_{109}$  and  $f_{124}$  (during the early phase), and  $f_{110}$  in dimension 5. The SNES parameters were chosen for large unimodal, separable benchmarks, but we still observe a graceful decay in performance when using the algorithm on multimodal and noisy benchmarks as well. As expected, the highly non-separable problems become too hard with the separability assumption.

<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 [10].



**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).

Interestingly, from Table 4 we can see that in the early phase of convergence ( $\#FEs \approx 100d$ ), SNES is still performing well, with a median loss ratio of only 2 to 7 across all benchmarks taken together. So it appears that initial progress can be made with SNES even on non-separable functions, and that estimating the full covariance becomes more important later on for fine-tuning.

## 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.

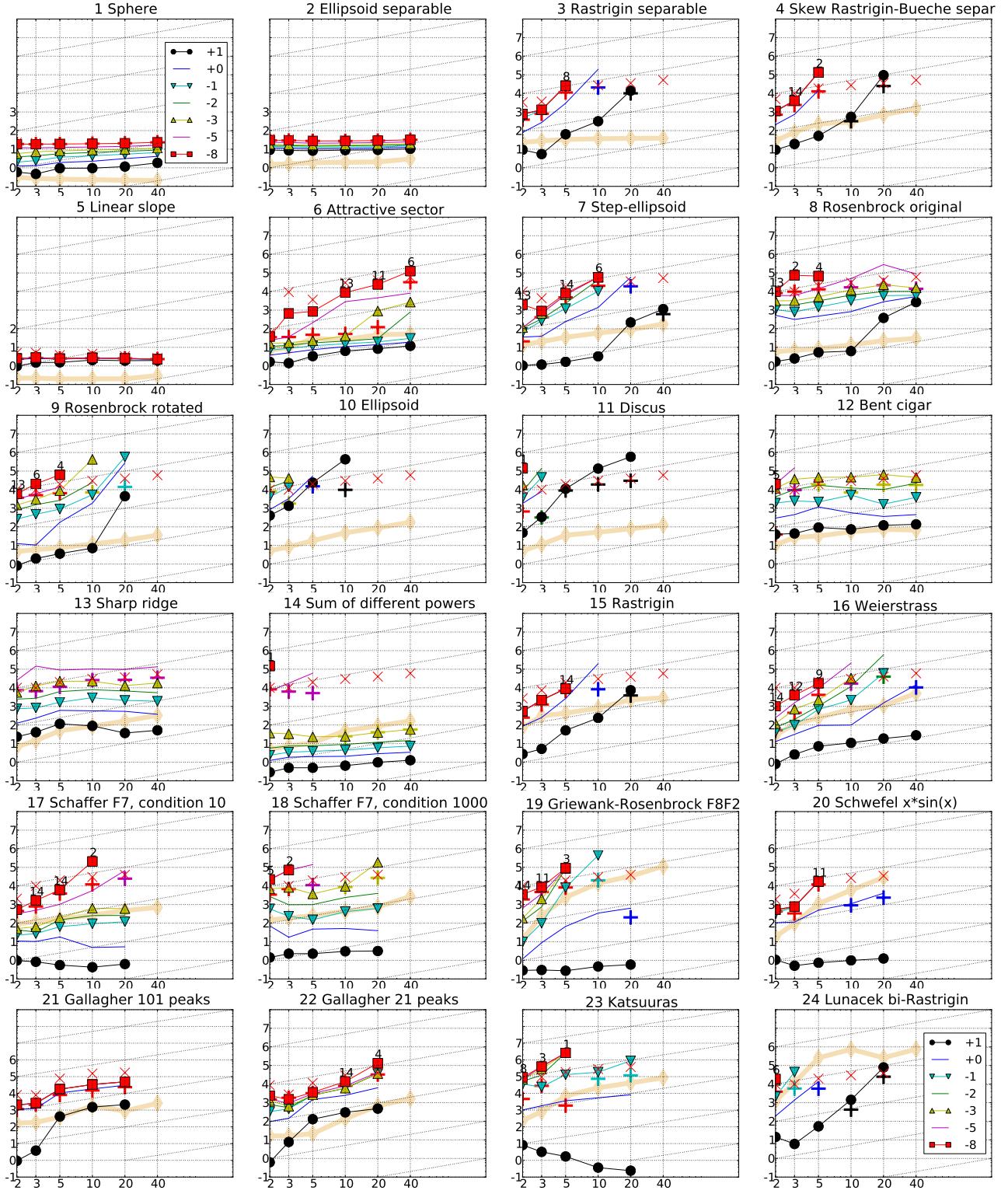
This work was funded in part through AFR postdoc grant number 2915104, of the National Research Fund Luxembourg.

## 7. REFERENCES

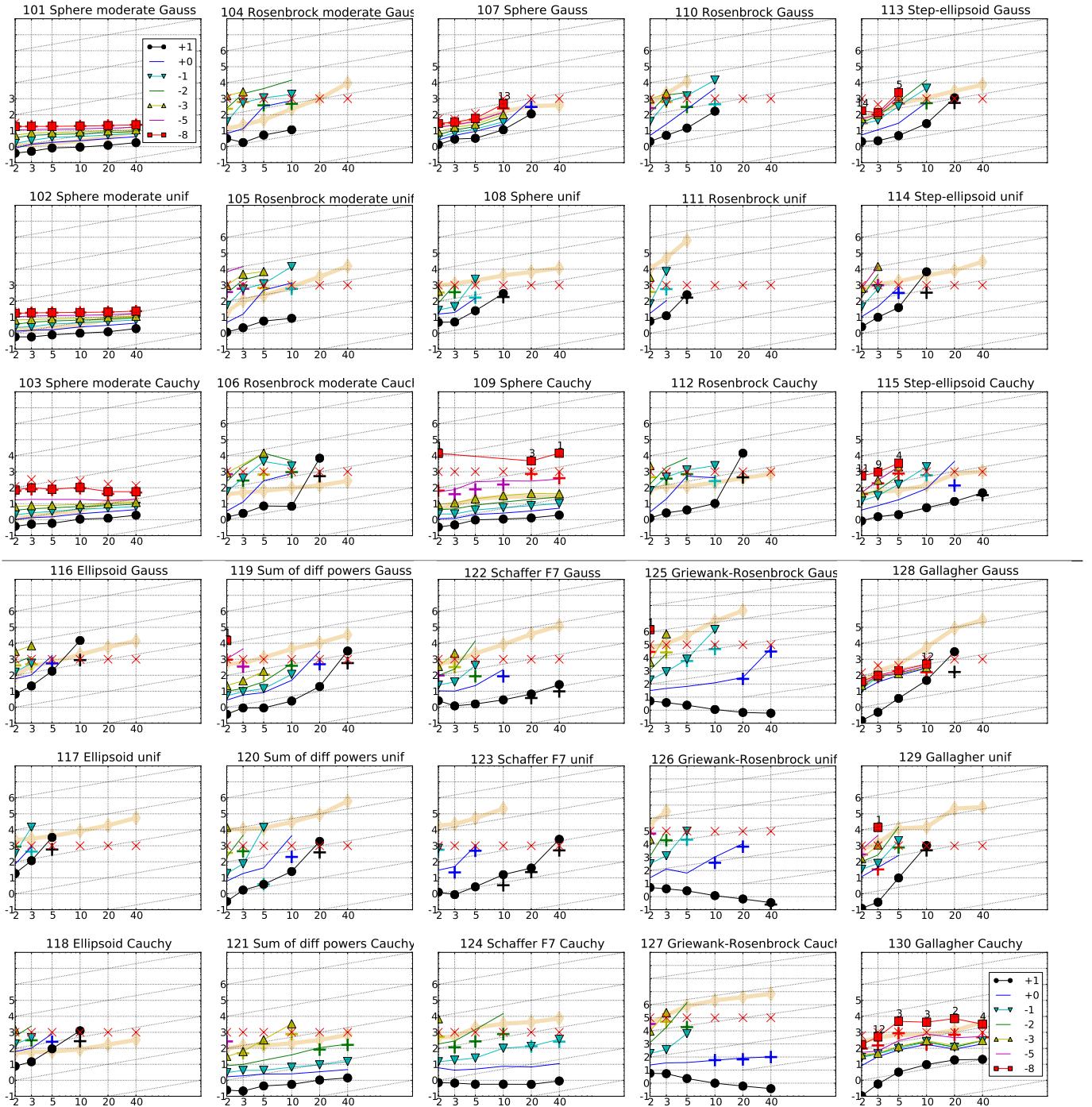
- [1] S. I. Amari. Natural Gradient Works Efficiently in Learning. *Neural Computation*, 10:251–276, 1998.
- [2] S. Finck, N. Hansen, R. Ros, and A. Auger. Real-parameter black-box optimization benchmarking 2009: Presentation of the noiseless functions. Technical Report 2009/20, Research Center PPE, 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] T. Glasmachers, T. Schaul, Y. Sun, D. Wierstra, and J. Schmidhuber. Exponential Natural Evolution Strategies. In *Genetic and Evolutionary Computation Conference (GECCO)*, Portland, OR, 2010.
- [5] N. Hansen, A. Auger, S. Finck, and R. Ros. Real-parameter black-box optimization benchmarking 2012: Experimental setup. Technical report, INRIA, 2012.
- [6] 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.
- [7] 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.

**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 RL<sub>US</sub>/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.

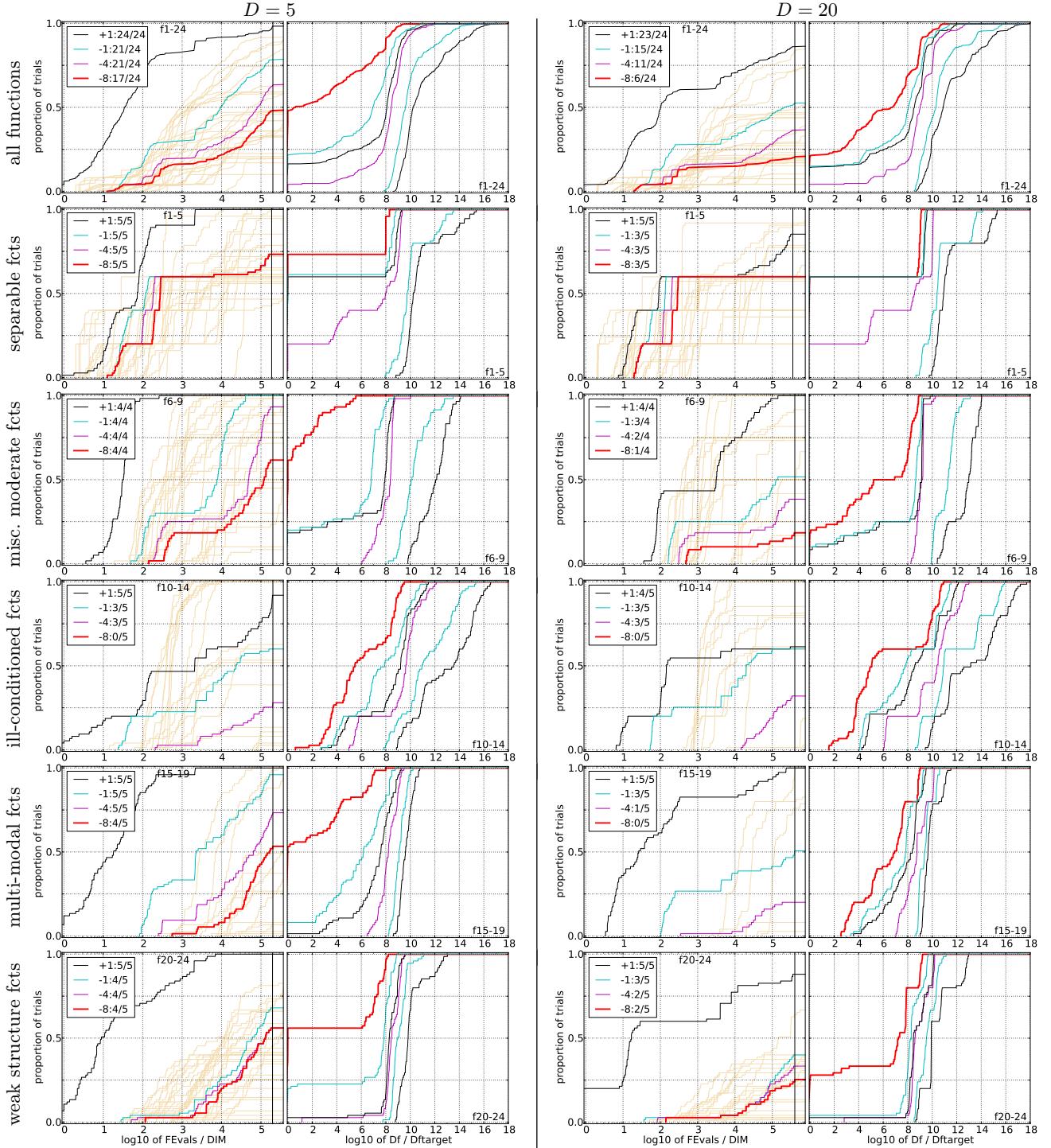
<i>f1-f24</i> in 5-D, maxFE/D=200320						
#FEs/D	best	10%	25%	<b>med</b>	75%	90%
2	1.5	2.4	4.8	7.0	9.2	10
10	2.1	2.3	2.7	3.4	4.6	14
100	0.93	2.0	4.3	7.1	14	42
1e3	1.3	3.9	7.6	29	65	80
1e4	5.9	7.9	13	69	2.5e2	4.4e2
1e5	5.2	14	38	1.2e2	1.4e3	2.1e3
1e6	12	15	33	1.8e2	5.5e3	1.2e4
RL <sub>US</sub> /D	2e5	2e5	2e5	2e5	2e5	2e5
<i>f1-f24</i> in 20-D, maxFE/D=400110						
#FEs/D	best	10%	25%	<b>med</b>	75%	90%
2	1.0	1.9	11	31	40	40
10	0.79	1.7	2.3	3.5	5.9	27
100	0.64	1.3	2.6	5.8	31	71
1e3	1.1	4.0	7.4	22	76	2.6e2
1e4	6.1	9.0	23	83	1.3e2	7.6e2
1e5	12	24	43	2.2e2	6.5e2	2.0e3
1e6	12	15	1.9e2	5.9e2	4.6e3	1.7e4
1e7	12	51	3.5e2	3.6e3	4.2e4	1.4e5
RL <sub>US</sub> /D	3e5	4e5	4e5	4e5	4e5	4e5
<i>f101-f130</i> in 5-D, maxFE/D=10152						
#FEs/D	best	10%	25%	<b>med</b>	75%	90%
2	0.86	5.6	7.1	10	10	10
10	1.3	1.9	2.4	5.1	16	50
100	0.63	0.98	1.7	2.8	9.9	2.7e2
1e3	0.47	1.1	1.2	2.1	11	2.5e3
1e4	0.42	1.4	3.1	6.3	35	2.5e4
RL <sub>US</sub> /D	1e4	1e4	1e4	1e4	1e4	1e4
<i>f101-f130</i> in 20-D, maxFE/D=10047						
#FEs/D	best	10%	25%	<b>med</b>	75%	90%
2	1.0	2.6	29	40	40	40
10	0.58	0.68	1.0	4.2	2.0e2	2.0e2
100	0.62	1.1	1.3	2.1	16	2.0e3
1e3	0.19	1.0	2.8	7.0	20	2.0e4
1e4	0.75	4.5	6.6	18	54	2.0e5
1e5	2.8	5.4	32	68	1.7e2	1.0e6
RL <sub>US</sub> /D	1e4	1e4	1e4	1e4	1e4	1e4



**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 (x), 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.

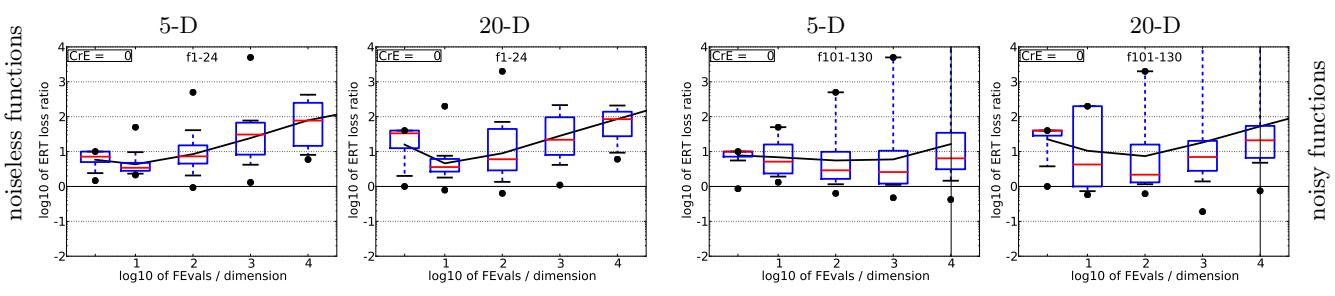


**Figure 3: Empirical cumulative distribution functions (ECDFs), plotting the fraction of trials with an outcome not larger than the respective value on the  $x$ -axis.** Left subplots: ECDF of number of function evaluations (FEvals) divided by search space dimension  $D$ , to fall below  $f_{\text{opt}} + \Delta f$  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 4.3(3)	12 7.9(3)	12 15(5)	12 33(5)	12 51(4)	12 68(5)	15/15
<b>f<sub>2</sub></b>	83 5.0(1)	87 5.9(0.9)	88 7.0(0.9)	90 9.3(0.6)	92 11(0.6)	94 13(0.6)	15/15
<b>f<sub>3</sub></b>	716 4.4(7)	1622 91(146)	1637 785(788)	1646 781(884)	1650 779(724)	1654 778(688)	8/15
<b>f<sub>4</sub></b>	809 3.2(6)	1633 443(389)	1688 3934(4283)	1817 3654(3975)	1886 3521(3907)	1903 3489(3798)	15/15 2/15
<b>f<sub>5</sub></b>	10 7.8(3)	10 12(4)	10 12(4)	10 12(4)	10 12(4)	10 12(4)	15/15
<b>f<sub>6</sub></b>	114 1.5(1)	214 1.7(0.6)	281 2.2(0.5)	580 2.0(0.6)	1038 10(15)	1332 23(51)	15/15
<b>f<sub>7</sub></b>	24 3.5(3)	324 37(76)	1171 51(56)	1572 212(256)	1572 212(256)	1597 237(253)	15/15 14/15
<b>f<sub>8</sub></b>	73 3.7(1)	273 87(92)	336 219(164)	391 667(328)	410 1770(1489)	422 4064(3811)	15/15 4/15
<b>f<sub>9</sub></b>	35 5.3(2)	127 69(48)	214 211(80)	300 1480(1872)	335 2065(1866)	369 5488(5520)	15/15 4/15
<b>f<sub>10</sub></b>	349 3419(3283)	500 5375(5002)	574 $\infty$	626 $\infty$	829 $\infty$	880 $\infty$	15/15 0/15
<b>f<sub>11</sub></b>	143 3907(2964)	202 $\infty$	763 $\infty$	1177 $\infty$	1467 $\infty$	1673 $\infty$	15/15 0/15
<b>f<sub>12</sub></b>	108 43(92)	268 220(274)	371 296(610)	461 5016(4650)	1303 $\infty$	1494 $\infty$	15/15 0/15
<b>f<sub>13</sub></b>	132 44(76)	195 157(175)	250 335(295)	1310 860(642)	1752 2575(2559)	2255 6385(6733)	15/15 0/15
<b>f<sub>14</sub></b>	10 2.6(3)	41 2.4(1)	58 3.3(1)	139 8.3(10)	251 12105(14446)	476 $\infty$	15/15 0/15
<b>f<sub>15</sub></b>	511 5.0(10)	9310 14(14)	19369 23(20)	20073 22(19)	20769 22(18)	21359 21(18)	14/15 14/15
<b>f<sub>16</sub></b>	120 3.0(3)	612 7.8(16)	2662 13(24)	10449 11(10)	11644 41(46)	12095 61(61)	15/15 9/15
<b>f<sub>17</sub></b>	5.2 5.3(4)	215 4.2(0.9)	899 3.6(6)	3669 2.7(3)	6351 8.1(9)	7934 19(20)	15/15 14/15
<b>f<sub>18</sub></b>	103 1.1(0.9)	378 6.3(13)	3968 1.9(3)	9280 20(22)	10905 647(648)	12469 $\infty$	15/15 0/15
<b>f<sub>19</sub></b>	1 14(12)	1 3327(5308)	242 1668(2179)	1.2e5 36(45)	1.2e5 36(41)	1.2e5 36(37)	15/15 3/15
<b>f<sub>20</sub></b>	16 2.3(2)	851 29(29)	38111 23(20)	54470 16(13)	54861 16(12)	55313 16(12)	14/15 11/15
<b>f<sub>21</sub></b>	41 51(123)	1157 46(74)	1674 53(65)	1705 52(63)	1729 51(62)	1757 51(62)	14/15 15/15
<b>f<sub>22</sub></b>	71 91(152)	386 191(260)	938 134(155)	1008 129(144)	1040 149(188)	1068 161(189)	14/15 15/15
<b>f<sub>23</sub></b>	3.0 2.8(2)	518 35(38)	14249 46(66)	31654 416(468)	33030 399(470)	34256 384(412)	15/15 1/15
<b>f<sub>24</sub></b>	1622 1.6(2)	2.2e5 3.2(4)	6.4e6 $\infty$	9.6e6 $\infty$	1.3e7 $\infty$	1.3e7 $\infty$	3/15 0/15

20-D							
$\Delta f$	1e+1	1e+0	1e-1	1e-3	1e-5	1e-7	#succ
<b>f<sub>1</sub></b>	43 5.4(0.8)	43 14(1)	43 25(3)	43 45(2)	43 66(2)	43 86(2)	15/15
<b>f<sub>2</sub></b>	385 4.8(0.3)	386 5.9(0.2)	387 7.0(0.3)	390 9.2(0.3)	391 11(0.3)	393 14(0.4)	15/15
<b>f<sub>3</sub></b>	5066 550(526)	7626 $\infty$	7635 $\infty$	7643 $\infty$	7646 $\infty$	7651 $\infty$	15/15 0/15
<b>f<sub>4</sub></b>	4722 4050(3891)	7628 $\infty$	7666 $\infty$	7700 $\infty$	7758 $\infty$	1.4e5 $\infty$	9/15 0/15
<b>f<sub>5</sub></b>	41 9.4(2)	41 12(3)	41 12(3)	41 12(3)	41 12(3)	41 12(3)	15/15
<b>f<sub>6</sub></b>	1296 1.3(0.2)	2343 1.2(0.2)	3413 1.2(0.2)	5220 35(57)	6728 137(396)	8409 445(523)	15/15
<b>f<sub>7</sub></b>	1351 32(59)	4274 2715(3070)	9503 $\infty$	16524 $\infty$	16524 $\infty$	16969 $\infty$	15/15 0/15
<b>f<sub>8</sub></b>	2039 37(10)	3871 145(138)	4040 299(251)	4219 1107(1149)	4371 12891(14612)	4484 25981(28550)	15/15 0/15
<b>f<sub>9</sub></b>	1716 520(442)	3102 17624(19345)	3277 35043(37845)	3455 $\infty$	3594 $\infty$	3727 $\infty$	15/15 0/15
<b>f<sub>10</sub></b>	7413 $\infty$	8661 $\infty$	10735 $\infty$	14920 $\infty$	17073 $\infty$	17476 $\infty$	15/15 0/15
<b>f<sub>11</sub></b>	1002 1.2e5(1e5)	2228 $\infty$	6278 $\infty$	9762 $\infty$	12285 $\infty$	14831 $\infty$	15/15
<b>f<sub>12</sub></b>	1042 23(38)	1938 37(41)	2740 122(57)	4140 3134(3400)	12407 $\infty$	13827 $\infty$	15/15
<b>f<sub>13</sub></b>	652 11(0.5)	2021 54(59)	2751 153(139)	18749 140(126)	24455 796(800)	30201 $\infty$	15/15 0/15
<b>f<sub>14</sub></b>	75 2.6(0.9)	239 2.4(0.4)	304 3.9(0.4)	932 8.5(4)	1648 $\infty$	15661 $\infty$	15/15 0/15
<b>f<sub>15</sub></b>	30378 49(76)	1.5e5 $\infty$	3.1e5 $\infty$	3.2e5 $\infty$	4.5e5 $\infty$	4.6e5 $\infty$	15/15 0/15
<b>f<sub>16</sub></b>	1384 2.7(1)	27265 11(15)	77015 157(157)	1.9e5 $\infty$	2.0e5 $\infty$	2.2e5 $\infty$	15/15
<b>f<sub>17</sub></b>	63 2.0(1)	1030 1.0(0.3)	4005 5.9(10)	30677 3.9(5)	56288 239(245)	80472 $\infty$	15/15
<b>f<sub>18</sub></b>	621 1.0(0.4)	3972 2.0(0.4)	19561 6.3(4)	67569 547(593)	1.3e5 $\infty$	1.5e5 $\infty$	15/15
<b>f<sub>19</sub></b>	1 118(40)	1 1.3e5(1e5)	3.4e5 $\infty$	6.2e6 $\infty$	6.7e6 $\infty$	6.7e6 $\infty$	15/15 0/15
<b>f<sub>20</sub></b>	82 3.1(0.9)	46150 18(19)	3.1e6 $\infty$	5.5e6 $\infty$	5.6e6 $\infty$	5.6e6 $\infty$	14/15 0/15
<b>f<sub>21</sub></b>	561 76(71)	6541 93(116)	14103 70(84)	14643 67(81)	15567 63(74)	17589 56(67)	15/15
<b>f<sub>22</sub></b>	467 205(255)	5580 227(281)	23491 312(318)	24948 294(334)	26847 317(325)	1.3e5 111(116)	12/15
<b>f<sub>23</sub></b>	3.2 1.5(1)	1614 102(102)	67457 261(283)	4.9e5 $\infty$	8.1e5 $\infty$	8.4e5 $\infty$	15/15
<b>f<sub>24</sub></b>	1.3e6 12(14)	7.5e6 $\infty$	5.2e7 $\infty$	5.2e7 $\infty$	5.2e7 $\infty$	5.2e7 $\infty$	3/15 0/15

**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 FEvals. The target value  $f_t$  used for a given FEvals is the smallest (best) recorded function value such that  $\text{ERT}(f_t) \leq \text{FEvals}$  for the presented algorithm. Shown is FEvals divided by the respective best 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
$f_{101}$	11 3.8(2)	37 2.5(0.9)	44 4.3(0.8)	62 6.1(0.7)	69 8.8(1.0)	75 11(1)	15/15	$f_{101}$	59 4.1(0.7)	425 1.5(0.2)	571 1.9(0.1)	700 2.8(0.2)	739 3.9(0.1)	783 4.7(0.2)	15/15
$f_{102}$	11 3.5(2)	35 2.3(1)	50 3.7(1)	72 5.4(0.7)	86 7.3(0.7)	99 8.6(0.4)	15/15	$f_{102}$	231 1.1(0.2)	399 1.6(0.3)	579 1.9(0.1)	921 2.1(0.1)	1157 2.5(0.1)	1407 2.7(0.1)	15/15
$f_{103}$	11 2.7(2)	28 2.7(1)	30 5.5(1)	31 13(2)	35 26(15)	115 21(11)	15/15	$f_{103}$	65 3.8(0.9)	417 1.5(0.2)	629 1.7(0.1)	1313 1.6(0.1)	1893 1.7(0.1)	2464 3.4(2)	14/15
$f_{104}$	173 1.5(0.6)21(26)	773 44(42)	1287 44(42)	1768 $\infty$	2040 $\infty$	2284 $\infty$	15/15	$f_{104}$	23690 $\infty$	85656 $\infty$	1.7e5 $\infty$	1.8e5 $\infty$	1.9e5 $\infty$	2.0e5 $\infty$	15/15
$f_{105}$	167 1.7(0.4)17(21)	1436 12(12)	5174 35(40)	10388 $\infty$	10824 $\infty$	11202 $\infty$	15/15	$f_{105}$	1.9e5 $\infty$	6.1e5 $\infty$	6.3e5 $\infty$	6.5e5 $\infty$	6.6e5 $\infty$	6.7e5 $\infty$	15/15
$f_{106}$	92 3.9(1)	529 26(32)	1050 209(228)	2666 276(291)	2887 $\infty$	3087 $\infty$	15/15	$f_{106}$	11480 123(140)	21668 $\infty$	23746 $\infty$	25470 $\infty$	26492 $\infty$	27360 $\infty$	15/15
$f_{107}$	40 4.2(6)	228 2.0(1)	453 1.4(0.8)	940 1.3(0.9)	1376 1.4(0.7)	1850 1.3(0.5)	15/15	$f_{107}$	8571 2.6(3)	13582 14(15)	16226 $\infty$	27357 $\infty$	52486 $\infty$	65052 $\infty$	15/15
$f_{108}$	87 15(20)	5144 1.5(3)	14469 8.4(9)	30935 $\infty$	58628 $\infty$	80667 $\infty$	0/15	$f_{108}$	58063 $\infty$	97228 $\infty$	2.0e5 $\infty$	4.5e5 $\infty$	6.3e5 $\infty$	9.0e5 $\infty$	15/15
$f_{109}$	11 4.4(2)	57 1.9(0.7)	216 0.93(0.3)	572 1.8(1)	873 13(21)	946 371(411)	15/15	$f_{109}$	333 0.77(0.1) $\downarrow^2$	632 1.1(0.1)	1138 1.3(0.2)	2287 3.8(3)	3583 15(13)	4952 103(104)	3/15
$f_{110}$	949 0.76(1)	33625 0.33(0.2) $\downarrow$	1.2e5 0.65(0.7)	5.9e5 $\infty$	6.0e5 $\infty$	6.1e5 $\infty$	15/15	$f_{110}$	$\infty$ $\infty$	$\infty$ $\infty$	$\infty$ $\infty$	$\infty$ $\infty$	$\infty$ $\infty$	$\infty$ $\infty$	0/15
$f_{111}$	6856 1.9(2)	6.1e5 $\infty$	8.8e6 $\infty$	2.3e7 $\infty$	3.1e7 $\infty$	3.1e7 $\infty$	3/15	$f_{111}$	$\infty$ $\infty$	$\infty$ $\infty$	$\infty$ $\infty$	$\infty$ $\infty$	$\infty$ $\infty$	$\infty$ $\infty$	0/15
$f_{112}$	107 1.9(0.4)16(19)	1684 19(20)	3421 $\infty$	4502 $\infty$	5132 $\infty$	5596 $\infty$	15/15	$f_{112}$	25552 113(120)	64124 $\infty$	69621 $\infty$	73557 $\infty$	76137 $\infty$	78238 $\infty$	15/15
$f_{113}$	133 1.8(2)	1883 0.78(0.9)	8081 2.1(3)	24128 3.1(3)	24128 4.1(5)	24402 $\infty$	15/15	$f_{113}$	50123 4.5(5)	3.6e5 $\infty$	5.6e5 $\infty$	5.9e5 $\infty$	5.9e5 $\infty$	5.9e5 $\infty$	15/15
$f_{114}$	767 2.6(3)	14720 2.4(3)	56311 $\infty$	83272 $\infty$	83272 $\infty$	84949 $\infty$	15/15	$f_{114}$	2.1e5 $\infty$	1.1e6 $\infty$	1.4e6 $\infty$	1.6e6 $\infty$	1.6e6 $\infty$	1.6e6 $\infty$	15/15
$f_{115}$	64 1.6(0.8)	485 1.8(2)	1829 4.4(4)	2550 40(45)	2550 40(40)	2970 45(44)	15/15	$f_{115}$	2405 1.2(1)	30268 29(32)	91749 $\infty$	1.3e5 $\infty$	1.3e5 $\infty$	1.3e5 $\infty$	15/15
$f_{116}$	5730 1.6(1)	14472 5.9(6)	22311 $\infty$	26868 $\infty$	30329 $\infty$	31661 $\infty$	15/15	$f_{116}$	5.0e5 $\infty$	6.9e5 $\infty$	8.9e5 $\infty$	1.0e6 $\infty$	1.1e6 $\infty$	1.1e6 $\infty$	15/15
$f_{117}$	26686 6.3(7)	76052 $\infty$	1.1e5 $\infty$	1.4e5 $\infty$	1.7e5 $\infty$	1.9e5 $\infty$	15/15	$f_{117}$	1.8e6 $\infty$	2.5e6 $\infty$	2.6e6 $\infty$	2.9e6 $\infty$	3.2e6 $\infty$	3.6e6 $\infty$	15/15
$f_{118}$	429 11(9)	1217 33(36)	1555 $\infty$	1998 $\infty$	2430 $\infty$	2913 $\infty$	15/15	$f_{118}$	6908 $\infty$	11786 $\infty$	17514 $\infty$	26342 $\infty$	30062 $\infty$	32659 $\infty$	15/15
$f_{119}$	12 3.9(6)	657 0.64(0.9)	1136 0.64(0.5)	10372 0.84(0.9)	35296 $\infty$	49747 $\infty$	15/15	$f_{119}$	2771 1.4(2)	29365 22(22)	35930 $\infty$	4.1e5 $\infty$	1.4e6 $\infty$	1.9e6 $\infty$	15/15
$f_{120}$	16 12(32)	2900 0.72(0.9)	18698 38(44)	72438 $\infty$	3.3e5 $\infty$	5.5e5 $\infty$	15/15	$f_{120}$	36040 10(11)	1.8e5 $\infty$	2.8e5 $\infty$	1.6e6 $\infty$	6.7e6 $\infty$	1.4e7 $\infty$	13/15
$f_{121}$	8.6 2.6(3)	111 1.1(0.8)	273 0.77(0.4)	1583 11(14)	3870 $\infty$	6195 $\infty$	15/15	$f_{121}$	249 0.83(0.2)	769 0.89(0.2)	1426 1.3(0.2)	9304 $\infty$	34434 $\infty$	57404 $\infty$	15/15
$f_{122}$	10 12(16)	1727 3.1(3)	9190 $\infty$	30087 $\infty$	53743 $\infty$	1.1e5 $\infty$	15/15	$f_{122}$	692 7.7(9)	52008 $\infty$	1.4e5 $\infty$	7.9e5 $\infty$	2.0e6 $\infty$	5.8e6 $\infty$	15/15
$f_{123}$	11 12(16)	16066 3.1(3)	81505 $\infty$	3.4e5 $\infty$	6.7e5 $\infty$	2.2e6 $\infty$	15/15	$f_{123}$	1063 7.7(9)	5.3e5 $\infty$	1.5e6 $\infty$	5.3e6 $\infty$	2.7e7 $\infty$	1.6e8 $\infty$	0/15
$f_{124}$	10 2.9(4)	202 1.2(1.0)	1040 1.2(0.9)	20478 $\infty$	45337 $\infty$	95200 $\infty$	15/15	$f_{124}$	192 0.58(0.4) $\downarrow$	1959 0.69(0.2) $\downarrow$	40840 0.66(0.5)	1.3e5 $\infty$	3.9e5 $\infty$	8.0e5 $\infty$	15/15
$f_{125}$	1 1.2(0.5)33(34)	1 3958(3495)	1 $\infty$	1 $\infty$	2.4e5 $\infty$	2.4e5 $\infty$	15/15	$f_{125}$	1 1.3(0.5)	1 625(509)	1 $\infty$	2.5e7 $\infty$	8.0e7 $\infty$	8.1e7 $\infty$	4/15
$f_{126}$	1 1.4(1)	1 32(50)	1 51876(53162)	1 $\infty$	1 $\infty$	1 $\infty$	0/15	$f_{126}$	1 1.3(0.5)	1 22572(23656)	1 $\infty$	1 $\infty$	1 $\infty$	0 $\infty$	0/15
$f_{127}$	1 1.1(0.5)19(16)	1 3060(2752)	1 $\infty$	3.4e5 $\infty$	3.9e5 $\infty$	4.0e5 $\infty$	15/15	$f_{127}$	1 1.2(0.5)	1 167(80)	1 $\infty$	4.4e6 $\infty$	7.3e6 $\infty$	7.4e6 $\infty$	15/15
$f_{128}$	111 1.6(2)	4248 1.1(1)	7808 0.78(0.8)	12447 0.51(0.5)	17217 0.40(0.3) $\downarrow^2$	21162 0.45(0.4) $\downarrow^2$	15/15	$f_{128}$	1.4e5 4.2(5)	1.3e7 $\infty$	1.7e7 $\infty$	1.7e7 $\infty$	1.7e7 $\infty$	1.7e7 $\infty$	9/15
$f_{129}$	64 7.6(14)	10710 1.2(1)	59443 1.8(2)	2.8e5 $\infty$	5.1e5 $\infty$	5.8e5 $\infty$	15/15	$f_{129}$	7.8e6 $\infty$	4.1e7 $\infty$	4.2e7 $\infty$	4.2e7 $\infty$	4.2e7 $\infty$	5/15	0/15
$f_{130}$	55 2.9(7)	812 4.8(5)	3034 1.6(2)	32823 0.19(0.2)	33889 0.44(0.4)	34528 2.7(2)	10/15	$f_{130}$	4904 0.76(1)	93149 0.19(0.3)	2.5e5 0.09(0.1)	2.5e5 0.11(0.1)	2.6e5 0.36(0.3)	2.6e5 2.1(2)	7/15 2/15

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

- [8] N. Hansen and A. Ostermeier. Completely derandomized self-adaptation in evolution strategies. *IEEE Transactions on Evolutionary Computation*, 9:159–195, 2001.
- [9] R. Ros and N. Hansen. A Simple Modification in CMA-ES Achieving Linear Time and Space Complexity. Technical Report April, 2008.
- [10] T. Schaul. *Studies in Continuous Black-box Optimization*. Ph.D. thesis, Technische Universität München, 2011.
- [11] T. Schaul. Natural Evolution Strategies Converge on Sphere Functions. In *Genetic and Evolutionary Computation Conference (GECCO)*, Philadelphia, PA, 2012.
- [12] 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.
- [13] T. Schaul, T. Glasmachers, and J. Schmidhuber. High Dimensions and Heavy Tails for Natural Evolution

Strategies. In *Genetic and Evolutionary Computation Conference (GECCO)*, Dublin, Ireland, 2011.

- [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.