Benchmarking the SMS-EMOA with Self-adaptation on the `bbob-biobj` Test Suite

Simon Wessing
TU Dortmund
Otto-Hahn-Str. 14
Dortmund, Germany 44227
simon.wessing@tu-dortmund.de

ABSTRACT

Variation operators have seemingly been less in the focus than selection operators during the first years of research on evolutionary multiobjective optimization. Several new developments in benchmarking and hypervolume selection have now sparked a renewed interest in the topic. Here, we benchmark a variant of the S-metric selection evolutionary multi-objective optimization algorithm with self-adaptive mutative control of a single step size parameter, but without recombination. It obtains better results than variants with differential evolution or polynomial mutation as variation.

CCS CONCEPTS

•**Computing methodologies → Continuous space search;**

KEYWORDS

Benchmarking, Black-box optimization, Bi-objective optimization

ACM Reference format:
Simon Wessing. 2017. Benchmarking the SMS-EMOA with Self-adaptation on the `bbob-biobj` Test Suite. In *Proceedings of GECCO '17 Companion, Berlin, Germany, July 15-19, 2017,* 8 pages.
DOI: http://dx.doi.org/10.1145/3067695.3084205

1 INTRODUCTION

For a long time, the prevailing variation operators in multiobjective optimization were simulated binary crossover combined with polynomial mutation (SBX/PM) and differential evolution (DE) [1, 15]. Also the more sophisticated approach of covariance matrix adaptation has been transferred from single-objective to multiobjective optimization [8] and usually reaches a better performance [10]. However, the historically important variation operator of classical evolution strategies for single-objective optimization [2] has been noticeably absent so far, with a few exceptions [9, 11]. While it is not expected to reach the top performance of todays derandomized variation operators, it is an interesting question if and how it would work in multiobjective optimization. This question was recently answered positively in [16], where the incorporation of self-adaptive step size control into the S-metric selection evolutionary multi-objective optimization algorithm (SMS-EMOA) was investigated. Thus, we are benchmarking the best known configuration of the algorithm on the `bbob-biobj` test suite here.

Algorithm 1 $(\mu + \lambda)$-SMS-EMOA

Input: population size μ, initial population P_0, number of offspring λ
$t \leftarrow 0$
while stopping criterion not fulfilled **do**
  $O_t \leftarrow \text{createOffspring}(P_t)$ // create λ offspring
  $\text{evaluate}(O_t)$ // calculate objective values
  $Q_t \leftarrow P_t \cup O_t$
  $\boldsymbol{r} \leftarrow \text{createReferencePoint}(Q_t)$
  while $|Q_t| > \mu$ **do**
    $\{F_1, \ldots, F_w\} \leftarrow \text{nondominatedSort}(Q_t)$ // sort in fronts
    $\boldsymbol{x}^* \leftarrow \text{argmin}_{\boldsymbol{x} \in F_w}(\Delta_s(\boldsymbol{x}, F_w, \boldsymbol{r}))$ // determine $\boldsymbol{x}^*$ with smallest contribution
    $Q_t \leftarrow Q_t \setminus \{\boldsymbol{x}^*\}$ // remove worst individual
  end while
  $P_{t+1} \leftarrow Q_t$
  $t \leftarrow t + 1$
end while

2 ALGORITHM PRESENTATION

The applicability of the Gaussian mutation operator to the SMS-EMOA was already shown in [9, 11] in different variations. It creates an offspring $\vec{y}$ from an individual $\vec{x}$ by calculating

$$\vec{y} = \vec{x} + \sigma \mathcal{N}(\vec{0}, \mathbf{I}) , \qquad (1)$$

where $\mathbf{I}$ is the identity matrix. In (1), the simplest variant with one step size parameter σ is presented. This variant requires one exogenous learning parameter τ in its adaptation rule, which is typically recommended to be chosen as $\tau = 1/\sqrt{D}$ for D-dimensional unimodal single-objective problems [2]. In this case, the step size is varied by a multiplicative mutation

$$\sigma = \tilde{\sigma} \cdot \exp(\tau \mathcal{N}(0, 1)) , \qquad (2)$$

where $\tilde{\sigma}$ is the step size of the parent individual. It is important to note that for the adaptation to work, (2) has to be carried out before the actual mutation of the decision variable in (1).

In [16], it was shown that for the self-adaptation to work well, a relatively high selection pressure $(\mu + \lambda)/\mu$ has to be used, just as in single-objective optimization [2]. Thus, the original selection scheme has to be extended to a $(\mu + \lambda)$-selection. We do this

GECCO '17 Companion, Berlin, Germany

DOI: http://dx.doi.org/10.1145/3067695.3084205

pragmatically by using a decremental greedy selection [4], also known as backward elimination. It removes the individual with the worst hypervolume contribution per iteration.

Algorithm 1 shows the pseudocode of the resulting algorithm. The corresponding implementation is avaiable in [14]. The parameters for this algorithm were determined in [16] by a full-factorial experiment on unimodal instances of the 2016 edition of `bbob-biobj`, with a budget of $10^4 D$. In this experiment, the best configuration turned out to be no recombination, a learning parameter of $\tau = 1/\sqrt{D}$, and a $(50+250)$-selection. With the budget of $10^4 D$, this configuration performed significantly better than a $(50+250)$-SMS-EMOA with standard SBX variation operator on the whole `bbob-biobj` 2016 test suite [16].

3 CPU TIMING

In order to evaluate the CPU timing of the algorithm, we have run the SMS-ES on the entire bbob-biobj test suite [13] for $10^3 D$ function evaluations according to [7]. The Python code was run on an Intel Xeon 5140 CPU @2.33GHz with 2 processors and 4 cores. The time per function evaluation for dimensions 2, 3, 5, 10, 20 equals 3.4×10^{-4}, 3.4×10^{-4}, 3.6×10^{-4}, 4.1×10^{-4}, and 5.0×10^{-4} seconds respectively.

4 EXPERIMENTS

We compare the algorithm proposed in Sect. 2 (named SMS-ES in the following) with the $(\mu + 1)$-SMS-EMOAs with simulated binary crossover/polynomial mutation (SMS-PM) and differential evolution (SMS-DE) that were benchmarked on `bbob-biobj` in 2016. Besides the selection scheme, the initialization is another notable difference between the implementations. For the SMS-ES, the hypercube $[-100, 100]^D$ is normalized to the unit hypercube. The algorithm is then initialized in $[0.475, 0.525]^D$, corresponding to the region $[-5, 5]^D$ in the original problem space. The initial mutation strength is chosen as $\sigma_{\text{init}} = 0.025$. The competitors are initalized in $[-100, 100]^D$ instead; details can be found in [1].

4.1 Results

Results from experiments according to [7], [5] and [3] on the benchmark functions given in [13] are presented in Figures 1, 2, 3 and 4 and in Tables 1 and 2. The experiments were performed with COCO [6], version 2.0, the plots were produced with version 2.1.

The **average runtime (aRT)**, used in the tables, depends on a given quality indicator value, $I_{\text{target}} = I_{\text{ref}} + \Delta I_{\text{HV}}^{\text{COCO}}$, and is computed over all relevant trials as the number of function evaluations executed during each trial while the best indicator value did not reach I_{target}, summed over all trials and divided by the number of trials that actually reached I_{target} [7, 12]. **Statistical significance** is tested with the rank-sum test for a given target I_{target} using, for each trial, either the number of needed function evaluations to reach I_{target} (inverted and multiplied by -1), or, if the target was not reached, the best $\Delta I_{\text{HV}}^{\text{COCO}}$-value achieved, measured only up to the smallest number of overall function evaluations for any unsuccessful trial under consideration.

4.2 Discussion

The figures show that the SMS-ES usually obtains a better performance than both other SMS-EMOA variants. However, SMS-DE often manages to score slightly better than SMS-ES after a budget of $10^5 D$ on multimodal problems, and dramatically better on the sphere problem. SMS-ES is in many cases on a par with the best approaches of 2016 for budgets around $10^3 D$, but inferior otherwise.

It is our impression that the greedy $(50+250)$-selection is quite beneficial for large budgets on the `bbob-biobj` test suite, because it makes it less likely to converge to a local Pareto front too quickly. So, it might also improve the other two algorithms. An alternative may of course be detecting the convergence and restarting the algorithms.

5 CONCLUSIONS

We considerably improved the performance of the SMS-EMOA by changing the selection scheme and the variation operator. As these changes are relatively easy to do, the improvement is quite satisfying. However, the SMS-ES algorithm is clearly not designed for an anytime scenario with this configuration (but neither are the other two variants), because it only provides slow progress in the early stages. This is not surprising, as the configuration was chosen for a good performance with a large budget.

Using one individual mutation strength per dimension is expected to improve the performance of the algorithm further on some problems, but is usually considered to represent the maximum potential of this approach. For further details on this topic, we refer to [2].

REFERENCES

[1] Anne Auger, Dimo Brockhoff, Nikolaus Hansen, Dejan Tušar, Tea Tušar, and Tobias Wagner. 2016. The Impact of Variation Operators on the Performance of SMS-EMOA on the Bi-objective BBOB-2016 Test Suite. In *Proceedings of the 2016 on Genetic and Evolutionary Computation Conference Companion (GECCO '16 Companion)*. ACM, 1225–1232.

[2] Hans-Georg Beyer and Hans-Paul Schwefel. 2002. Evolution strategies – A comprehensive introduction. *Natural Computing* 1, 1 (2002), 3–52.

[3] D. Brockhoff, T. Tušar, D. Tušar, T. Wagner, N. Hansen, and A. Auger. 2016. Biobjective Performance Assessment with the COCO Platform. *ArXiv e-prints* arXiv:1605.01746 (2016).

[4] A. P. Guerreiro and C. M. Fonseca. 2017. *Computing and Updating Hypervolume Contributions in Up to Four Dimensions*. Technical Report TR-2017-001. University of Coimbra.

[5] N. Hansen, A Auger, D. Brockhoff, D. Tušar, and T. Tušar. 2016. COCO: Performance Assessment. *ArXiv e-prints* arXiv:1605.03560 (2016).

[6] N. Hansen, A. Auger, O. Mersmann, T. Tušar, and D. Brockhoff. 2016. COCO: A Platform for Comparing Continuous Optimizers in a Black-Box Setting. *ArXiv e-prints* arXiv:1603.08785 (2016).

[7] N. Hansen, T. Tušar, O. Mersmann, A. Auger, and D. Brockhoff. 2016. COCO: The Experimental Procedure. *ArXiv e-prints* arXiv:1603.08776 (2016).

[8] Christian Igel, Nikolaus Hansen, and Stefan Roth. 2007. Covariance Matrix Adaptation for Multi-objective Optimization. *Evolutionary Computation* 15, 1 (2007), 1–28. DOI: http://dx.doi.org/10.1162/evco.2007.15.1.1

[9] Jan-Willem Klinkenberg, Michael T. M. Emmerich, André H. Deutz, Ofer M. Shir, and Thomas Bäck. 2010. A Reduced-Cost SMS-EMOA Using Kriging, Self-Adaptation, and Parallelization. In *Proceedings of the 19th International Conference on Multiple Criteria Decision Making*, Matthias Ehrgott, Boris Naujoks, J. Theodor Stewart, and Jyrki Wallenius (Eds.). Springer, 301–311. DOI: http://dx.doi.org/10.1007/978-3-642-04045-0_26

[10] Ilya Loshchilov and Tobias Glasmachers. 2016. Anytime Bi-Objective Optimization with a Hybrid Multi-Objective CMA-ES (HMO-CMA-ES). In *Proceedings of the 2016 on Genetic and Evolutionary Computation Conference Companion (GECCO '16 Companion)*. ACM, 1169–1176.

[11] Boris Naujoks, Domenico Quagliarella, and Thomas Bartz-Beielstein. 2006. Sequential Parameter Optimisation of Evolutionary Algorithms for Airfoil Design.

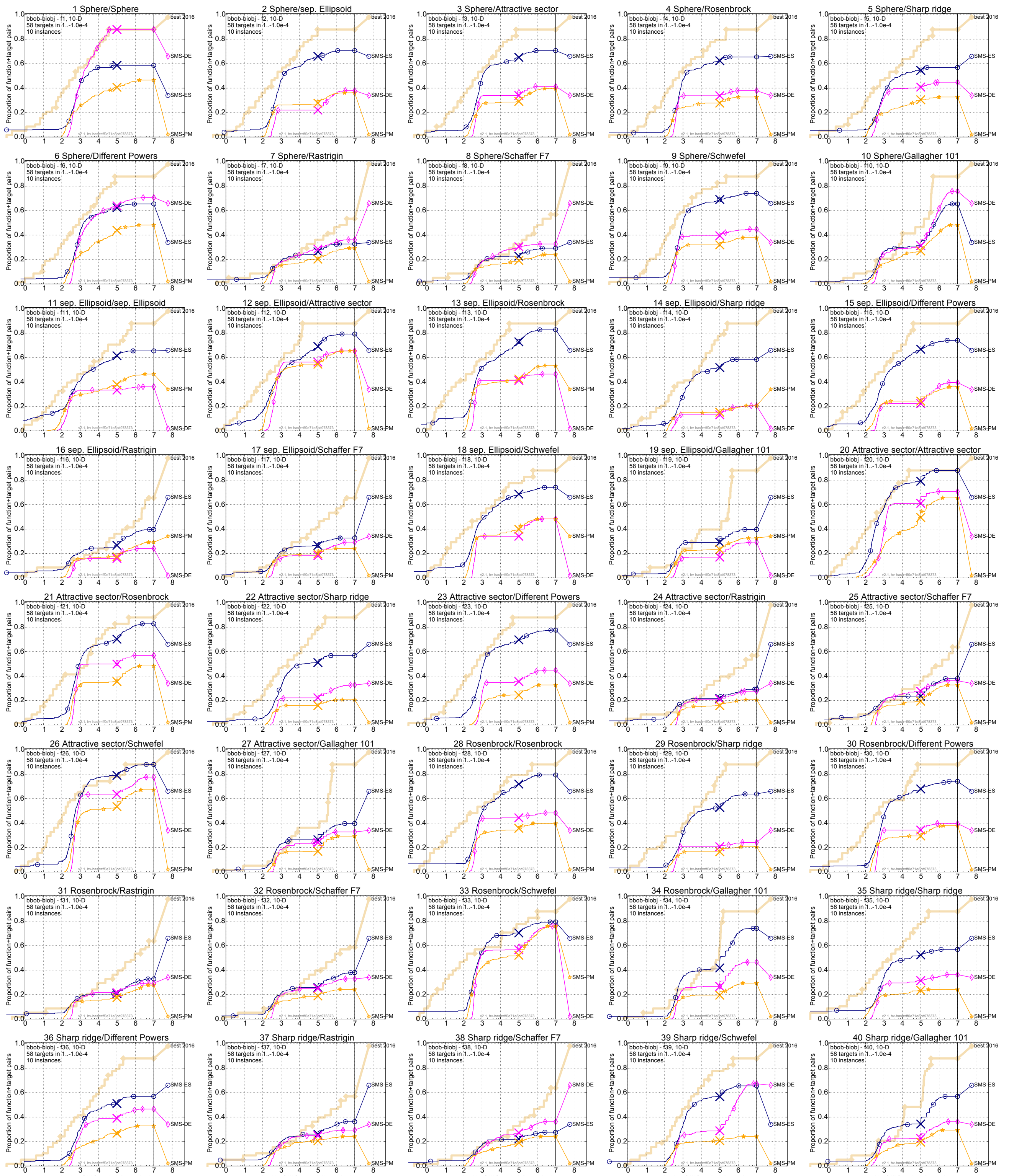

Figure 1: Empirical cumulative distribution of simulated (bootstrapped) runtimes, measured in number of objective function evaluations, divided by dimension (FEvals/DIM) for the 58 targets $\{-10^{-4}, -10^{-4.2}, -10^{-4.4}, -10^{-4.6}, -10^{-4.8}, -10^{-5}, 0, 10^{-5}, 10^{-4.9}, 10^{-4.8}, \ldots, 10^{-0.1}, 10^{0}\}$ in dimension 10.

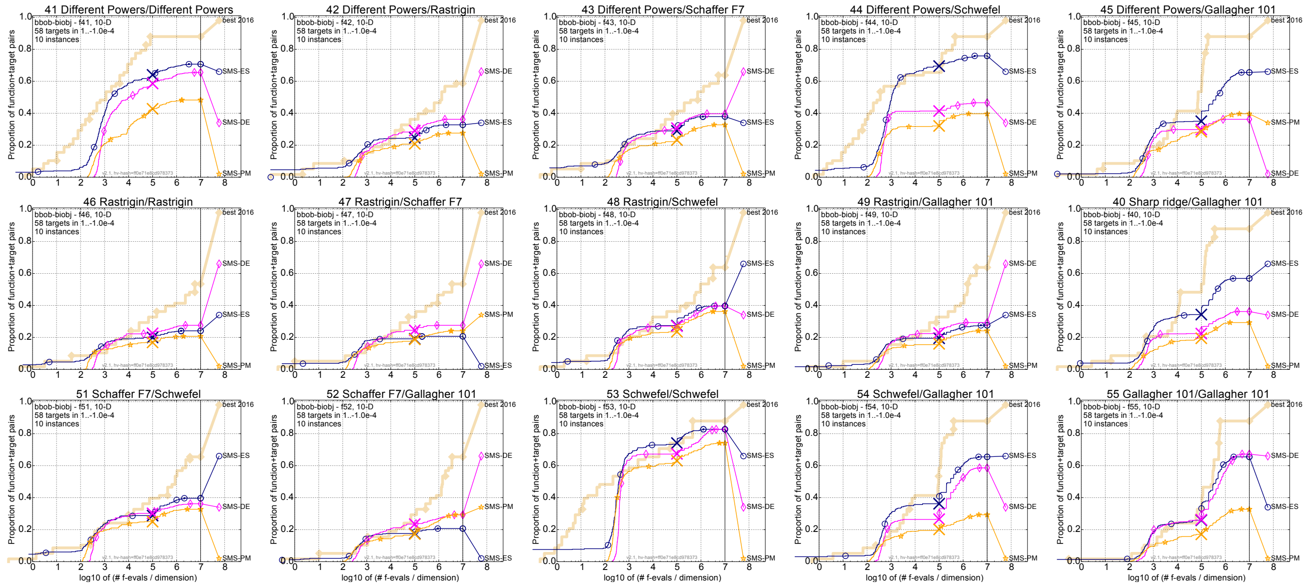

Figure 2: Bootstrapped empirical cumulative distribution of the number of objective function evaluations divided by dimension (FEvals/DIM) as in Fig. 1 but for functions f_{41} to f_{55} in 10-D.

In *Proc. Design and Optimization: Methods and Applications, (ERCOFTAC'06)*, G. Winter (Ed.). University of Las Palmas de Gran Canaria, 231–235.

[12] Kenneth Price. 1997. Differential evolution vs. the functions of the second ICEO. In *Proceedings of the IEEE International Congress on Evolutionary Computation*. 153–157.

[13] T. Tušar, D. Brockhoff, N. Hansen, and A. Auger. 2016. COCO: The Bi-objective Black-Box Optimization Benchmarking (bbob-biobj) Test Suite. *ArXiv e-prints* arXiv:1604.00359 (2016).

[14] Simon Wessing. 2017. *evoalgos: Modular evolutionary algorithms*. https://pypi.python.org/pypi/evoalgos Python package version 0.7.

[15] Simon Wessing, Nicola Beume, Günter Rudolph, and Boris Naujoks. 2010. Parameter Tuning Boosts Performance of Variation Operators in Multiobjective Optimization. In *Parallel Problem Solving from Nature – PPSN XI*. Lecture Notes in Computer Science, Vol. 6238. Springer, 728–737.

[16] Simon Wessing, Rosa Pink, Kai Brandenbusch, and Günter Rudolph. 2017. Toward Step-Size Adaptation in Evolutionary Multiobjective Optimization. In *Evolutionary Multi-Criterion Optimization, EMO 2017*. Springer, 670–684.

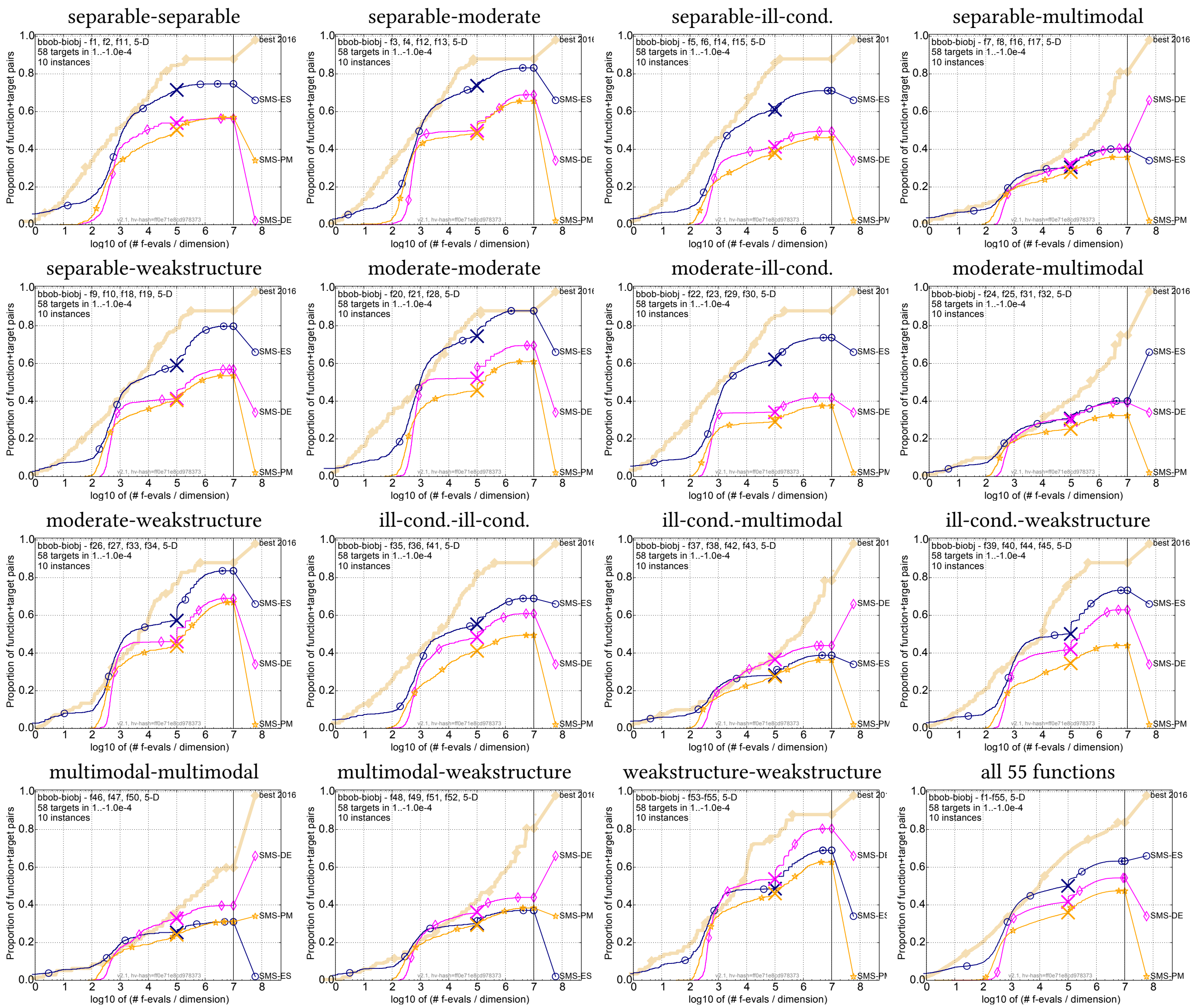

Figure 3: Bootstrapped empirical cumulative distribution of the number of objective function evaluations divided by dimension (FEvals/DIM) for 58 targets with target precision in $\{-10^{-4}, -10^{-4.2}, -10^{-4.4}, -10^{-4.6}, -10^{-4.8}, -10^{-5}, 0, 10^{-5}, 10^{-4.9}, 10^{-4.8}, \ldots, 10^{-0.1}, 10^{0}\}$ for all functions and subgroups in 5-D. The "best 2016" line corresponds to the best aRT observed during BBOB 2016 for each selected target.

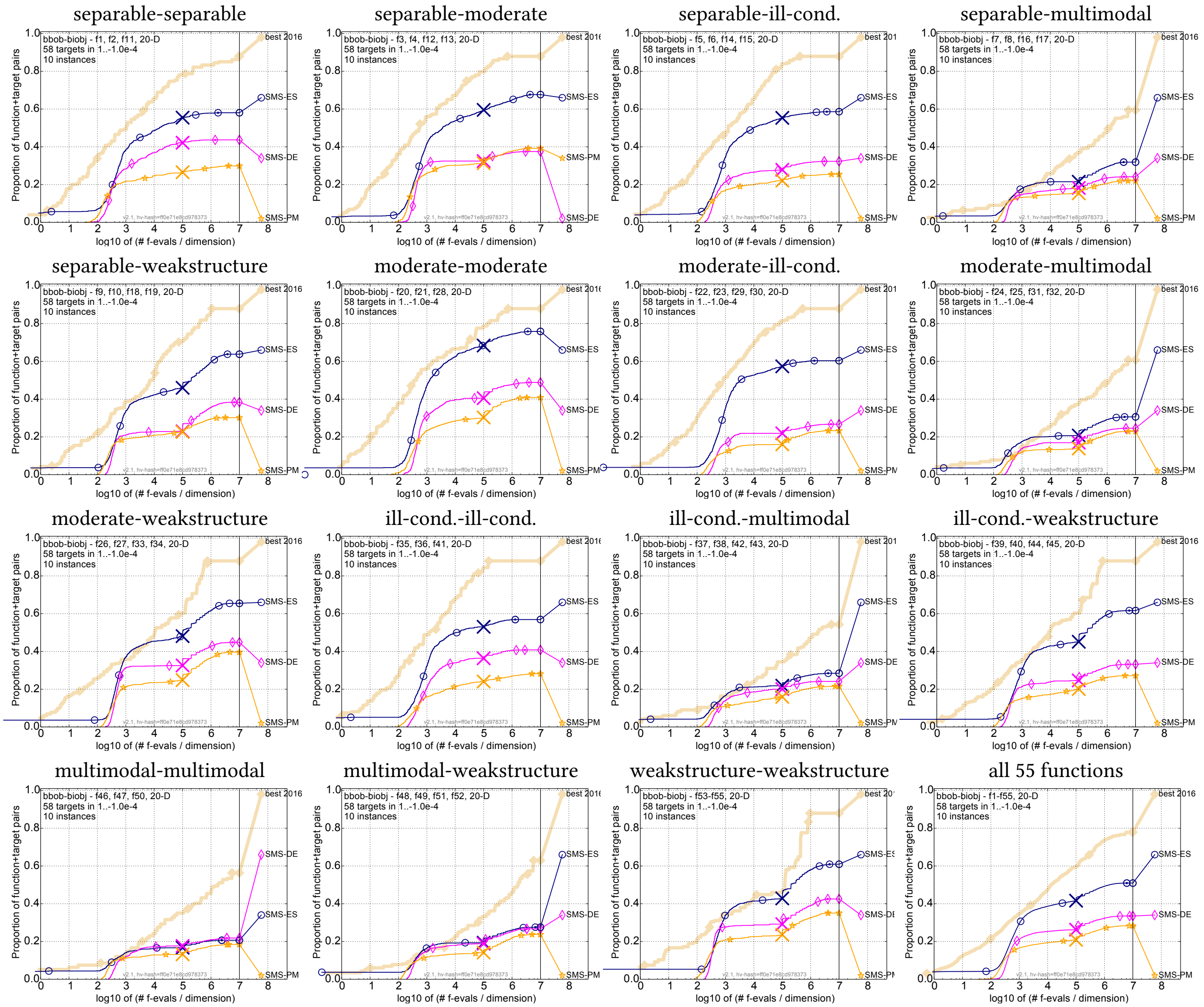

Figure 4: Bootstrapped empirical cumulative distribution of the number of objective function evaluations divided by dimension (FEvals/DIM) for 58 **targets with target precision in** $\{-10^{-4}, -10^{-4.2}, -10^{-4.4}, -10^{-4.6}, -10^{-4.8}, -10^{-5}, 0, 10^{-5}, 10^{-4.9}, 10^{-4.8}, \ldots, 10^{-0.1}, 10^{0}\}$ **for all functions and subgroups in 20-D. The "best 2016" line corresponds to the best** aRT **observed during BBOB 2016 for each selected target.**

Δf	1e0	1e-1	1e-2	1e-3	#succ
f1	1	75	584	3660	10/10
SMS-ES	**3.1**(5)$^{\star 2}$	**17**(3)	**5.3**(0.4)	2.3(0.2)	0/10
SMS-DE	1157(267)	28(2)	5.5(0.4)	**1.7**(0.1)$^{\star 2}$	10/10
SMS-PM	569(145)	20(4)	16(5)	56(32)	0/10
f2	5.0	105	601	3715	10/10
SMS-ES	**4.5**(13)$^{\star 2}$	**14**(9)	**5.5**(2)	**2.3**(0.1)$^{\star 2}$	0/10
SMS-DE	310(126)	26(2)	561(1456)	∞ *5e5*	0/10
SMS-PM	131(31)	16(4)	90(71)	∞ *5e5*	0/10
f3	3.0	115	665	5170	10/10
SMS-ES	**1.3**(0.1)$^{\star 2}$	**13**(5)	**5.0**(2)	**1.5**(0.7)*	0/10
SMS-DE	462(95)	21(4)	89(377)	874(750)	0/10
SMS-PM	263(39)	15(5)	1782(3191)	948(1185)	0/10
f4	2.0	109	571	2669	10/10
SMS-ES	**21**(103)*	**12**(4)	**5.2**(0.6)	**2.2**(0.2)$^{\star 2}$	0/10
SMS-DE	709(105)	22(2)	225(219)	∞ *5e5*	0/10
SMS-PM	357(62)	15(3)	1362(1177)	∞ *5e5*	0/10
f5	2.0	120	1277	21889	10/10
SMS-ES	**1.1**(2)$^{\star 2}$	**12**(5)*	**3.5**(0.6)	**19**(23)	0/10
SMS-DE	564(93)	20(1)	48(196)	∞ *5e5*	0/10
SMS-PM	292(76)	23(16)	49(20)	∞ *5e5*	0/10
f6	3.0	73	688	3976	10/10
SMS-ES	**10**(4)$^{\star 2}$	**21**(6)	**4.9**(0.5)	**2.0**(0.5)*	0/10
SMS-DE	507(132)	35(6)	5.7(0.6)	3.2(1)	0/10
SMS-PM	244(68)	27(5)	21(7)	50(51)	0/10
f7	2.0	2763	1.2e5	3.5e6	0/10
SMS-ES	**0.60**(0.5)$^{\star 2}$	**0.99**(0.5)	17(19)	∞ *5e5*	0/10
SMS-DE	681(94)	1.3(0.9)	**1.7**(0.3)	∞ *5e5*	0/10
SMS-PM	318(92)	2.2(2)	19(34)	∞ *5e5*	0/10
f8	3.0	2167	1.6e5	2.1e6	0/10
SMS-ES	**19**(30)$^{\star 2}$	**1.3**(0.5)	13(31)	∞ *5e5*	0/10
SMS-DE	467(74)	1.6(0.5)	**1.0**(1)	∞ *5e5*	0/10
SMS-PM	265(41)	4.8(8)	14(13)	∞ *5e5*	0/10
f9	4.0	96	521	1986	10/10
SMS-ES	**16**(39)*	**14**(4)	**5.6**(1)	**2.7**(0.4)$^{\star 2}$	0/10
SMS-DE	332(58)	22(2)	6.0(0.4)	∞ *5e5*	0/10
SMS-PM	155(60)	15(5)	64(285)	∞ *5e5*	0/10
f10	4.0	323	9839	52107	10/10
SMS-ES	**9.2**(21)$^{\star 2}$	**5.6**(1)	51(64)	39(79)	0/10
SMS-DE	331(53)	8.6(1)	**15**(15)	**15**(31)	0/10
SMS-PM	187(39)	10(3)	18(17)	25(29)	0/10
f11	5.0	56	436	9189	10/10
SMS-ES	**0.80**(1)$^{\star 2}$	**6.9**(8)	17(38)	**6.8**(14)	0/10
SMS-DE	89(99)	19(8)	**4.7**(0.4)	∞ *5e5*	0/10
SMS-PM	65(53)	16(5)	31(69)	54(56)	0/10
f12	5.0	44	641	3991	10/10
SMS-ES	**0.76**(0.7)*	**13**(23)	**2.7**(2)	**4.6**(2)	1/10
SMS-DE	147(129)	42(29)	91(196)	55(31)	1/10
SMS-PM	76(94)	1281(5693)	90(199)	56(66)	0/10
f13	7.0	60	560	5582	10/10
SMS-ES	**12**(26)*	**12**(8)	4.5(5)	**2.2**(0.4)	1/10
SMS-DE	248(50)	36(6)	5.5(2)	39(67)	0/10
SMS-PM	96(28)	18(5)	**3.2**(0.8)	60(67)	0/10
f14	5.0	247	1713	12801	10/10
SMS-ES	**1.0**(0.6)$^{\star 2}$	**9.2**(2)	**37**(74)$^{\star 2}$	**45**(39)$^{\star 2}$	0/10
SMS-DE	269(77)	882(1521)	∞	∞ *5e5*	0/10
SMS-PM	148(62)	1443(1644)	∞	∞ *5e5*	0/10
f15	6.0	74	677	6559	10/10
SMS-ES	**22**(54)*	**21**(20)	**7.2**(3)*	**12**(23)$^{\star 2}$	0/10
SMS-DE	300(118)	795(3382)	1731(1478)	∞ *5e5*	0/10
SMS-PM	145(63)	28(10)	1732(1847)	∞ *5e5*	0/10
f16	3.0	2810	2.0e5	4.2e6	3/10
SMS-ES	**1.7**(3)$^{\star 2}$	**21**(1)	**6.0**(5)	∞ *5e5*	0/10
SMS-DE	541(169)	46(45)	∞	∞ *5e5*	0/10
SMS-PM	238(95)	26(59)	∞	∞ *5e5*	0/10
f17	29	2935	48624	2.4e6	3/10
SMS-ES	**6.7**(1.0)	13(31)	**16**(23)	∞ *5e5*	0/10
SMS-DE	55(16)	**1.5**(0.6)	93(69)	∞ *5e5*	0/10
SMS-PM	25(10)	2.9(0.5)	96(159)	∞ *5e5*	0/10
f18	2.0	56	557	6912	8/10
SMS-ES	**4.8**(6)$^{\star 2}$	**9.2**(10)	**3.4**(2)	**1.6**(2)	0/10
SMS-DE	654(144)	36(5)	104(225)	290(543)	0/10
SMS-PM	304(127)	20(5)	13(20)	316(452)	0/10
f19	9.0	1292	6164	88464	9/10
SMS-ES	**8.8**(21)	**3.4**(0.3)	**82**(264)	**8.8**(10)	0/10
SMS-DE	154(34)	45(1.0)	731(1663)	∞ *5e5*	0/10
SMS-PM	72(16)	48(194)	325(385)	∞ *5e5*	0/10
f20	4.0	70	1162	5724	10/10
SMS-ES	**27**(64)*	**17**(13)	**2.5**(2)	**1.5**(2)	2/10
SMS-DE	431(240)	41(13)	51(215)	205(197)	0/10
SMS-PM	268(149)	34(12)	304(230)	358(655)	0/10
f21	5.0	86	3249	9924	10/10
SMS-ES	**24**(54)*	**15**(14)	**18**(77)	**6.1**(13)	2/10
SMS-DE	344(85)	29(7)	39(0.3)	76(88)	0/10
SMS-PM	184(70)	22(13)	22(77)	60(150)	0/10
f22	3.0	97	1168	12608	9/10
SMS-ES	**25**(0.1)*	**17**(9)	**5.2**(3)$^{\star 2}$	**32**(36)$^{\star 2}$	0/10
SMS-DE	505(79)	31(8)	3867(7705)	∞ *5e5*	0/10
SMS-PM	258(88)	631(78)	∞	∞ *5e5*	0/10
f23	1	59	618	4410	10/10
SMS-ES	**1.4**(2)$^{\star 2}$	**18**(8)*	**5.1**(1)	**1.6**(0.4)$^{\star 2}$	0/10
SMS-DE	1683(410)	48(4)	354(810)	∞ *5e5*	0/10
SMS-PM	1268(327)	117(184)	1895(1824)	∞ *5e5*	0/10
f24	5.0	2347	1.8e5	4.1e6	0/10
SMS-ES	**1.1**(2)$^{\star 2}$	**1.4**(0.9)	**25**(38)	∞ *5e5*	0/10
SMS-DE	380(75)	2.2(1)	26(53)	∞ *5e5*	0/10
SMS-PM	224(59)	56(107)	∞	∞ *5e5*	0/10
f25	9.0	3143	1.2e5	2.5e6	3/10
SMS-ES	**10**(24)*	**0.84**(0.6)	6.4(5)	∞ *5e5*	0/10
SMS-DE	182(57)	1.3(0.7)	**3.4**(7)	∞ *5e5*	0/10
SMS-PM	105(37)	3.2(3)	37(35)	∞ *5e5*	0/10
f26	7.0	59	2182	13673	9/10
SMS-ES	**25**(39)*	**18**(9)	26(0.7)	**4.4**(18)	0/10
SMS-DE	266(59)	40(10)	**1.6**(0.4)	25(18)	0/10
SMS-PM	143(45)	31(9)	29(173)	41(70)	0/10
f27	6.0	2631	21971	44576	10/10
SMS-ES	**22**(104)	**1.4**(0.3)	34(40)	45(73)	1/10
SMS-DE	274(122)	22(95)	**15**(46)	**26**(20)	0/10
SMS-PM	142(65)	26(56)	91(80)	101(109)	0/10
f28	2.0	20	145	1230	10/10
SMS-ES	**15**(35)$^{\star 2}$	**36**(31)	**14**(3)	**3.5**(0.8)	3/10
SMS-DE	727(105)	111(15)	21(3)	631(1423)	0/10
SMS-PM	452(137)	86(57)	261(440)	∞ *5e5*	0/10
f29	3.0	114	1413	13660	10/10
SMS-ES	**1.5**(0.2)$^{\star 2}$	**19**(7)	**45**(265)*	**64**(59)*	0/10
SMS-DE	444(152)	25(3)	∞	∞ *5e5*	0/10
SMS-PM	233(64)	193(424)	∞	∞ *5e5*	0/10
f30	1	33	366	2294	10/10
SMS-ES	**3.6**(6)$^{\star 2}$	**39**(21)	**8.2**(1)	**2.8**(0.5)$^{\star 2}$	0/10
SMS-DE	1634(326)	72(10)	162(684)	∞ *5e5*	0/10
SMS-PM	833(173)	51(19)	651(2051)	∞ *5e5*	0/10
f31	3.0	2166	50028	3.2e6	0/10
SMS-ES	**31**(12)*	27(58)	90(40)	∞ *5e5*	0/10
SMS-DE	516(88)	**2.0**(0.7)	**41**(53)	∞ *5e5*	0/10
SMS-PM	279(67)	5.7(8)	∞	∞ *5e5*	0/10
f32	3.0	2131	1.1e5	2.4e6	2/10
SMS-ES	**1.9**(3)$^{\star 2}$	**0.88**(0.4)	**2.4**(2)	∞ *5e5*	0/10
SMS-DE	491(100)	1.3(0.4)	6.9(7)	∞ *5e5*	0/10
SMS-PM	239(70)	1.1(0.5)	∞	∞ *5e5*	0/10
f33	6.0	627	1214	3730	3/10
SMS-ES	**42**(40)	**90**(1)	60(206)	**20**(67)	1/10
SMS-DE	263(47)	92(798)	105(206)	35(67)	0/10
SMS-PM	142(85)	91(0.6)	**55**(225)	23(4)	0/10
f34	9.0	1376	15575	54195	10/10
SMS-ES	**1.3**(2)$^{\star 2}$	**1.2**(0.2)	**8.3**(0.1)	**9.5**(14)	0/10
SMS-DE	170(47)	1.9(0.3)	14(40)	83(125)	0/10
SMS-PM	83(21)	2.4(0.8)	76(97)	∞ *5e5*	0/10
f35	1	141	2268	51388	9/10
SMS-ES	**4.0**(8)$^{\star 2}$	**12**(4)	**57**(0.6)	**17**(22)	0/10
SMS-DE	1059(168)	18(1)	362(494)	∞ *5e5*	0/10
SMS-PM	560(206)	28(8)	944(1047)	∞ *5e5*	0/10
f36	2.0	204	4640	41237	10/10
SMS-ES	**2.1**(2)$^{\star 2}$	**10**(1)	**14**(0.6)	49(42)	0/10
SMS-DE	680(223)	15(2)	43(54)	**19**(30)	0/10
SMS-PM	383(60)	16(18)	98(168)	∞ *5e5*	0/10
f37	2.0	3956	1.5e5	2.5e6	1/10
SMS-ES	**1.4**(2)$^{\star 2}$	**0.94**(0.7)	31(45)	∞ *5e5*	0/10
SMS-DE	685(212)	1.2(0.4)	**4.2**(3)	∞ *5e5*	0/10
SMS-PM	324(111)	14(30)	32(18)	∞ *5e5*	0/10
f38	4.0	4366	2.7e5	5.4e6	2/10
SMS-ES	**22**(53)*	**1.1**(0.9)	∞	∞ *5e5*	0/10
SMS-DE	361(123)	1.3(0.4)	**1.5**(2)*	∞ *5e5*	0/10
SMS-PM	182(69)	17(36)	∞	∞ *5e5*	0/10
f39	2.0	215	1160	47472	8/10
SMS-ES	**1.1**(2)$^{\star 2}$	**7.1**(2)	**112**(108)	**42**(103)	0/10
SMS-DE	665(85)	12(0.9)	1010(2695)	95(111)	0/10
SMS-PM	355(112)	11(3)	∞	∞ *5e5*	0/10
f40	2.0	998	34442	2.0e5	8/10
SMS-ES	**3.9**(6)$^{\star 2}$	**2.5**(0.6)	15(22)	**10**(23)	0/10
SMS-DE	691(141)	3.3(0.6)	**7.3**(7)	∞ *5e5*	0/10
SMS-PM	351(100)	70(142)	69(127)	∞ *5e5*	0/10
f41	2.0	48	708	6343	10/10
SMS-ES	**36**(3)$^{\star 2}$	**27**(22)	**4.9**(2)	**1.4**(0.5)	0/10
SMS-DE	971(346)	60(13)	7.0(1.0)	3.7(2)	0/10
SMS-PM	482(123)	57(23)	27(8)	25(23)	0/10
f42	2.0	2525	3.4e5	4.3e6	0/10
SMS-ES	**8.8**(20)$^{\star 2}$	23(99)	13(28)	∞ *5e5*	0/10
SMS-DE	863(221)	**1.8**(1)	**2.1**(3)	∞ *5e5*	0/10
SMS-PM	419(116)	10(4)	14(18)	∞ *5e5*	0/10
f43	4.0	2468	1.5e5	3.8e6	2/10
SMS-ES	**12**(29)*	**0.92**(0.7)	8.1(12)	∞ *5e5*	0/10
SMS-DE	416(86)	1.4(0.5)	**1.3**(2)	∞ *5e5*	0/10
SMS-PM	210(114)	3.7(5)	8.6(12)	∞ *5e5*	0/10
f44	6.0	86	513	1853	10/10
SMS-ES	**30**(74)	**16**(7)	**6.1**(1)	**33**(0.6)	0/10
SMS-DE	288(86)	29(5)	7.2(2)	635(405)	0/10
SMS-PM	163(80)	21(11)	374(516)	2522(3103)	0/10
f45	4.0	816	5730	53653	10/10
SMS-ES	**1.6**(0.9)$^{\star 2}$	**2.5**(0.3)	39(87)	22(7)	0/10
SMS-DE	347(90)	3.9(0.5)	40(65)	**16**(26)	0/10
SMS-PM	213(75)	27(8)	**32**(31)	27(35)	0/10
f46	4.0	11925	4.9e5	5.6e7	0/10
SMS-ES	**12**(4)$^{\star 2}$	14(22)	∞	∞ *5e5*	0/10
SMS-DE	424(101)	**1.0**(0.7)	∞	∞ *5e5*	0/10
SMS-PM	244(68)	14(14)	∞	∞ *5e5*	0/10
f47	3.0	4172	2.7e5	4.4e6	0/10
SMS-ES	**0.57**(0.8)$^{\star 2}$	**1.6**(2)	∞	∞ *5e5*	0/10
SMS-DE	468(68)	1.7(0.9)	**5.0**(7)	∞ *5e5*	0/10
SMS-PM	246(27)	23(56)	∞	∞ *5e5*	0/10
f48	9.0	2160	1.1e5	2.1e6	2/10
SMS-ES	**16**(40)	27(59)	18(12)	∞ *5e5*	0/10
SMS-DE	174(54)	**2.5**(4)	**18**(32)	∞ *5e5*	0/10
SMS-PM	97(37)	5.6(10)	20(16)	∞ *5e5*	0/10
f49	9.0	3737	2.0e5	1.3e6	1/10
SMS-ES	**13**(1)	2.6(9)	∞	∞ *5e5*	0/10
SMS-DE	161(55)	**1.3**(0.4)	**5.9**(4)	∞ *5e5*	0/10
SMS-PM	96(41)	10(13)	23(34)	∞ *5e5*	0/10
f50	6.0	3913	2.2e5	2.6e6	1/10
SMS-ES	**18**(39)*	**0.84**(0.5)	20(39)	∞ *5e5*	0/10
SMS-DE	238(49)	1.3(1)	**1.2**(2)	∞ *5e5*	0/10
SMS-PM	126(64)	21(64)	22(23)	∞ *5e5*	0/10
f51	8.0	1251	32465	2.5e6	1/10
SMS-ES	**2.6**(0.8)$^{\star 2}$	46(300)	7.6(5)	∞ *5e5*	0/10
SMS-DE	176(15)	**2.3**(0.7)	**3.1**(5)	∞ *5e5*	0/10
SMS-PM	100(41)	49(300)	39(43)	∞ *5e5*	0/10
f52	2.0	3027	1.4e5	2.1e6	1/10
SMS-ES	**2.5**(3)$^{\star 2}$	**0.83**(0.1)	14(12)	∞ *5e5*	0/10
SMS-DE	629(175)	1.1(0.4)	**2.6**(2)	∞ *5e5*	0/10
SMS-PM	345(126)	17(75)	34(24)	∞ *5e5*	0/10
f53	2.0	13	42	1443	2/10
SMS-ES	**2.8**(12)$^{\star 2}$	**24**(19)$^{\star 2}$	**28**(9)	40(87)	0/10
SMS-DE	651(95)	146(7)	53(8)	**2.2**(1)	1/10
SMS-PM	376(104)	83(22)	38(9)	40(261)	0/10
f54	98	1514	12856	45502	10/10
SMS-ES	**3.6**(9)	**1.2**(0.7)	**10**(19)	**17**(11)	0/10
SMS-DE	15(2)	1.7(0.3)	26(39)	44(58)	0/10
SMS-PM	7.8(4)	1.7(1)	163(237)	∞ *5e5*	0/10
f55	3.0	1415	18086	51245	9/10
SMS-ES	**9.1**(0.8)$^{\star 2}$	41(177)	249(325)	∞ *5e5*	0/10
SMS-DE	318(65)	**2.7**(1)	**29**(29)	**23**(32)	1/10
SMS-PM	224(47)	10(32)	48(128)	88(93)	0/10

Table 1: Average runtime (aRT in number of function evaluations) divided by the respective best aRT measured during BBOB-2016 in dimension 5. The aRT and in braces, as dispersion measure, the half difference between 10 and 90%-tile of bootstrapped run lengths appear for each algorithm and target, the corresponding reference aRT in the first row. The different target $\Delta I_{\mathrm{HV}}^{\mathrm{COCO}}$-values are shown in the top row. #succ is the number of trials that reached the (final) target $I_{\mathrm{ref}} + 10^{-5}$. The median number of conducted function evaluations is additionally given in *italics*, if the target in the last column was never reached. Entries, succeeded by a star, are statistically significantly better (according to the rank-sum test) when compared to all other algorithms of the table, with $p = 0.05$ or $p = 10^{-k}$ when the number k following the star is larger than 1, with Bonferroni correction of 110. A ↓ indicates the same tested against the best algorithm from BBOB 2016. Best results are printed in bold.

Data produced with COCO v2.1, hv-hash=ff0e71e8cd978373

Δf	1e0	1e-1	1e-2	1e-3	#succ
f1	1	157	1468	8244	10/10
SMS-ES	**1**(0)$^{\star 2}$	47(6)	10(0.6)	135(21)	0/10
SMS-DE	3698(353)	**44**(5)	**9.4**(0.9)	**6.0**(0.4)$^{\star 2}$	0/10
SMS-PM	2060(374)	63(19)	∞	∞ *2e6*	0/10
f2	5.0	163	2170	21153	10/10
SMS-ES	**36**(180)$^{\star 2}$	**43**(11)	**7.2**(1.0)$^{\star 2}$	**10**(5)$^{\star 2}$	0/10
SMS-DE	845(75)	2.9e4(2e4)	∞	∞ *2e6*	0/10
SMS-PM	505(64)	54(14)	∞	∞ *2e6*	0/10
f3	1	216	2384	11104	10/10
SMS-ES	**1**(0)$^{\star 2}$	**36**(6)	**6.0**(0.7)$^{\star 2}$	**8.0**(4)$^{\star 2}$	0/10
SMS-DE	6675(926)	49(4)	∞	∞ *2e6*	0/10
SMS-PM	3860(294)	9314(6944)	∞	∞ *2e6*	0/10
f4	1	172	1682	10200	9/10
SMS-ES	**1**(0)$^{\star 2}$	**42**(7)	**8.3**(0.7)$^{\star 2}$	**6.3**(1)$^{\star 2}$	0/10
SMS-DE	4535(231)	44(5)	∞	∞ *2e6*	0/10
SMS-PM	2603(412)	43(13)	∞	∞ *2e6*	0/10
f5	1	190	3207	32860	10/10
SMS-ES	**1**(0)$^{\star 2}$	48(6)	**7.0**(1)*	**604**(654)$^{\star 2}$	0/10
SMS-DE	3855(327)	**43**(4)	113(91)	∞ *2e6*	0/10
SMS-PM	2092(90)	279(134)	∞	∞ *2e6*	0/10
f6	3.0	236	2134	16568	10/10
SMS-ES	**90**(224)*	**31**(5)	**7.2**(1)*	**10**(4)*	0/10
SMS-DE	1686(221)	37(4)	11(3)	208(228)	0/10
SMS-PM	922(174)	49(13)	∞	∞ *2e6*	0/10
f7	1	24981	6.8e5	3.8e7	0/10
SMS-ES	**1**(0)$^{\star 2}$	**0.80**(0.4)	**27**(35)	∞ *2e6*	0/10
SMS-DE	4712(281)	1.5(2)	∞	∞ *2e6*	0/10
SMS-PM	2580(210)	90(100)	∞	∞ *2e6*	0/10
f8	4.0	16448	2.4e6	2.8e7	0/10
SMS-ES	**106**(0)	53(61)	∞	∞ *2e6*	0/10
SMS-DE	1184(298)	**24**(32)	∞	∞ *2e6*	0/10
SMS-PM	674(195)	545(763)	∞	∞ *2e6*	0/10
f9	1	144	1468	5944	10/10
SMS-ES	**1**(0)$^{\star 2}$	49(9)	**8.6**(0.7)	**4.2**(0.4)$^{\star 2}$	0/10
SMS-DE	4079(169)	44(1)	5457(1e4)	∞ *2e6*	0/10
SMS-PM	2344(273)	**37**(5)$^{\star 2}$	∞	∞ *2e6*	0/10
f10	1	6124	2.0e5	2.4e5	10/10
SMS-ES	**1**(0)$^{\star 2}$	**84**(163)	**6.7**(12)	78(66)	0/10
SMS-DE	4000(378)	85(84)	15(35)	**74**(86)	0/10
SMS-PM	2133(228)	221(245)	∞	∞ *2e6*	0/10
f11	3.0	246	3177	1.1e5	2/10
SMS-ES	**0.67**(0)$^{\star 2}$	14(7)	**8.3**(10)$^{\star 2}$	**20**(25)$^{\star 2}$	0/10
SMS-DE	634(115)	17(2)	∞	∞ *2e6*	0/10
SMS-PM	380(95)	**14**(3)	6219(5666)	∞ *2e6*	0/10
f12	11	131	1483	11435	10/10
SMS-ES	**10**(49)$^{\star 2}$	**31**(4)	12(5)	**34**(33)	0/10
SMS-DE	374(158)	65(25)	**10**(4)	1576(918)	0/10
SMS-PM	206(77)	58(33)	37(63)	1577(2143)	0/10
f13	7.0	139	1087	17899	10/10
SMS-ES	**61**(69)$^{\star 2}$	39(9)	**10**(1)	**7.4**(5)$^{\star 2}$	0/10
SMS-DE	713(82)	1649(3599)	7368(1e4)	∞ *2e6*	0/10
SMS-PM	458(94)	**36**(5)	1495(2661)	∞ *2e6*	0/10
f14	3.0	349	6102	63789	10/10
SMS-ES	**0.63**(0)$^{\star 2}$	**31**(2)$^{\star 2}$	**5.7**(0.4)$^{\star 2}$	**33**(25)$^{\star 2}$	0/10
SMS-DE	1520(182)	∞	∞	∞ *2e6*	0/10
SMS-PM	900(220)	∞	∞	∞ *2e6*	0/10
f15	6.0	210	2099	25094	10/10
SMS-ES	**53**(0.6)*	**33**(9)*	**8.2**(3)$^{\star 2}$	**8.8**(10)$^{\star 2}$	0/10
SMS-DE	1160(245)	2.2e4(4e4)	∞	∞ *2e6*	0/10
SMS-PM	889(548)	4152(12)	∞	∞ *2e6*	0/10
f16	6.0	32242	1.0e6	1.9e7	0/10
SMS-ES	**11**(3)$^{\star 2}$	**27**(62)*	∞	∞ *2e6*	0/10
SMS-DE	1138(248)	∞	∞	∞ *2e6*	0/10
SMS-PM	701(103)	∞	∞	∞ *2e6*	0/10
f17	1	33890	9.3e5	2.7e7	0/10
SMS-ES	**1**(0)$^{\star 2}$	**89**(133)	∞	∞ *2e6*	0/10
SMS-DE	5528(405)	236(384)	∞	∞ *2e6*	0/10
SMS-PM	3547(545)	138(148)	∞	∞ *2e6*	0/10
f18	6.0	141	1507	17541	10/10
SMS-ES	**1.6**(2)$^{\star 2}$	**37**(11)	**6.9**(2)	**6.1**(5)$^{\star 2}$	0/10
SMS-DE	802(48)	2.1e4(4e4)	1.2e4(2e4)	∞ *2e6*	0/10
SMS-PM	496(151)	40(16)	5974(2307)	∞ *2e6*	0/10
f19	7.0	18889	2.4e5	5.1e6	6/10
SMS-ES	**66**(326)	**160**(185)	**75**(85)	**3.5**(4)	0/10
SMS-DE	714(99)	∞	∞	∞ *2e6*	0/10
SMS-PM	435(70)	∞	∞	∞ *2e6*	0/10

Δf	1e0	1e-1	1e-2	1e-3	#succ
f20	3.0	136	1870	8641	10/10
SMS-ES	**22**(0)$^{\star 2}$	**34**(12)	**5.0**(2)*	**5.3**(3)	0/10
SMS-DE	1844(1349)	788(1732)	293(273)	255(338)	0/10
SMS-PM	1086(786)	3644(1e4)	1626(3478)	2094(1099)	0/10
f21	5.0	331	1753	9785	8/10
SMS-ES	**155**(185)$^{\star 2}$	**20**(4)*	**7.2**(2)	**5.9**(14)$^{\star 2}$	0/10
SMS-DE	1537(293)	31(4)	497(571)	∞ *2e6*	0/10
SMS-PM	1162(204)	109(191)	∞	∞ *2e6*	0/10
f22	1	313	4469	23797	10/10
SMS-ES	**1**(0)$^{\star 2}$	**33**(4)$^{\star 2}$	**5.5**(1)$^{\star 2}$	**64**(80)$^{\star 2}$	0/10
SMS-DE	6929(1322)	2788(4795)	∞	∞ *2e6*	0/10
SMS-PM	4826(1563)	∞	∞	∞ *2e6*	0/10
f23	3.0	332	2660	16975	10/10
SMS-ES	**35**(0)$^{\star 2}$	**25**(5)$^{\star 2}$	**6.8**(1)$^{\star 2}$	**6.1**(7)$^{\star 2}$	0/10
SMS-DE	6621(663)	788(3031)	∞	∞ *2e6*	0/10
SMS-PM	5276(2825)	6389(1e4)	∞	∞ *2e6*	0/10
f24	1	61504	8.1e5	6.5e7	0/10
SMS-ES	**166**(824)$^{\star 2}$	**14**(49)	∞	∞ *2e6*	0/10
SMS-DE	1.1e4(1583)	50(66)	∞	∞ *2e6*	0/10
SMS-PM	1.2e4(3733)	293(260)	∞	∞ *2e6*	0/10
f25	1	34160	9.2e5	1.2e7	0/10
SMS-ES	**1**(0)$^{\star 2}$	**25**(15)	**20**(22)	∞ *2e6*	0/10
SMS-DE	8568(867)	40(59)	∞	∞ *2e6*	0/10
SMS-PM	5990(988)	88(132)	∞	∞ *2e6*	0/10
f26	3.0	136	1117	4425	9/10
SMS-ES	**118**(295)$^{\star 2}$	**49**(14)	**10**(3)	**4.7**(3)	2/10
SMS-DE	2316(422)	71(9)	12(3)	681(1017)	0/10
SMS-PM	1726(435)	88(69)	2715(2696)	1092(1356)	0/10
f27	1	60235	1.3e6	6.4e6	7/10
SMS-ES	**610**(3046)	**50**(58)	**14**(22)	∞ *2e6*	0/10
SMS-DE	7077(794)	133(133)	∞	∞ *2e6*	0/10
SMS-PM	4869(810)	299(307)	∞	∞ *2e6*	0/10
f28	1	62	539	6898	9/10
SMS-ES	**1**(0)$^{\star 2}$	**97**(16)	**21**(4)	**6.1**(4)$^{\star 2}$	0/10
SMS-DE	5337(398)	117(17)	950(1857)	∞ *2e6*	0/10
SMS-PM	3326(210)	100(30)	∞	∞ *2e6*	0/10
f29	1	359	3841	30739	9/10
SMS-ES	**1**(0)$^{\star 2}$	**29**(1)	**6.4**(1)$^{\star 2}$	**57**(58)$^{\star 2}$	0/10
SMS-DE	4853(247)	2.2e4(3e4)	∞	∞ *2e6*	0/10
SMS-PM	2842(279)	∞	∞	∞ *2e6*	0/10
f30	1	220	1777	10156	9/10
SMS-ES	**1.4**(1)$^{\star 2}$	**34**(8)	**8.9**(0.6)$^{\star 2}$	**5.0**(4)$^{\star 2}$	0/10
SMS-DE	5888(772)	43(11)	∞	∞ *2e6*	0/10
SMS-PM	3655(649)	2316(13)	1.0e4(7597)	∞ *2e6*	0/10
f31	3.0	27722	8.6e5	3.0e7	0/10
SMS-ES	**196**(267)*	**49**(144)	∞	∞ *2e6*	0/10
SMS-DE	1971(335)	289(379)	∞	∞ *2e6*	0/10
SMS-PM	1122(87)	650(794)	∞	∞ *2e6*	0/10
f32	6.0	32985	1.1e6	2.6e7	0/10
SMS-ES	**174**(404)	**42**(46)	∞	∞ *2e6*	0/10
SMS-DE	917(94)	142(258)	∞	∞ *2e6*	0/10
SMS-PM	535(128)	142(273)	∞	∞ *2e6*	0/10
f33	1	33	278	3348	8/10
SMS-ES	**1**(0)$^{\star 2}$	177(22)	33(3)	**6.7**(4)$^{\star 2}$	0/10
SMS-DE	4720(501)	191(10)	**31**(2)	∞ *2e6*	0/10
SMS-PM	2902(536)	**147**(15)	107(128)	∞ *2e6*	0/10
f34	3.0	45588	1.9e5	2.3e5	8/10
SMS-ES	**122**(0)	**5.2**(0.3)	**25**(29)	**79**(87)	0/10
SMS-DE	1536(109)	103(154)	95(95)	∞ *2e6*	0/10
SMS-PM	945(153)	395(384)	∞	∞ *2e6*	0/10
f35	1	338	4782	41429	10/10
SMS-ES	**1**(0)$^{\star 2}$	**28**(3)	**5.3**(0.3)$^{\star 2}$	∞ *2e6*	0/10
SMS-DE	4106(713)	29(2)	∞	∞ *2e6*	0/10
SMS-PM	2122(220)	4039(7736)	∞	∞ *2e6*	0/10
f36	1	454	4824	48159	10/10
SMS-ES	**1**(0)$^{\star 2}$	**23**(5)	**6.0**(1)*	**78**(63)$^{\star 2}$	0/10
SMS-DE	5357(641)	31(11)	652(847)	∞ *2e6*	0/10
SMS-PM	3476(391)	225(221)	∞	∞ *2e6*	0/10
f37	1	10528	5.2e5	3.5e7	0/10
SMS-ES	**1**(0)$^{\star 2}$	**2.1**(1)	∞	∞ *2e6*	0/10
SMS-DE	5515(1640)	5.4(5)	∞	∞ *2e6*	0/10
SMS-PM	2654(563)	458(431)	∞	∞ *2e6*	0/10

Δf	1e0	1e-1	1e-2	1e-3	#succ
f38	3.0	36705	3.9e6	2.2e8	0/10
SMS-ES	**0.50**(0.4)$^{\star 2}$	**26**(27)	∞	∞ *2e6*	0/10
SMS-DE	1706(166)	86(191)	∞	∞ *2e6*	0/10
SMS-PM	1009(107)	250(572)	∞	∞ *2e6*	0/10
f39	1	457	3876	25958	10/10
SMS-ES	**1**(0)$^{\star 2}$	**21**(2)	**7.0**(1)$^{\star 2}$	**43**(60)$^{\star 2}$	0/10
SMS-DE	4127(653)	508(2192)	∞	∞ *2e6*	0/10
SMS-PM	2425(257)	1.8e4(1e4)	∞	∞ *2e6*	0/10
f40	1	26134	2.1e5	2.6e6	7/10
SMS-ES	**1**(0)$^{\star 2}$	**9.2**(0.2)	**23**(12)	**7.5**(8)	0/10
SMS-DE	3828(234)	13(47)	∞	∞ *2e6*	0/10
SMS-PM	2130(316)	328(134)	∞	∞ *2e6*	0/10
f41	2.0	245	2389	15818	10/10
SMS-ES	**179**(0)*	**32**(5)*	**7.6**(0.9)$^{\star 2}$	**4.9**(2)$^{\star 2}$	0/10
SMS-DE	5379(1870)	85(17)	30(17)	∞ *2e6*	0/10
SMS-PM	3290(1216)	88(49)	7797(8790)	∞ *2e6*	0/10
f42	1	34578	2.9e6	2.3e7	0/10
SMS-ES	**1**(0)$^{\star 2}$	7.2(15)	∞	∞ *2e6*	0/10
SMS-DE	8267(1050)	**5.6**(12)	∞	∞ *2e6*	0/10
SMS-PM	5042(595)	142(131)	∞	∞ *2e6*	0/10
f43	2.0	55957	2.3e6	3.9e7	0/10
SMS-ES	**1.4**(0)$^{\star 2}$	**16**(27)	∞	∞ *2e6*	0/10
SMS-DE	3108(330)	58(102)	∞	∞ *2e6*	0/10
SMS-PM	2120(252)	148(170)	∞	∞ *2e6*	0/10
f44	3.0	168	1485	6584	10/10
SMS-ES	**216**(305)	**39**(14)	**9.1**(3)	**4.3**(2)$^{\star 2}$	0/10
SMS-DE	1726(246)	49(10)	3151(4378)	∞ *2e6*	0/10
SMS-PM	989(151)	1361(15)	1.2e4(1e4)	∞ *2e6*	0/10
f45	1	49797	2.4e6	5.2e6	7/10
SMS-ES	**3.5**(6)$^{\star 2}$	17(20)	**1.9**(2)	**1.6**(0.6)	0/10
SMS-DE	5408(316)	**5.6**(2)	7.5(9)	∞ *2e6*	0/10
SMS-PM	3163(326)	28(32)	∞	∞ *2e6*	0/10
f46	1	55141	1.0e6	1.1e8	0/10
SMS-ES	**1**(0)$^{\star 2}$	28(18)	∞	∞ *2e6*	0/10
SMS-DE	7415(1583)	**18**(27)	∞	∞ *2e6*	0/10
SMS-PM	4104(716)	∞	∞	∞ *2e6*	0/10
f47	2.0	54618	2.2e6	5.2e7	0/10
SMS-ES	**0.65**(0)$^{\star 2}$	90(75)	∞	∞ *2e6*	0/10
SMS-DE	3194(385)	**87**(92)	∞	∞ *2e6*	0/10
SMS-PM	1771(329)	330(476)	∞	∞ *2e6*	0/10
f48	1	22301	3.5e5	3.3e7	0/10
SMS-ES	**1**(0)$^{\star 2}$	11(23)	∞	∞ *2e6*	0/10
SMS-DE	4989(854)	**1.8**(1)	∞	∞ *2e6*	0/10
SMS-PM	2841(809)	223(426)	∞	∞ *2e6*	0/10
f49	2.0	1.6e5	1.1e7	7.1e7	0/10
SMS-ES	**0.65**(0.4)$^{\star 2}$	30(40)	∞	∞ *2e6*	0/10
SMS-DE	2409(307)	**5.8**(9)	∞	∞ *2e6*	0/10
SMS-PM	1346(194)	54(38)	∞	∞ *2e6*	0/10
f50	3.0	22169	1.3e6	2.0e7	0/10
SMS-ES	**0.93**(2)$^{\star 2}$	**212**(293)	∞	∞ *2e6*	0/10
SMS-DE	2692(4529)	365(543)	∞	∞ *2e6*	0/10
SMS-PM	1062(125)	∞	∞	∞ *2e6*	0/10
f51	1	8574	9.9e5	2.2e7	0/10
SMS-ES	**1**(0)$^{\star 2}$	235(350)	∞	∞ *2e6*	0/10
SMS-DE	4426(428)	**161**(179)	∞	∞ *2e6*	0/10
SMS-PM	2713(447)	596(499)	∞	∞ *2e6*	0/10
f52	1	82166	4.9e6	2.4e7	1/10
SMS-ES	**1**(0)$^{\star 2}$	**11**(24)	∞	∞ *2e6*	0/10
SMS-DE	4388(405)	28(19)	∞	∞ *2e6*	0/10
SMS-PM	2694(390)	43(31)	∞	∞ *2e6*	0/10
f53	1	28	159	1952	8/10
SMS-ES	**1**(0)$^{\star 2}$	167(29)	47(5)	**5.9**(1)	1/10
SMS-DE	4138(341)	202(10)	**44**(3)	261(256)	0/10
SMS-PM	2507(248)	**147**(23)	1808(4125)	2401(2305)	0/10
f54	7.0	6049	1.6e5	2.5e6	8/10
SMS-ES	**177**(618)	**144**(166)	**12**(15)	**3.2**(2)	0/10
SMS-DE	620(43)	222(248)	∞	∞ *2e6*	0/10
SMS-PM	369(36)	1330(1243)	∞	∞ *2e6*	0/10
f55	1	8.4e5	8.8e6	8.8e6	7/10
SMS-ES	**1**(0)$^{\star 2}$	**10**(15)	∞	∞ *2e6*	0/10
SMS-DE	3955(242)	21(14)	**2.1**(3)	∞ *2e6*	0/10
SMS-PM	1785(55)	22(18)	∞	∞ *2e6*	0/10

Table 2: Average runtime (aRT in number of function evaluations) divided by the respective best aRT measured during BBOB-2016 in dimension 20. The aRT and in braces, as dispersion measure, the half difference between 10 and 90%-tile of bootstrapped run lengths appear for each algorithm and target, the corresponding reference aRT in the first row. The different target $\Delta I_{\mathrm{HV}}^{\mathrm{COCO}}$-values are shown in the top row. #succ is the number of trials that reached the (final) target $I_{\mathrm{ref}} + 10^{-5}$. The median number of conducted function evaluations is additionally given in *italics*, if the target in the last column was never reached. Entries, succeeded by a star, are statistically significantly better (according to the rank-sum test) when compared to all other algorithms of the table, with $p = 0.05$ or $p = 10^{-k}$ when the number k following the star is larger than 1, with Bonferroni correction of 110. A ↓ indicates the same tested against the best algorithm from BBOB 2016. Best results are printed in bold.

Data produced with COCO v2.1, hv-hash=ff0e71e8cd978373