JADE, an Adaptive Differential Evolution Algorithm, Benchmarked on the BBOB Noiseless Testbed

Petr Pošík
Czech Technical University in Prague
FEE, Dept. of Cybernetics
Technická 2, 166 27 Prague 6, Czech Republic
petr.posik@fel.cvut.cz

Václav Klemš
Czech Technical University in Prague
FEE, Dept. of Cybernetics
Technická 2, 166 27 Prague 6, Czech Republic

ABSTRACT

JADE, an adaptive version of the differential evolution (DE) algorithm, is benchmarked on the testbed of 24 noiseless functions chosen for the Black-Box Optimization Benchmarking workshop. The results of full-featured JADE are then compared with the results of 3 other DE variants ("downgraded" JADE variants) to reveal the contributions of the algorithm components. Another adaptive DE variant benchmarked during BBOB 2010 is used as a reference algorithm. The results confirm that the original JADE outperforms the other (JA)DE versions, while the comparison with the other adaptive DE shows that the different sources of adaptivity make the algorithms suitable for different functions.

Categories and Subject Descriptors

G.1.6 [**Numerical Analysis**]: Optimization—*global optimization, unconstrained optimization*; F.2.1 [**Analysis of Algorithms and Problem Complexity**]: Numerical Algorithms and Problems

General Terms

Algorithms

Keywords

Benchmarking, Black-box optimization, Differential evolution, Adaptation

1. INTRODUCTION

Differential Evolution (DE) [9] is a population-based optimization algorithm popular thanks to its simple structure and wide applicability. Similarly to other optimizers, it has a few parameters which must be properly chosen for the particular task being solved. This fact led to the birth of adaptive versions of DE [8, 1, 2] differing in (1) what they adapt and (2) how. For this article, we chose the JADE algorithm which was shown [10] to be more efficient than the approaches in [8, 1] on a set of several benchmark functions.

GECCO'12 Companion, July 7–11, 2012, Philadelphia, PA, USA.

The purpose of this paper is to evaluate the performance of the JADE algorithm using the COCO framework [5] and to assess the benefits of its individual parts. We also compare the JADE algorithm against the DE-F-AUC [2], another adaptive DE benchmarked in the COCO framework recently.

In Sec. 2 we briefly reiterate the DE algorithm and describe the JADE algorithm in more detail. In Sec. 3, we present the experiment design together with the algorithm parameters settings. Sec. 4 then presents the results and Sec. 5 discusses them.

2. ALGORITHM PRESENTATION

Differential Evolution (DE) [9] is a population-based optimization algorithm. Each generation, for each population member $\mathbf{x}_i$ (the parent), a *donor* $\mathbf{v}_i$ is created using a mutation operator. The donor $\mathbf{v}_i$ is then crossed over with its parent $\mathbf{x}_i$ to create the offspring $\mathbf{u}_i$. The offspring $\mathbf{u}_i$ then replaces its parent $\mathbf{x}_i$ if it is better.

DE mutation operators create the donor individual $\mathbf{v}_i$ as a linear combination of several individuals in the current population. Eq. 1 describes one of the possible mutation operators, the so called "best" mutation operator:

$$\mathbf{v}_i = \mathbf{x}_{\text{best}} + F \cdot (\mathbf{x}_{r1} - \mathbf{x}_{r2}), \tag{1}$$

where F is the mutation factor (a positive number typically chosen from $[0.5, 1]$) and $\mathbf{x}_{r1}$ and $\mathbf{x}_{r2}$ are randomly chosen population members.

The crossover creates the offspring $\mathbf{u}_i$ by taking some solution components from the parent $\mathbf{x}_i$ and other components from the donor $\mathbf{v}_i$. Eq. (2) describes the binomial crossover. It creates the offspring $\mathbf{u}_i = (u_{i,1}, \ldots, u_{i,D})$ as follows:

$$u_{i,j} = \begin{cases} v_{i,j} & \text{if } r_j \le CR_i \text{ or } j = j_{i,\text{rand}}, \\ x_{i,j} & \text{otherwise,} \end{cases} \tag{2}$$

where r_j is a random number uniformly distributed in $[0, 1]$, $CR_i \in [0, 1]$ is the crossover probability representing the average proportion of components the offspring gets from its donor, and $j_{i,\text{rand}}$ is the randomly chosen index of the solution component surely taken from the donor.

JADE [10] is an adaptive version of DE. It was shown to have better performance than other adaptive DE versions (jDE, SaDE) on many benchmark functions. It uses a simple form of adaptation, see Alg. 1. The $\leftarrow$ symbol represents the assignment, while the $\rightarrow$ symbol means addition of a new member to a set . The functions `rn` and `rc` are Gaussian and Cauchy random number generators, respectively, while `meanA` and `meanL` designate the arithmetic and Lehmer (contraharmonic) mean, respectively.

Algorithm 1: JADE

Set $\mu_{CR} \leftarrow 0.5$, $\mu_F \leftarrow 0.5$, archive $A \leftarrow \emptyset$.
Initialize the population $\{\mathbf{x}_i\}_{i=1}^{NP}$.
for $g \leftarrow 1$ *to* G **do**
    $S_F \leftarrow \emptyset$; $S_{CR} \leftarrow \emptyset$;
    for $i \leftarrow 1$ *to* NP **do**
        $F_i \leftarrow$ `rc(`μ_F, *0.1*`)`, $CR_i \leftarrow$ `rn(`μ_{CR}, *0.1*`)`.
        $\mathbf{v}_i \leftarrow$ `mutate(`$\mathbf{x}_i$`)` (Eq. 3)
        $\mathbf{u}_i \leftarrow$ `crossover(`$\mathbf{x}_i$, $\mathbf{v}_i$`)` (Eq. 2)
        if $f(\mathbf{u}_i) < f(\mathbf{x}_i)$ **then**
            $\mathbf{x}_i \rightarrow A$; $CR_i \rightarrow S_{CR}$; $F_i \rightarrow S_F$.
            $\mathbf{x}_i \leftarrow \mathbf{u}_i$
    Randomly remove members of A while $|A| > NP$.
    $\mu_{CR} \leftarrow (1-c) \cdot \mu_{CR} + c \cdot$ `meanA(`S_{CR}`)`
    $\mu_F \leftarrow (1-c) \cdot \mu_F + c \cdot$ `meanL(`S_F`)`

JADE differs from DE in 3 aspects. First, JADE can optionally use an *archive* of parent solutions recently replaced with more successful offspring. The archive is used in the JADE mutation operator.

The second difference from DE is a special mutation operator called "current-to-*p*best":

$$\mathbf{v}_i = \mathbf{x}_i + F_i \cdot (\mathbf{x}_{\text{best}}^p - \mathbf{x}_i) + F_i \cdot (\mathbf{x}_{r1} - \mathbf{x}_{r2}), \qquad (3)$$

where $\mathbf{x}_i$ is the parent individual, $\mathbf{x}_{\text{best}}^p$ is an individual randomly chosen from the best $100p$ % individuals in the current population, $p \in (0, 1]$, $\mathbf{x}_{r1}$ and $\mathbf{x}_{r2}$ are individuals randomly chosen from the population and from the union of the current population and the archive, respectively. The F_i is the mutation factor. The individuals $\mathbf{x}_{\text{best}}^p$, $\mathbf{x}_{r1}$ and $\mathbf{x}_{r2}$, and the value of F_i are chosen anew for each mutation.

The third and most important difference is the adaptation of F and CR. In classic DE, both factors are usually constant (or sampled from a static distribution). In JADE, the crossover probability CR_i is sampled from a normal distribution with mean μ_{CR} and standard deviation of 0.1. Similarly, F_i is sampled from a Cauchy distribution with the location parameter μ_F and scale parameter 0.1. The parameters μ_{CR} and μ_F are updated each generation using the arithmetic and contraharmonic mean, respectively, of the CR_i and F_i values used to create the successful offspring individuals (successful = better than the respective parent).

DE-F-AUC is a DE algorithm able to choose among several (4 in this case) available mutation strategies based on their previous success using a technique called *F-AUC-Bandit* [3]. The results from BBOB 2010 article [2] are used. The algorithm does not contain the crossover operator and relies only on the rotationally invariant mutations.

3. EXPERIMENT DESIGN

The goal of the experiment is to assess the benefits of (1) using the "current-to-*p*best" mutation strategy (refered to also as "ctpb") as opposed to the "best" strategy, and (2) using the JADE parameter adaptation. We thus designed 4 algorithms:

1. JADEctpb, adaptive with "ctpb" (the original JADE),
2. JADEb, adaptive with "best" (a downgraded JADE),
3. DEctpb, non-adaptive with "ctpb", and
4. DEb, non-adaptive with "best" (a conventional DE).

The evaluations budget was set to $5 \cdot 10^4 D$ for each run. For most of the parameters, default values from the literature were used. For DE: $CR = 0.5$, $F \sim U(0.5, 1)$ (sampled anew each generation). For JADE: initial $\mu_{CR} = 0.5$, initial $\mu_F = 0.5$, $p = 0.1$, $|A| = 0.1NP$. The population size was set to $NP = 5D$ for all 4 algorithms after a small systematic study performed on JADEctpb and DEb using the values $(3, 4, 5, 6, 8, 10, 15, 20) \cdot D$. Values lower than $5D$ gave erratic behavior even on uni-modal functions, values larger than $5D$ wasted evaluations on uni-modal functions and did not bring significant advantages on multi-modal functions. All algorithms were restarted when they stagnate for more than 30 generations and the population diversity measure $\frac{1}{D}\sum_{i=1}^{D} Var(X_i) < 10^{-10}$.

4. RESULTS

Results from experiments according to [5] on the benchmark functions given in [4, 6] are presented in Figures 1, 2 and 3 and in Tables 1 and 2. The **expected running time (ERT)**, used in the figures and table, depends on a given target function value, $f_t = f_{\text{opt}} + \Delta f$, and is computed over all relevant trials as the number of function evaluations executed during each trial while the best function value did not reach f_t, summed over all trials and divided by the number of trials that actually reached f_t [5, 7]. **Statistical significance** is tested with the rank-sum test for a given target Δf_t (10^{-8} as in Figure 1) using, for each trial, either the number of needed function evaluations to reach Δf_t (inverted and multiplied by -1), or, if the target was not reached, the best Δf-value achieved, measured only up to the smallest number of overall function evaluations for any unsuccessful trial under consideration.

4.1 CPU Timing Experiments

The timing experiments were carried out with f_8 on a machine with Intel Core 2 Duo processor, 2.4 Ghz, with 4 GB RAM, on Windows 7 64bit in MATLAB R2009b 64bit. The average time per function evaluation in 2, 3, 5, 10, 20, 40 dimensions was about 52, 35, 21, 12, 8, and 7×10^{-6} s for both DE variants, and about 70, 45, 28, 16, 9, 10×10^{-6} s for both JADE variants.

5. DISCUSSION

Influence of the Mutation Strategy. By comparing the algorithm pairs DEb vs. DEctpb and JADEb vs. JADEctpb, we can make some observation about the influence of the chosen mutation strategy. Generally speaking, the "best" strategy is very exploitative, it allows the algorithm to converge (and loose diversity) faster, while the "current-to-*p*best" strategy preserves more diversity in the population which in turn can prevent the algorithm from restarting more often.

Regarding DEb, on uni-modal functions, there is usually not much difference between these two mutation strategies with the exception of the functions f_6 and f_7 where the increased diversity due to the "ctpb" strategy allowed the DE algorithm to solve these problems faster in dimensions ≥ 10. For the multi-modal functions, the results are mixed: sometimes it is better to restart more often (and the "best" strategy allows for this), while sometimes the better preserved diversity ensures better results than restarts (and then the "ctpb" strategy works better).

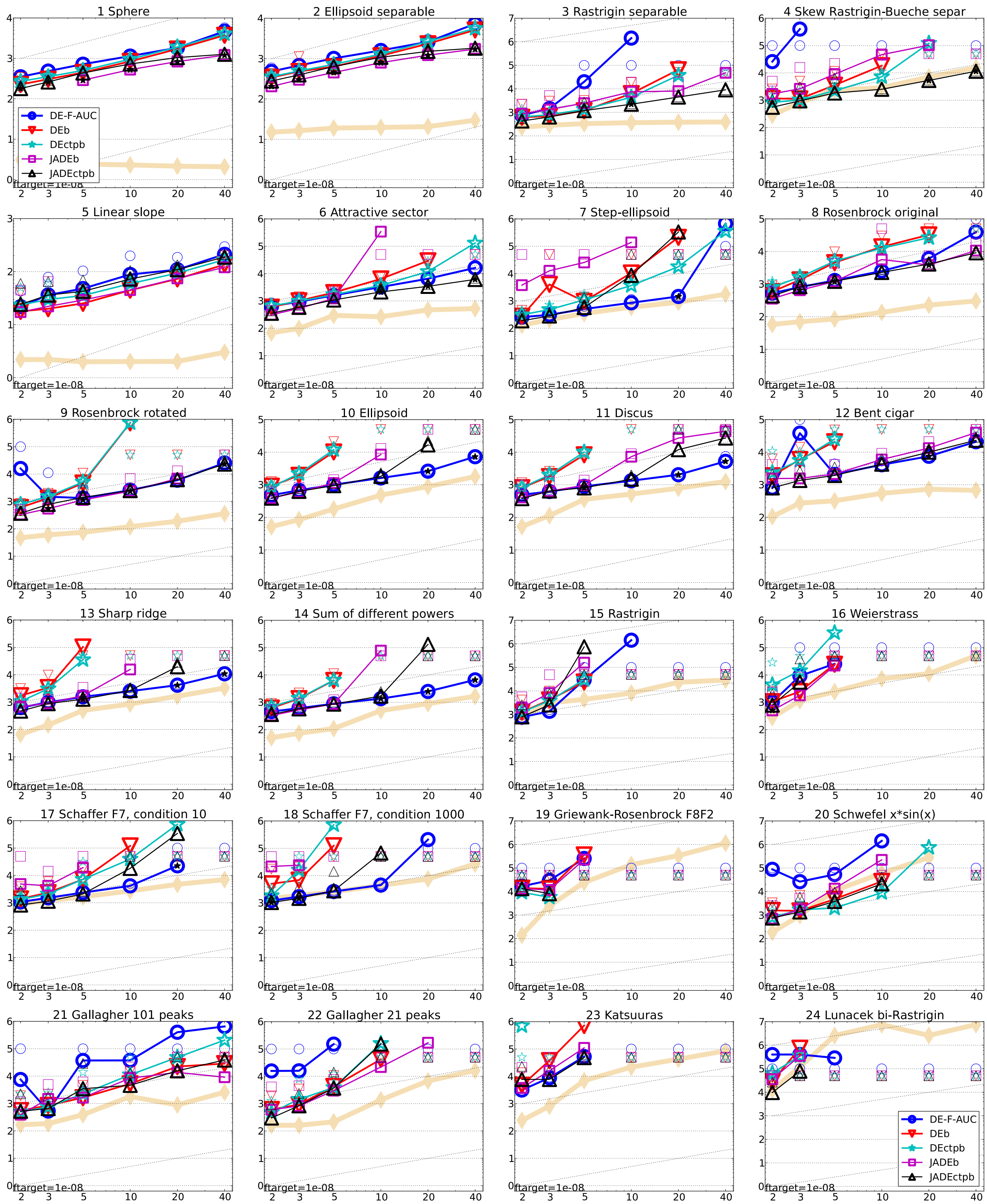

Figure 1: Expected running time (ERT in number of f-evaluations) divided by dimension for target function value 10^{-8} as $\log_{10}$ values versus dimension. Different symbols correspond to different algorithms given in the legend of f_1 and f_{24}. Light symbols give the maximum number of function evaluations from the longest trial divided by dimension. Horizontal lines give linear scaling, slanted dotted lines give quadratic scaling. Black stars indicate statistically better result compared to all other algorithms with $p < 0.01$ and Bonferroni correction number of dimensions (six). Legend: ○: DE-F-AUC, ▽: DEb, ⋆: DEctpb, □: JADEb, △: JADEctpb.

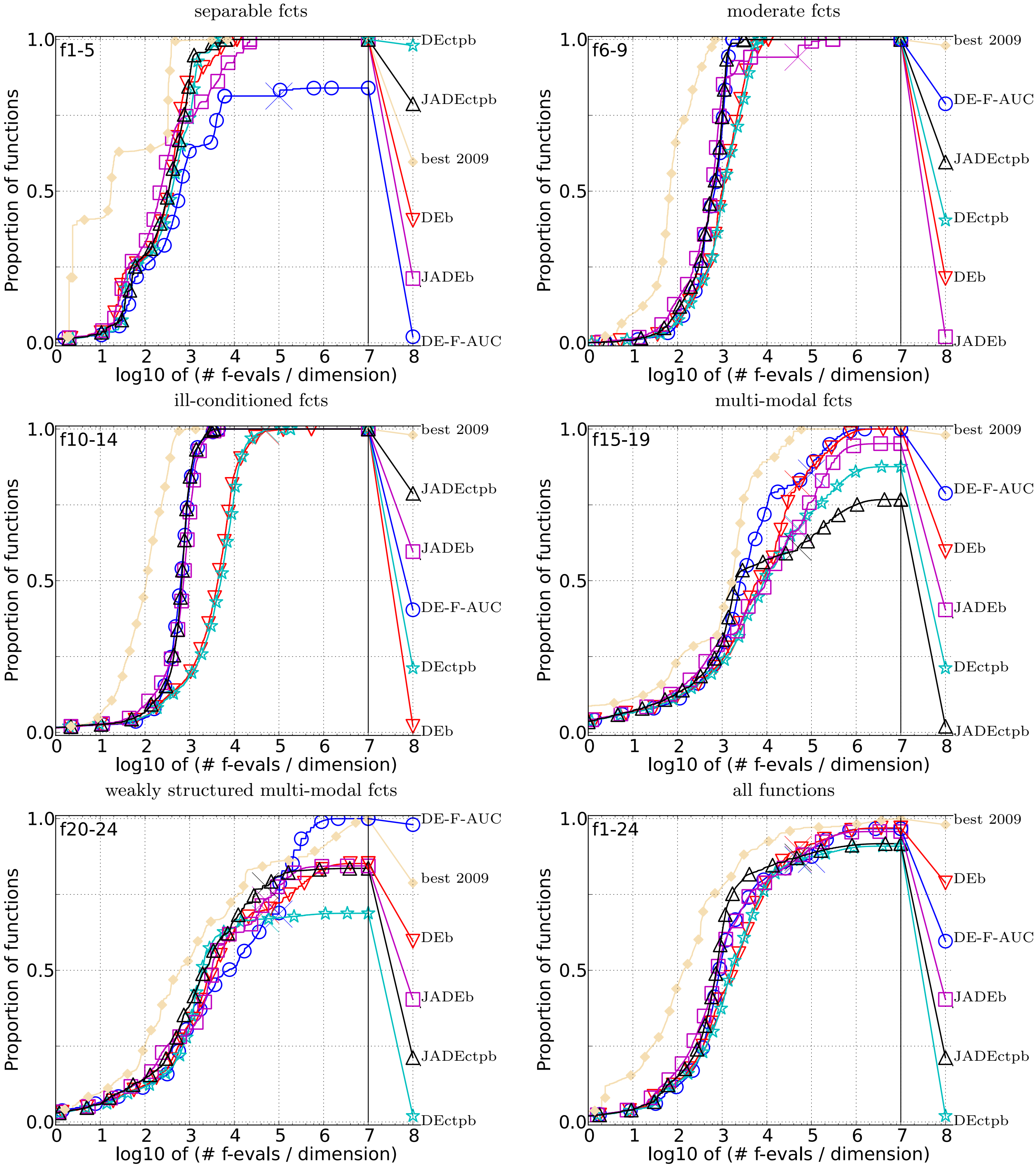

Figure 2: Bootstrapped empirical cumulative distribution of the number of objective function evaluations divided by dimension (FEvals/D) for 50 targets in $10^{[-8..2]}$ for all functions and subgroups in 5-D. The "best 2009" line corresponds to the best ERT observed during BBOB 2009 for each single target.

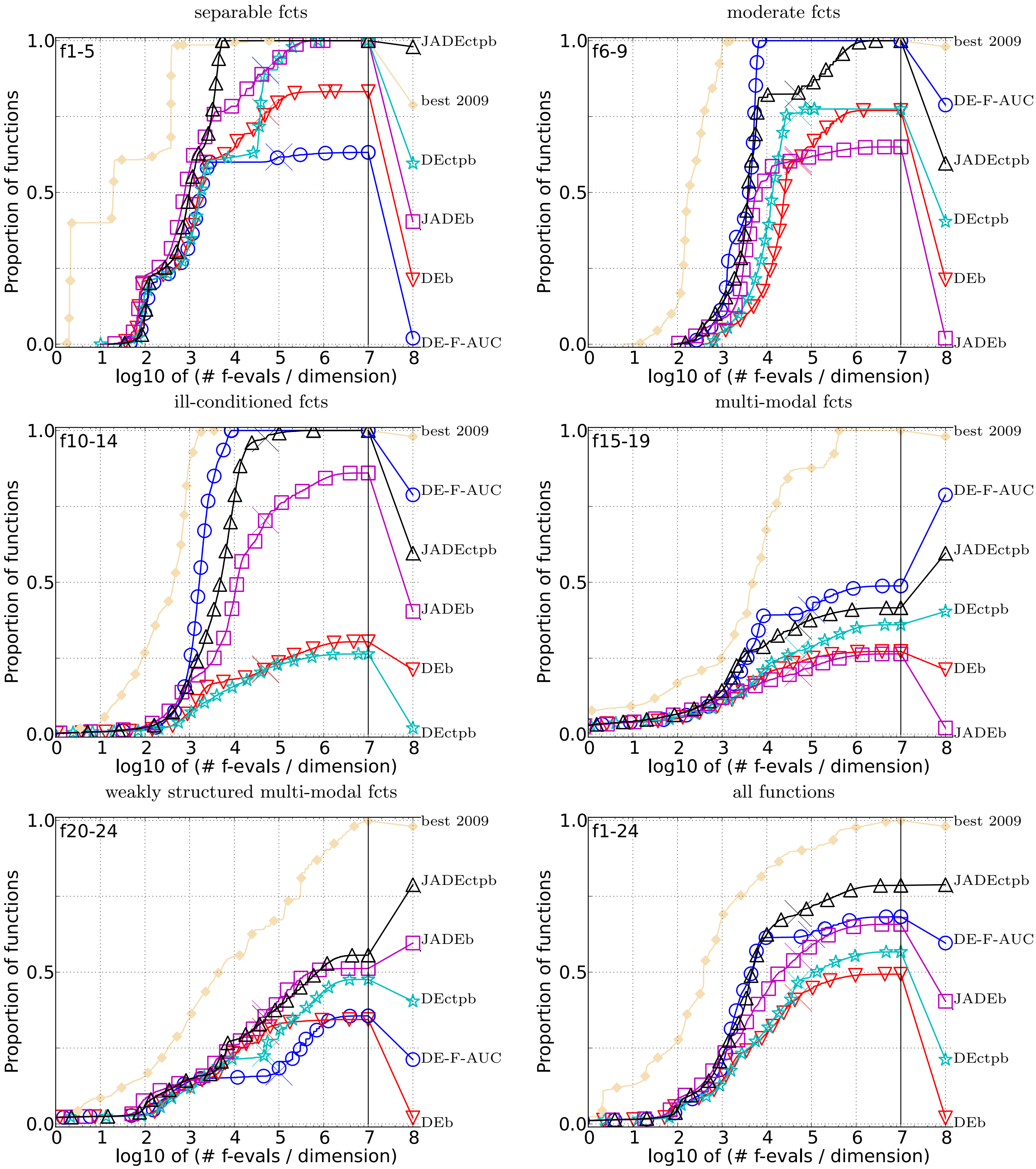

Figure 3: Bootstrapped empirical cumulative distribution of the number of objective function evaluations divided by dimension (FEvals/D) for 50 targets in $10^{[-8..2]}$ for all functions and subgroups in 20-D. The "best 2009" line corresponds to the best ERT observed during BBOB 2009 for each single target.

Δf_{opt}	1e1	1e0	1e-1	1e-3	1e-5	1e-7	#succ
f1	11	12	12	12	12	12	15/15
FAUC	5.9(8)	37(13)	67(13)	132(11)	203(14)	266(14)	15/15
DEb	5.0(4)	21(8)	39(8)	82(8)	122(9)	164(7)	15/15
DEctpb	5.8(5)	26(9)	45(9)	92(10)	139(12)	183(13)	15/15
JDb	**3.4**(3)	**14**(6)	**27**(5)$^{\star 2}$	**50**(6)$^{\star 4}$	**76**(7)$^{\star 4}$	**104**(9)$^{\star 4}$	15/15
JDctpb	4.1(3)	18(6)	36(7)	77(7)	115(10)	154(10)	15/15

Δf_{opt}	1e1	1e0	1e-1	1e-3	1e-5	1e-7	#succ
f2	83	87	88	90	92	94	15/15
FAUC	18(2)	22(2)	26(2)	35(2)	42(2)	50(3)	15/15
DEb	11(1)	13(2)	16(2)	21(2)	27(2)	31(2)	15/15
DEctpb	13(1.0)	16(1)	19(2)	24(2)	30(2)	35(2)	15/15
JDb	**6.8**(1)$^{\star 3}$	**8.5**(2)$^{\star 3}$	**10**(2)$^{\star 4}$	**14**(2)$^{\star 4}$	**18**(2)$^{\star 4}$	**22**(3)$^{\star 4}$	15/15
JDctpb	10(1)	12(2)	15(2)	20(2)	26(2)	31(2)	15/15

Δf_{opt}	1e1	1e0	1e-1	1e-3	1e-5	1e-7	#succ
f3	716	1622	1637	1646	1650	1654	15/15
FAUC	3.4(2)	12(3)	59(153)	60(152)	60(152)	60(152)	13/15
DEb	1.1(0.4)	**1.4**(0.2)	2.5(2)	2.8(2)	3.1(2)	**3.4**(2)	15/15
DEctpb	1.3(0.5)	2.1(0.7)	2.5(0.5)	3.2(0.5)	3.6(0.5)	3.9(0.4)	15/15
JDb	**0.80**(0.2)	1.9(2)	6.6(7)	6.8(7)	7.1(7)	7.3(7)	15/15
JDctpb	1.1(0.5)	1.6(0.3)	**2.2**(0.4)	**2.7**(0.3)	**3.0**(0.3)	3.4(0.3)	15/15

Δf_{opt}	1e1	1e0	1e-1	1e-3	1e-5	1e-7	#succ
f4	809	1633	1688	1817	1886	1903	15/15
FAUC	5.7(2)	625(766)	∞	∞	∞	∞ *5e5*	0/15
DEb	1.2(0.3)	**1.7**(0.3)	9.4(14)	9.0(13)	9.0(13)	9.2(13)	15/15
DEctpb	1.4(0.4)	2.6(0.4)	4.9(3)	5.2(3)	5.4(3)	5.7(3)	15/15
JDb	**0.80**(0.4)$^{\star 2}$	4.7(5)	25(28)	24(26)	23(25)	23(25)	15/15
JDctpb	1.4(0.4)	2.0(0.5)	**3.9**(3)	**4.1**(3)	**4.4**(3)	**4.7**(3)	15/15

Δf_{opt}	1e1	1e0	1e-1	1e-3	1e-5	1e-7	#succ
f5	10	10	10	10	10	10	15/15
FAUC	15(6)	23(9)	24(11)	24(11)	24(11)	24(11)	15/15
DEb	**8.3**(3)	**12**(4)	**12**(3)	**13**(3)	**13**(3)	**13**(3)	15/15
DEctpb	11(5)	16(4)	17(4)	18(2)	18(2)	18(2)	15/15
JDb	8.4(4)	14(4)	15(4)	15(4)	15(4)	15(4)	15/15
JDctpb	11(6)	20(8)	21(7)	21(7)	21(7)	21(7)	15/15

Δf_{opt}	1e1	1e0	1e-1	1e-3	1e-5	1e-7	#succ
f6	114	214	281	580	1038	1332	15/15
FAUC	6.5(2)	7.6(1)	8.5(1)	7.0(0.6)	5.3(0.5)	5.4(0.4)	15/15
DEb	5.4(1)	6.6(2)	8.4(2)	8.0(2)	6.3(1)	6.7(2)	15/15
DEctpb	6.0(3)	6.6(1)	8.2(2)	6.9(1)	5.6(1)	5.8(1.0)	15/15
JDb	**2.6**(1)	**3.4**(1)$^{\star 2}$	**4.5**(1)	4.4(0.9)	3.7(0.9)	4.0(2)	15/15
JDctpb	4.1(2)	4.5(1.0)	5.4(0.9)	**4.4**(0.5)	**3.6**(0.4)	**3.7**(0.5)	15/15

Δf_{opt}	1e1	1e0	1e-1	1e-3	1e-5	1e-7	#succ
f7	24	324	1171	1572	1572	1597	15/15
FAUC	13(7)	2.4(0.6)	**1.1**(0.2)	**1.2**(0.2)	**1.2**(0.2)	**1.3**(0.1)	15/15
DEb	13(11)	3.4(2)	1.9(0.6)	2.5(0.7)	2.5(0.7)	2.7(0.8)	15/15
DEctpb	10(8)	3.3(1)	2.1(0.9)	2.7(0.8)	2.7(0.8)	3.0(0.9)	15/15
JDb	**6.1**(3)	57(0.7)	54(107)	81(159)	81(81)	80(157)	10/15
JDctpb	8.4(5)	**2.0**(0.7)	1.3(0.4)	1.4(0.3)	1.4(0.3)	1.6(0.3)	15/15

Δf_{opt}	1e1	1e0	1e-1	1e-3	1e-5	1e-7	#succ
f8	73	273	336	391	410	422	15/15
FAUC	13(3)	8.9(2)	11(2)	**11**(2)	13(2)	14(2)	15/15
DEb	9.3(4)	14(5)	21(8)	30(8)	40(7)	49(8)	15/15
DEctpb	11(2)	10(4)	22(9)	32(8)	43(7)	54(8)	15/15
JDb	**5.0**(1)*	11(12)	12(10)	12(9)	13(8)	**14**(8)*	15/15
JDctpb	7.5(3)	**7.3**(3)	**10**(3)	12(2)	**13**(1)	14(1)	15/15

Δf_{opt}	1e1	1e0	1e-1	1e-3	1e-5	1e-7	#succ
f9	35	127	214	300	335	369	15/15
FAUC	24(9)	**18**(4)	16(3)	16(2)	16(2)	17(2)	15/15
DEb	22(8)	37(7)	36(10)	40(11)	50(14)	58(17)	15/15
DEctpb	23(11)	29(7)	37(9)	43(11)	53(16)	62(19)	15/15
JDb	**12**(2)*	20(33)	**16**(20)	**15**(13)	**15**(12)*	**15**(11)*	15/15
JDctpb	14(2)	24(7)	22(5)	19(5)	19(4)	18(4)	15/15

Δf_{opt}	1e1	1e0	1e-1	1e-3	1e-5	1e-7	#succ
f10	349	500	574	626	829	880	15/15
FAUC	**4.5**(0.7)$^{\star 2}$	**3.9**(0.4)$^{\star 3}$	**4.0**(0.6)$^{\star 2}$	**4.9**(0.5)	**4.7**(0.4)	5.3(0.4)	15/15
DEb	27(8)	28(6)	32(6)	43(8)	42(6)	49(7)	15/15
DEctpb	32(7)	32(6)	37(8)	52(6)	51(5)	61(6)	15/15
JDb	6.7(2)	5.7(2)	6.0(1)	6.7(2)	5.8(2)	6.1(1)	15/15
JDctpb	6.1(1)	5.1(0.8)	5.0(0.8)	5.4(0.7)	4.8(0.5)	**5.1**(0.5)	15/15

Δf_{opt}	1e1	1e0	1e-1	1e-3	1e-5	1e-7	#succ
f11	143	202	763	1177	1467	1673	15/15
FAUC	**6.2**(1)	**6.6**(0.7)$^{\star 2}$	**2.2**(0.2)$^{\star 2}$	**2.0**(0.1)$^{\star 2}$	**2.2**(0.1)	2.4(0.1)	15/15
DEb	32(23)	42(19)	16(5)	17(4)	19(4)	23(4)	15/15
DEctpb	26(18)	44(17)	18(5)	21(5)	24(6)	27(6)	15/15
JDb	9.0(4)	12(6)	4.0(1)	3.0(0.9)	2.8(0.8)	2.7(0.7)	15/15
JDctpb	8.1(3)	9.0(1)	2.8(0.4)	2.3(0.3)	**2.2**(0.2)	**2.3**(0.2)	15/15

Δf_{opt}	1e1	1e0	1e-1	1e-3	1e-5	1e-7	#succ
f12	108	268	371	461	1303	1494	15/15
FAUC	22(3)	**13**(4)	13(7)	14(9)	6.4(4)	6.6(4)	15/15
DEb	103(94)	107(85)	117(116)	135(129)	63(51)	66(52)	15/15
DEctpb	89(41)	77(46)	104(117)	138(134)	70(57)	80(56)	14/15
JDb	**18**(13)	15(8)	15(10)	15(11)	6.7(4)	6.7(4)	15/15
JDctpb	24(3)	13(4)	**12**(7)	**14**(7)	**6.1**(3)	**6.2**(4)	15/15

Δf_{opt}	1e1	1e0	1e-1	1e-3	1e-5	1e-7	#succ
f13	132	195	250	1310	1752	2255	15/15
FAUC	10(1)	11(0.6)	**11**(1)	3.2(0.2)	3.2(0.2)	3.1(0.2)	15/15
DEb	14(4)	26(11)	39(11)	19(7)	28(9)	45(23)	6/15
DEctpb	17(8)	30(10)	49(16)	23(6)	30(6)	34(4)	12/15
JDb	**5.4**(2)$^{\star 3}$	**8.9**(3)	11(4)	3.5(1)	3.5(1)	3.3(0.8)	15/15
JDctpb	8.2(1)	12(2)	12(2)	**3.1**(0.3)	**2.9**(0.2)	**2.6**(0.2)	15/15

Δf_{opt}	1e1	1e0	1e-1	1e-3	1e-5	1e-7	#succ
f14	10	41	58	139	251	476	15/15
FAUC	2.2(2)	9.4(4)	15(2)	13(2)	11(0.9)	**8.2**(0.5)	15/15
DEb	1.8(3)	7.2(3)	12(2)	15(4)	37(8)	47(12)	15/15
DEctpb	1.1(1)	6.3(4)	12(2)	16(3)	40(12)	52(10)	15/15
JDb	1.3(1)	**4.0**(1.0)	**6.1**(1)$^{\star 3}$	**7.8**(1)*	**10**(2)	8.2(3)	15/15
JDctpb	**0.95**(0.8)	5.3(2)	8.9(1)	10(2)	12(1)	8.2(0.5)	15/15

Δf_{opt}	1e1	1e0	1e-1	1e-3	1e-5	1e-7	#succ
f15	511	9310	19369	20073	20769	21359	14/15
FAUC	5.4(3)	**2.0**(0.7)	7.5(13)	7.3(13)	7.1(12)	6.9(12)	12/15
DEb	6.1(5)	4.3(4)	**6.0**(7)	**5.9**(7)	**5.7**(6)	**5.6**(6)	13/15
DEctpb	6.0(4)	4.9(4)	7.1(8)	7.2(8)	7.1(8)	7.1(7)	12/15
JDb	**3.0**(2)	14(15)	32(36)	39(44)	38(42)	37(41)	4/15
JDctpb	3.7(1)	7.5(14)	39(51)	39(44)	59(65)	175(176)	1/15

Δf_{opt}	1e1	1e0	1e-1	1e-3	1e-5	1e-7	#succ
f16	120	612	2662	10449	11644	12095	15/15
FAUC	6.4(6)	39(27)	31(11)	12(24)	11(22)	10(21)	13/15
DEb	5.1(6)	31(13)	17(16)	11(10)	10(9)	10(8)	14/15
DEctpb	3.9(5)	54(30)	52(30)	165(191)	149(172)	144(175)	2/15
JDb	3.1(3)	**4.5**(2)$^{\star 2}$	**6.8**(8)*	**7.6**(12)	**8.4**(12)	**8.7**(11)	12/15
JDctpb	**2.9**(5)	10(5)	36(48)	∞	∞	∞ *2e5*	0/15

Δf_{opt}	1e1	1e0	1e-1	1e-3	1e-5	1e-7	#succ
f17	5.2	215	899	3669	6351	7934	15/15
FAUC	5.7(6)	3.9(1)	2.3(0.4)	1.3(0.2)	1.3(0.1)	1.4(0.1)	15/15
DEb	4.1(6)	3.2(1.0)	2.2(0.5)	2.1(0.5)	2.8(2)	3.3(3)	15/15
DEctpb	4.1(5)	3.4(1)	2.6(0.9)	2.1(0.5)	2.1(0.5)	3.7(2)	15/15
JDb	**3.1**(2)	**1.4**(0.6)$^{\star 2}$	3.3(7)	3.4(7)	5.3(5)	9.0(7)	13/15
JDctpb	3.3(4)	2.5(0.7)	**1.9**(0.3)	**1.2**(0.2)	**1.2**(0.3)	**1.2**(0.2)	15/15

Δf_{opt}	1e1	1e0	1e-1	1e-3	1e-5	1e-7	#succ
f18	103	378	3968	9280	10905	12469	15/15
FAUC	4.3(1)	3.8(0.5)	0.75(0.1)	0.69(0.1)	**0.88**(0.0)	**0.98**(0.1)	15/15
DEb	2.4(2)	4.4(2)	1.4(0.7)	2.8(3)	7.7(9)	40(45)	5/15
DEctpb	2.9(2)	4.9(2)	1.5(0.7)	2.3(1.0)	6.7(7)	290(311)	1/15
JDb	**1.8**(1)	4.0(1)	2.5(3)	12(13)	72(80)	∞ *2e5*	0/15
JDctpb	1.9(1)	**3.3**(0.7)	**0.72**(0.1)	**0.60**(0.1)$^{\star}_{\downarrow 2}$	1.1(0.2)	1.1(0.2)	15/15

Δf_{opt}	1e1	1e0	1e-1	1e-3	1e-5	1e-7	#succ
f19	1	1	242	1.2e5	1.2e5	1.2e5	15/15
FAUC	**29**(32)	2586(2184)	1726(2151)	11(12)	11(11)	10(11)	5/15
DEb	32(28)	4210(2571)	1106(1094)	15(15)	15(16)	15(16)	2/15
DEctpb	34(41)	3526(3425)	1050(1032)	∞	∞	∞ *2e5*	0/15
JDb	30(24)	3020(2620)	586(682)	**9.5**(10)	**9.4**(10)	**9.4**(10)	3/15
JDctpb	35(26)	**2139**(2314)	**276**(160)	∞	∞	∞ *2e5*	0/15

Δf_{opt}	1e1	1e0	1e-1	1e-3	1e-5	1e-7	#succ
f20	16	851	38111	54470	54861	55313	14/15
FAUC	6.8(6)	7.6(5)	6.9(7)	4.9(9)	4.8(9)	4.8(9)	10/15
DEb	8.8(6)	1.9(0.7)	0.57(0.8)	0.41(0.6)	0.42(0.6)	0.42(0.5)	15/15
DEctpb	7.7(6)	2.1(1)	**0.14**(0.0)$_{\downarrow 4}$	**0.14**(0.0)	**0.16**(0.0)	**0.17**(0.0)	15/15
JDb	5.8(4)	**0.91**(0.3)$^{\star 2}$	1.7(2)	1.2(2)	1.2(2)	1.2(2)	14/15
JDctpb	**4.9**(3)	2.3(2)	0.25(0.2)$_{\downarrow 3}$	0.25(0.2)	0.29(0.2)	0.33(0.2)	15/15

Δf_{opt}	1e1	1e0	1e-1	1e-3	1e-5	1e-7	#succ
f21	41	1157	1674	1705	1729	1757	14/15
FAUC	5.2(8)	1.7(0.9)	110(150)	109(147)	107(145)	106(143)	11/15
DEb	2.7(3)	3.9(6)	3.7(4)	4.0(4)	**4.3**(4)	**4.5**(4)	15/15
DEctpb	3.2(3)	5.5(5)	4.5(4)	5.3(4)	5.8(4)	6.2(4)	15/15
JDb	2.3(2)	4.4(5)	4.4(4)	4.4(4)	4.5(4)	4.6(4)	15/15
JDctpb	**1.7**(2)	**1.1**(1)	**1.2**(1)	**2.8**(4)	5.0(9)	8.2(13)	15/15

Δf_{opt}	1e1	1e0	1e-1	1e-3	1e-5	1e-7	#succ
f22	71	386	938	1008	1040	1068	14/15
FAUC	4.8(4)	203(647)	803(1066)	747(992)	724(961)	706(937)	6/15
DEb	5.0(4)	13(16)	17(25)	17(22)	18(22)	19(21)	15/15
DEctpb	5.1(5)	7.9(13)	11(15)	13(14)	16(13)	17(13)	15/15
JDb	12(30)	16(19)	14(10)	14(10)	**14**(9)	**14**(9)	15/15
JDctpb	**2.1**(1)	**3.4**(3)	**8.0**(11)	**12**(18)	16(27)	17(30)	15/15

Δf_{opt}	1e1	1e0	1e-1	1e-3	1e-5	1e-7	#succ
f23	3.0	518	14249	31654	33030	34256	15/15
FAUC	2.4(2)	**9.3**(5)	**2.3**(0.4)$^{\star 2}$	**3.4**(0.2)	**5.3**(0.4)	**6.9**(0.6)	15/15
DEb	2.9(4)	48(59)	57(61)	35(40)	53(60)	105(120)	1/15
DEctpb	3.5(4)	57(60)	∞	∞	∞	∞ *2e5*	0/15
JDb	**2.3**(3)	35(40)	32(28)	17(16)	17(16)	16(15)	6/15
JDctpb	2.4(3)	37(32)	8.3(6)	7.6(8)	7.3(7)	7.1(7)	10/15

Δf_{opt}	1e1	1e0	1e-1	1e-3	1e-5	1e-7	#succ
f24	1622	2.2e5	6.4e6	9.6e6	1.3e7	1.3e7	3/15
FAUC	**4.1**(3)	**3.7**(5)	**0.13**(0.2)	**0.15**(0.2)	**0.11**(0.1)	**0.11**(0.1)	4/15
DEb	10(8)	5.4(5)	∞	∞	∞	∞ *2e5*	0/15
DEctpb	12(8)	17(19)	∞	∞	∞	∞ *2e5*	0/15
JDb	4.2(4)	7.8(9)	∞	∞	∞	∞ *2e5*	0/15
JDctpb	6.5(5)	∞	∞	∞	∞	∞ *2e5*	0/15

Table 1: Expected running time (ERT in number of function evaluations) divided by the respective best ERT measured during BBOB-2009 (given in the respective first row) for different Δf values in dimension 5. The central 80% range divided by two is given in braces. The median number of conducted function evaluations is additionally given in *italics*, if $\mathrm{ERT}(10^{-7}) = \infty$. #succ is the number of trials that reached the final target $f_{opt} + 10^{-8}$. Best results are printed in bold.

Δf_{opt}	1e1	1e0	1e-1	1e-3	1e-5	1e-7	#succ
f1	43	43	43	43	43	43	15/15
FAUC	93(21)	180(19)	265(26)	431(43)	597(41)	763(48)	15/15
DEb	89(17)	162(31)	241(28)	400(27)	558(30)	717(39)	15/15
DEctpb	91(14)	181(15)	269(21)	440(21)	615(20)	803(34)	15/15
JDb	**35**(3)^{*3}	**67**(4)^{*4}	**102**(5)^{*4}	**179**(13)^{*4}	**260**(14)^{*4}	**346**(17)^{*4}	15/15
JDctpb	47(7)	94(8)	143(8)	240(8)	340(10)	437(13)	15/15
f2	385	386	387	390	391	393	15/15
FAUC	48(4)	58(4)	68(5)	86(7)	105(6)	123(8)	15/15
DEb	41(3)	50(3)	59(3)	76(5)	93(5)	110(5)	15/15
DEctpb	47(2)	56(3)	66(3)	86(4)	105(4)	125(5)	15/15
JDb	**20**(2)^{*4}	**25**(2)^{*4}	**30**(2)^{*4}	**39**(2)^{*4}	**48**(3)^{*4}	**57**(4)^{*4}	15/15
JDctpb	28(1)	34(1)	39(2)	50(2)	61(3)	71(4)	15/15
f3	5066	7626	7635	7643	7646	7651	15/15
FAUC	∞	∞	∞	∞	∞	∞ *2e6*	0/15
DEb	39(10)	67(49)	167(169)	168(169)	168(169)	169(162)	9/15
DEctpb	114(10)	94(8)	95(7)	96(7)	97(7)	98(7)	15/15
JDb	**6.3**(0.3)^{*2}	13(13)	17(20)	18(20)	19(20)	20(20)	15/15
JDctpb	6.4(0.3)	**6.0**(0.2)	**6.8**(0.2)	**8.3**(0.2)	**10**(0.2)	**11**(0.2)	15/15
f4	4722	7628	7666	7700	7758	1.4e5	9/15
FAUC	∞	∞	∞	∞	∞	∞ *2e6*	0/15
DEb	30(9)	∞	∞	∞	∞	∞ *1e6*	0/15
DEctpb	144(19)	119(14)	311(262)	311(261)	310(262)	17(14)	6/15
JDb	17(11)	102(110)	265(266)	265(267)	264(262)	15(15)	6/15
JDctpb	**8.0**(0.4)	**7.0**(0.3)	**8.0**(0.2)^{*2}	**10**(0.2)^{*2}	**11**(0.3)^{*2}	**0.71**(0.0)^{*2}	15/15
f5	41	41	41	41	41	41	15/15
FAUC	42(14)	52(15)	53(16)	54(15)	54(15)	54(15)	15/15
DEb	**27**(6)	35(4)	**36**(6)	**36**(6)	**36**(6)	**36**(6)	15/15
DEctpb	37(7)	43(4)	46(9)	46(10)	46(10)	46(10)	15/15
JDb	28(8)	**35**(7)	37(8)	37(8)	37(8)	37(8)	15/15
JDctpb	43(9)	52(7)	54(8)	54(8)	54(8)	54(8)	15/15
f6	1296	2343	3413	5220	6728	8409	15/15
FAUC	19(2)	15(1)	14(0.7)	14(0.7)	14(0.5)	14(0.4)	15/15
DEb	53(9)	46(7)	46(8)	50(9)	57(11)	61(12)	15/15
DEctpb	29(3)	23(2)	21(2)	21(1)	21(1)	21(1)	15/15
JDb	25(13)	72(53)	362(353)	∞	∞	∞ *1e6*	0/15
JDctpb	**9.4**(0.7)^{*4}	**7.8**(0.8)^{*4}	**7.3**(0.7)^{*4}	**7.2**(0.9)^{*4}	**7.4**(0.9)^{*4}	**7.4**(0.9)^{*4}	15/15
f7	1351	4274	9503	16524	16524	16969	15/15
FAUC	5.2(0.4)	**3.1**(0.4)^{*2}	**1.9**(0.2)^{*2}	**1.6**(0.2)^{*4}	**1.6**(0.2)^{*4}	**1.6**(0.2)^{*4}	15/15
DEb	24(6)	94(121)	235(265)	256(288)	256(272)	249(295)	3/15
DEctpb	29(5)	32(10)	28(5)	20(4)	20(4)	20(6)	14/15
JDb	119(370)	3279(3861)	∞	∞	∞	∞ *1e6*	0/15
JDctpb	**4.8**(0.9)	272(351)	686(842)	402(467)	402(450)	391(467)	2/15
f8	2039	3871	4040	4219	4371	4484	15/15
FAUC	23(5)	21(3)	23(3)	24(3)	25(3)	26(3)	15/15
DEb	53(4)	95(6)	106(9)	120(11)	130(11)	138(12)	14/15
DEctpb	52(3)	66(2)	76(2)	89(4)	100(4)	111(4)	15/15
JDb	**12**(4)^{*4}	**14**(12)*	**14**(11)*	**15**(11)*	**16**(10)*	**17**(10)*	15/15
JDctpb	18(1)	16(0.6)	16(0.6)	17(0.6)	17(0.7)	18(0.7)	15/15
f9	1716	3102	3277	3455	3594	3727	15/15
FAUC	26(4)	**25**(2)	**27**(2)	**29**(2)	**30**(2)	**31**(2)	15/15
DEb	∞	∞	∞	∞	∞	∞ *1e6*	0/15
DEctpb	417(90)	∞	∞	∞	∞	∞ *1e6*	0/15
JDb	**24**(6)	32(24)	33(23)	35(22)	35(21)	36(20)	15/15
JDctpb	36(3)	30(3)	32(3)	33(2)	33(2)	33(2)	15/15
f10	7413	8661	10735	14920	17073	17476	15/15
FAUC	**2.5**(0.2)^{*4}	**2.6**(0.2)^{*4}	**2.4**(0.1)^{*4}	**2.2**(0.1)^{*4}	**2.4**(0.1)^{*4}	**2.8**(0.2)^{*4}	15/15
DEb	∞	∞	∞	∞	∞	∞ *1e6*	0/15
DEctpb	∞	∞	∞	∞	∞	∞ *1e6*	0/15
JDb	39(17)	54(24)	64(16)	116(94)	862(893)	∞ *1e6*	0/15
JDctpb	12(5)	15(4)	15(4)	15(4)	15(3)	18(4)	15/15
f11	1002	2228	6278	9762	12285	14831	15/15
FAUC	**7.3**(1)^{*4}	**5.1**(0.6)^{*4}	**2.4**(0.3)^{*4}	**2.3**(0.2)^{*4}	**2.4**(0.2)^{*4}	**2.5**(0.2)^{*4}	15/15
DEb	7377(7485)	∞	∞	∞	∞	∞ *1e6*	0/15
DEctpb	∞	∞	∞	∞	∞	∞ *1e6*	0/15
JDb	281(506)	140(229)	54(83)	41(52)	37(42)	35(35)	12/15
JDctpb	92(19)	46(8)	18(4)	15(3)	15(3)	15(3)	14/15
f12	1042	1938	2740	4140	12407	13827	15/15
FAUC	27(6)	**20**(12)	**19**(11)	**20**(7)*	**9.4**(2)*	**10**(2)*	15/15
DEb	99(6)	478(538)	1497(1643)	∞	∞	∞ *1e6*	0/15
DEctpb	194(208)	406(392)	1276(1136)	∞	∞	∞ *1e6*	0/15
JDb	23(24)	34(30)	40(29)	39(17)	17(6)	18(5)	15/15
JDctpb	**19**(4)	20(15)	28(15)	28(10)	13(3)	14(3)	15/15
f13	652	2021	2751	18749	24455	30201	15/15
FAUC	25(2)	**12**(0.8)	**11**(0.8)*	**2.4**(0.2)^{*4}	**2.5**(0.2)^{*4}	**2.5**(0.2)^{*4}	15/15
DEb	41(7)	214(306)	702(845)	∞	∞	∞ *1e6*	0/15
DEctpb	50(8)	103(76)	607(587)	∞	∞	∞ *1e6*	0/15
JDb	61(53)	91(112)	175(182)	795(880)	∞	∞ *1e6*	0/15
JDctpb	**17**(2)	14(5)	15(4)	3.6(0.6)	4.8(0.8)	9.0(2)	15/15
f14	75	239	304	932	1648	15661	15/15
FAUC	33(5)	30(3)	38(4)	23(2)	**19**(3)^{*2}	**2.8**(0.3)^{*4}	15/15
DEb	55(19)	46(6)	53(6)	502(176)	∞	∞ *1e6*	0/15
DEctpb	57(23)	47(8)	57(10)	814(261)	∞	∞ *1e6*	0/15
JDb	**14**(4)	**13**(2)^{*3}	**18**(2)^{*4}	21(3)	77(30)	∞ *1e6*	0/15
JDctpb	18(6)	18(1)	23(2)	**20**(1)	38(24)	62(64)	5/15
f15	30378	1.5e5	3.1e5	3.2e5	4.5e5	4.6e5	15/15
FAUC	474(490)	∞	∞	∞	∞	∞ *2e6*	0/15
DEb	∞	∞	∞	∞	∞	∞ *1e6*	0/15
DEctpb	∞	∞	∞	∞	∞	∞ *1e6*	0/15
JDb	∞	∞	∞	∞	∞	∞ *1e6*	0/15
JDctpb	**39**(41)^{*3}	∞	∞	∞	∞	∞ *1e6*	0/15
f16	1384	27265	77015	1.9e5	2.0e5	2.2e5	15/15
FAUC	∞	∞	∞	∞	∞	∞ *2e6*	0/15
DEb	∞	∞	∞	∞	∞	∞ *1e6*	0/15
DEctpb	∞	∞	∞	∞	∞	∞ *1e6*	0/15
JDb	60(42)	∞	∞	∞	∞	∞ *1e6*	0/15
JDctpb	**24**(8)*	∞	∞	∞	∞	∞ *1e6*	0/15
f17	63	1030	4005	30677	56288	80472	15/15
FAUC	23(13)	13(2)	6.6(0.8)	1.9(0.3)	**1.9**(0.2)	**5.5**(12)*	13/15
DEb	26(19)	16(4)	13(4)	8.8(9)	∞	∞ *1e6*	0/15
DEctpb	16(12)	17(3)	11(1)	4.3(0.6)	63(63)	183(205)	1/15
JDb	9.3(6)	14(12)	85(134)	466(554)	∞	∞ *1e6*	0/15
JDctpb	**7.8**(5)	**7.4**(1)	**4.4**(0.7)*	**1.7**(0.5)*	7.2(9)	23(25)	2/15
f18	621	3972	19561	67569	1.3e5	1.5e5	15/15
FAUC	11(1)	4.6(0.7)	1.7(0.2)	**3.2**(0.2)	**11**(15)^{*2}	**28**(34)	5/15
DEb	17(4)	16(5)	15(7)	∞	∞	∞ *1e6*	0/15
DEctpb	17(5)	14(4)	7.8(2)	50(47)	∞	∞ *1e6*	0/15
JDb	**6.3**(3)	13(15)	127(146)	∞	∞	∞ *1e6*	0/15
JDctpb	7.2(2)	**4.4**(1)	**1.5**(0.4)	19(22)	∞	∞ *1e6*	0/15
f19	1	1	3.4e5	6.2e6	6.7e6	6.7e6	15/15
FAUC	1315(356)	9.5e6(1e7)	∞	∞	∞	∞ *2e6*	0/15
DEb	2739(802)	∞	∞	∞	∞	∞ *1e6*	0/15
DEctpb	2148(731)	∞	∞	∞	∞	∞ *1e6*	0/15
JDb	**826**(254)	3.1e6(4e6)	∞	∞	∞	∞ *1e6*	0/15
JDctpb	856(206)	**7.0e5**(6e5)	∞	∞	∞	∞ *1e6*	0/15
f20	82	46150	3.1e6	5.5e6	5.6e6	5.6e6	14/15
FAUC	40(4)	643(693)	∞	∞	∞	∞ *2e6*	0/15
DEb	51(9)	2.4(1)	0.79(0.9)	∞	∞	∞ *1e6*	0/15
DEctpb	56(8)	104(108)	4.8(5)	2.7(3)	**2.7**(3)	**2.7**(3)	1/15
JDb	**19**(3)^{*2}	**0.56**(0.6)*	∞	∞	∞	∞ *1e6*	0/15
JDctpb	24(3)	1.2(0.2)	**0.46**(0.4)	**0.85**(0.9)	2.7(3)	∞ *1e6*	0/15
f21	561	6541	14103	14643	15567	17589	15/15
FAUC	12(2)	613(765)	568(709)	547(615)	515(642)	456(569)	3/15
DEb	33(64)	45(68)	30(38)	29(34)	28(36)	25(31)	13/15
DEctpb	22(11)	139(194)	65(74)	63(102)	61(96)	55(85)	8/15
JDb	20(34)	**24**(36)	**19**(22)	**18**(22)	**17**(20)	**15**(18)	14/15
JDctpb	**7.2**(2)	33(62)	21(35)	20(34)	19(35)	18(31)	13/15
f22	467	5580	23491	24948	26847	1.3e5	12/15
FAUC	675(2142)	1436(1613)	∞	∞	∞	∞ *2e6*	0/15
DEb	**23**(10)	92(107)	∞	∞	∞	∞ *1e6*	0/15
DEctpb	98(14)	282(372)	∞	∞	∞	∞ *1e6*	0/15
JDb	46(66)	**75**(98)	**143**(156)	**135**(143)	**126**(130)	**25**(26)	4/15
JDctpb	25(45)	261(298)	638(681)	601(581)	559(596)	∞ *1e6*	0/15
f23	3.2	1614	67457	4.9e5	8.1e5	8.4e5	15/15
FAUC	1.5(2)	2385(2088)	**440**(474)	∞	∞	∞ *2e6*	0/15
DEb	2.4(2)	∞	∞	∞	∞	∞ *1e6*	0/15
DEctpb	**1.2**(0.6)	∞	∞	∞	∞	∞ *1e6*	0/15
JDb	1.4(0.9)	673(468)	∞	∞	∞	∞ *1e6*	0/15
JDctpb	2.0(2)	**131**(68)^{*4}	∞	∞	∞	∞ *1e6*	0/15
f24	1.3e6	7.5e6	5.2e7	5.2e7	5.2e7	5.2e7	3/15
FAUC	∞	∞	∞	∞	∞	∞ *2e6*	0/15
DEb	∞	∞	∞	∞	∞	∞ *1e6*	0/15
DEctpb	∞	∞	∞	∞	∞	∞ *1e6*	0/15
JDb	∞	∞	∞	∞	∞	∞ *1e6*	0/15
JDctpb	∞	∞	∞	∞	∞	∞ *1e6*	0/15

Table 2: Expected running time (ERT in number of function evaluations) divided by the respective best ERT measured during BBOB-2009 (given in the respective first row) for different Δf values in dimension 20. The central 80% range divided by two is given in braces. The median number of conducted function evaluations is additionally given in *italics*, if $\mathrm{ERT}(10^{-7}) = \infty$. #succ is the number of trials that reached the final target $f_{\mathrm{opt}} + 10^{-8}$. Best results are printed in bold.

Regarding the JADE algorithm, for dimensions ≤ 5, the two strategies work similarly well in terms of the ERT needed to find the $\Delta f = 10^{-8}$. The group of multi-modal functions is an exception where the JADEb algorithm was successful for problems related to larger number of functions and the bootstrapping procedure emphasized this fact. In larger dimensions, the difference is more pronounced and the "ctpb" strategy provides equal or better results in the vast majority of cases.

Influence of the Parameter Adaptation. Comparing the two variants of JADE with the two variants of DE reveals the pros and cons of the parameter adaptation as done in JADE. The JADEb variant works significantly worse than JADEctpb for several functions with $D \geq 10$, while the opposite is only seldom true. The results of JADEctpb compared to both variants of DE are more consistent. Generally speaking, the parameter adaptation as done in JADE is profitable—it reached comparable or better results than both DE variants. The seldom cases where JADEctpb is boldly worse than any of the DEs are f_7 and f_{20} which are probably misleading for the adaptation process and the static parameter settings used by DE is a better choice.

While in low-dimensional spaces, the results for JADEctpb are mixed, the results in 20D space suggest that JADEctpb is able to solve the largest proportion of functions using the smallest number of function evaluations among the two JADE and two DE variants.

Comparison with DE-F-AUC. On uni-modal functions, DE-F-AUC is a competent solver and is generally comparable or better than the JADE algorithm, especially in larger dimensions. The cases where DE-F-AUC is slower than JADE can be attributed to the 2 times larger population of DE-F-AUC, or to the initial adaptation phase.

On multi-modal functions, however, the results are not that clear. The DE-F-AUC algorithm misses the crossover operator which is a serious drawback in case of separable functions (see the results for f_3 and f_4). On non-separable functions, the results are mixed. DE-F-AUC is better for f_{15}, f_{17}, and f_{18} (i.e. the group called "multi-modal" functions), while JADEctpb is better for f_{20}, f_{21}, and f_{22} (i.e. the group of "multi-modal functions with weak structure"). The difference may be partially caused by the missing crossover operator, however, the exact cause remains to be investigated. The results over all functions in 20D suggest that JADEctpb is at least comparable to the DE-F-AUC.

6. SUMMARY AND CONCLUSIONS

We benchmarked the JADE algorithm, an adaptive version of DE, and compared it to a classic DE. JADE uses a different mutation operator and adapts its mutation and crossover parameters F and CR. We assessed the influence of these two features. As another reference algorithm, DE-F-AUC—yet another adaptive DE variant benchmarked during BBOB 2010—was chosen.

The results for low-dimensional spaces ($D \leq 5$) were indecisive, perhaps with the exception of the ill-conditioned functions where the non-adaptive DE variants were 2 to 10 times slower than the rest. In higher-dimensional spaces, the original JADE algorithm (here called JADEctpb) was more successful than its opponents, and comparable to the reference DE-F-AUC algorithm (which looses some "points" due to the absence of the crossover operator and its subsequent inability to solve separable problems efficiently).

The two adaptive DE variants, JADE and DE-F-AUC, use different sources of adaptivity: while JADE adapts only the strategy parameters, DE-F-AUC adapts the use of different strategies. The potential join of these algorithms remains to be investigated as a future work.

Acknowledgements

This work was supported by the Ministry of Education, Youth and Sports of the Czech Republic with the grant No. MSM6840770012 entitled "Transdisciplinary Research in Biomedical Engineering II".

7. REFERENCES

[1] J. Brest, S. Greiner, B. Boskovic, M. Mernik, and V. Zumer. Self-Adapting control parameters in differential evolution: A comparative study on numerical benchmark problems. *Evolutionary Computation, IEEE Transactions on*, 10(6):646–657, Dec. 2006.

[2] A. Fialho, M. Schoenauer, and M. Sebag. Fitness-AUC bandit adaptive strategy selection vs. the probability matching one within differential evolution: an empirical comparison on the bbob-2010 noiseless testbed. In *Proceedings of the 12th annual conference companion on Genetic and evolutionary computation*, GECCO '10, pages 1535–1542, New York, NY, USA, 2010. ACM.

[3] A. Fialho, M. Schoenauer, and M. Sebag. Toward comparison-based adaptive operator selection. In *Proceedings of the 12th annual conference on Genetic and evolutionary computation*, GECCO '10, pages 767–774, New York, NY, USA, 2010. ACM.

[4] 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.

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

[8] A. K. Qin and P. N. Suganthan. Self-adaptive differential evolution algorithm for numerical optimization. In *Evolutionary Computation, 2005. The 2005 IEEE Congress on*, volume 2, pages 1785–1791 Vol. 2. IEEE, 2005.

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

[10] J. Zhang and A. C. Sanderson. JADE: Adaptive differential evolution with optional external archive. *Evolutionary Computation, IEEE Transactions on*, 13(5):945–958, Oct. 2009.