

Modyfikacje/hybrydyzacje algorytmu PSO w zadaniu optymalizacji globalnej wielowymiarowej funkcji ciągłej

Jakub Ruszkowski, Mateusz Kaczmarek

20 maja 2015

1 Wstęp

Hybrydyzacja algorytmu *Particle Swarm Optimization* (PSO) z algorytmem *Differential Evolution* (DE).

2 Opis algorytmu

2.1 Optymalizacja Rojem Cząsteczek

Optymalizacja rojem cząsteczek (ang. Particle Swarm Optimization) jest algorytmem meta heurystycznym służącym do optymalizacji zadanego problemu. Inspiracją dla tego algorytmu była obserwacja zachowań organizmów żywych w populacjach (np. kolonia mrówek, ławica ryb). Cząsteczka (osobnik roju) posiada swoją aktualną pozycję, prędkość oraz najlepszą pozycję, której wartość zostaje zmieniona gdy cząstka znalazła położenie lepiej ocenione. Na początku wartości pozycji oraz prędkości inicjowane są losowymi liczbami. W każdej kolejnej iteracji algorytmu cząsteczki przemieszczają się do nowych położenia symulując adaptację roju do środowiska. Aktualizowane są wówczas najlepsze pozycje cząstek oraz wyznaczany jest lider roju, czyli osobnik o dotychczasowym najlepszym położeniu. Dla każdej cząsteczki obliczany jest także nowy wektor prędkości na podstawie jej bieżącej prędkości oraz położenia lidera roju. Iteracje są powtarzane dopóki nie spełniony zostanie warunek stopu.

Schemat działania algorytmu PSO:

```
dla każdej cząsteczki w roju:
    zainicjuj wartości położenia i prędkości liczbami losowymi;
    while(!stop)
    {
        za pomocą odpowiedniej funkcji dopasowania dokonaj oceny położenia
        cząstek w roju;
        wyznacz lidera roju;
```

dla każdej cząsteczki w roju:
zaktualizuj wektor prędkości oraz położenie;
}

Wektor położenia: $X_i = (x_{i1}, x_{i2}, \dots, x_{iD})$

Wektor prędkości: $V_i = (v_{i1}, v_{i2}, \dots, v_{iD})$

Zmiana położenia w poszczególnych iteracjach:

$$x_{id} = x_{id} + v_{id}$$

Zmiana wartości prędkości w poszczególnych iteracjach:

$$v_{id} = \omega \cdot v_{id} + c_1 \cdot r_1 \cdot (p_{id} - x_{id}) + c_2 \cdot r_2 \cdot (p_{gd} - x_{id}),$$

gdzie ω – stały współczynnik określający stopień kontynuacji ruchu cząstki w dotychczasowym kierunku, c_1, c_2 – współczynniki akceleracji, r_1, r_2 – losowe liczby z przedziału $[0,1]$, p_i, p_g – odpowiednio najlepsza dotychczasowa pozycja cząstki i oraz globalna najlepsza dotychczasowa pozycja wszystkich cząstek.

Algorytm PSO parametryzowany jest zatem trzema wartościami: ω , c_1 oraz c_2 .

2.2 Ewolucja Różnicowa

Algorytm ewolucji różnicowej jest podobnie jak PSO meta heurystycznym algorytmem optymalizacji numerycznej. Algorytm operuje na populacji osobników (odpowiednik cząsteczek w PSO). Każdy osobnik, analogicznie do poprzedniego algorytmu jest reprezentowany przez D-wymiarowy wektor liczb rzeczywistych. W każdym kroku algorytmu, dla każdego osobnika x_i jest tworzony osobnik próbny u_i , który powstaje poprzez zastosowanie operatorów mutacji oraz krzyżowania. Następnie u_i jest porównywany z x_i i jeśli jego dopasowanie jest lepsze, to wtedy zastępuje x_i w populacji. W przeciwnym przypadku osobnik u_i jest odrzucany.

Wynikiem mutacji jest wektor m_i otrzymany w następujący sposób:

$$m_i = x_{r_1} + F \cdot (x_{r_2} - x_{r_3}),$$

gdzie $0 \leq F \leq 1$ jest stałym parametrem, zwanym współczynnikiem amplifikacji, natomiast r_1, r_2, r_3 to trzy losowo wygenerowane numery osobników, przy czym spełniona jest zależność $i \neq r_1 \neq r_2 \neq r_3$. Tak powstały wektor m_i jest nazywany osobnikiem mutantem.

Z kolei wynikiem krzyżowania operującego na rodzicu x_i oraz mutancie m_i jest osobnik próbny u_i , którego każdy element jest wyznaczony w następujący sposób:

$$u_{i,j} = \begin{cases} m_{i,j} & \text{gdy } rnd_j < CR \text{ lub } j=d \\ x_{i,j} & \text{w przeciwnym przypadku} \end{cases}$$

gdzie rnd_j jest liczbą losową z przedziału $[0, 1)$ losowaną niezależnie dla każdego j , natomiast $0 \leq CR \leq 1$ jest stałym parametrem algorytmu, a d jest losowym numerem elementu wektora.

Algorytm DE parametryzowany jest zatem dwiema wartościami: F oraz CR .

2.3 Algorytm hybrydowy

Dane wejściowe:

PSO_DE(*JNIfgeneric* fgeneric, *int* dim, *double* maxfunevals, *Random* rand),
gdzie:

JNIfgeneric fgeneric – klasa z danymi definiującymi problem,

int dim – wymiar problemu,

double maxfunevals – maksymalna liczba iteracji,

Random rand – generator liczb losowych

Oznaczenia używane w algorytmie:

N - liczebność populacji

D - wymiar wektora osobnika

rand() - liczba losowa z $[0, 1]$

x_i – wektor o wymiarze D definiujący położenie osobnika i

p_i – najlepszy dotychczasowy wektor położenia osobnika i (p_g - najlepszy globalny)

v_i – wektor prędkości osobnika i

Schemat działania algorytmu hybrydowego łączącego DE oraz PSO:

```

    przypisz losowo początkowe wartości:  $x_i$ ,  $v_i$  oraz  $p_i$  i  $p_g$  dla  $i = 1, 2, \dots, N$ 
while(!stop)
{
    for  $i = 1$  to  $N$ 
    {
        użyj DE do wyznaczenia nowego kandydata -  $u$ 
        {
            wybierz losowo  $r_1, r_2, r_3$ , takie że  $i \neq r_1 \neq r_2 \neq r_3$ 
             $m_i = x_{r_1} + F \cdot (x_{r_2} - x_{r_3})$ 
            for  $j = 1$  to  $D$ 
            {
                wybierz losowo  $j_{rand}$  z przedziału  $[1, D]$ 
                if (rand() < CR or  $j == j_{rand}$ )
                     $u[j] = m_i[j]$ 
                else
                     $u[j] = x_i[j]$ 
            }
        }
        if ( $f(u) < f(x_i)$ )
             $x_i = u$ 
        else
        {
            użyj PSO do wyznaczenia nowego kandydata -  $TX_i$ 
            {
                oblicz wektor prędkości cząsteczki  $x_i$ :
                 $v_i = \omega \cdot v_i + c_1 \cdot r_1 \cdot (p_i - x_i) + c_2 \cdot r_2 \cdot (p_g - x_i)$ 
            }
        }
    }
}

```

```

    
$$TX_i = x_i + v_i$$

  }
  if ( $f(TX_i) < f(x_i)$ )
    
$$x_i = TX_i$$

  }
  if ( $f(x_i) < f(p_i)$ )
    
$$p_i = x_i$$

  if ( $f(x_i) < f(p_g)$ )
    
$$p_g = x_i$$

}
}

```

2.4 Modyfikacje algorytmu hybrydowego

Stworzyliśmy dodatkowo dwie modyfikacje algorytmu hybrydowego:

2.4.1 Algorytm hybrydowy z restartem

Modyfikacja w stosunku do algorytmu hybrydowego polega na zmianie całej populacji na wartości losowe. Reset populacji następuje w przypadku braku poprawy wyniku przez dotychczasowe osobniki w ciągu pewnej liczby iteracji algorytmu.

2.4.2 Algorytm hybrydowy z dodatkową aktualizacją prędkości

Modyfikacja w tym przypadku polega na dodaniu dodatkowej aktualizacji prędkości w kroku Ewolucji Różnicowej algorytmu. Jeśli wartość osobnika próbnego u jest lepsza niż osobnika oryginalnego, z którego powstał, wyliczana jest wartość prędkości osobnika na podstawie wzoru $v_i = u - x_i$.

3 Procedura eksperymentu

Eksperymenty algorytmu zostały przeprowadzane na zestawie funkcji benchmarkowych "COMparing Continuous Optimisers: COCO" o wymiarach 2, 3, 5, 10, 20. Maksymalna ilość ewaluacji dla każdej z nich wynosiła $D \cdot 10^5$, gdzie D - wymiar funkcji.

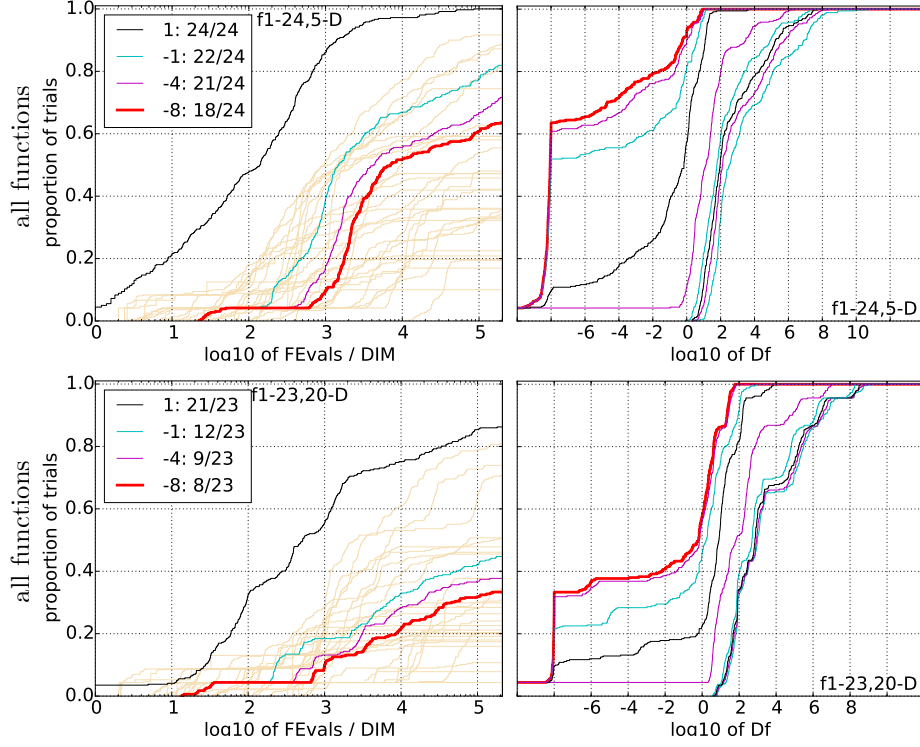
4 CPU Timing

Algorytm był uruchamiany na komputerze z systemem Windows 8 Intel(R) Core(TM) i7-4500U CPU @ 2.39GHz. Czasy ewaluacji funkcji o wymiarach 2, 3, 5, 10, 20 wynosiły odpowiednio $1,9e^{-10}$, $2,2e^{-10}$, $2,4e^{-10}$, $3,5e^{-10}$ and $6,1e^{-10}$ sekund.

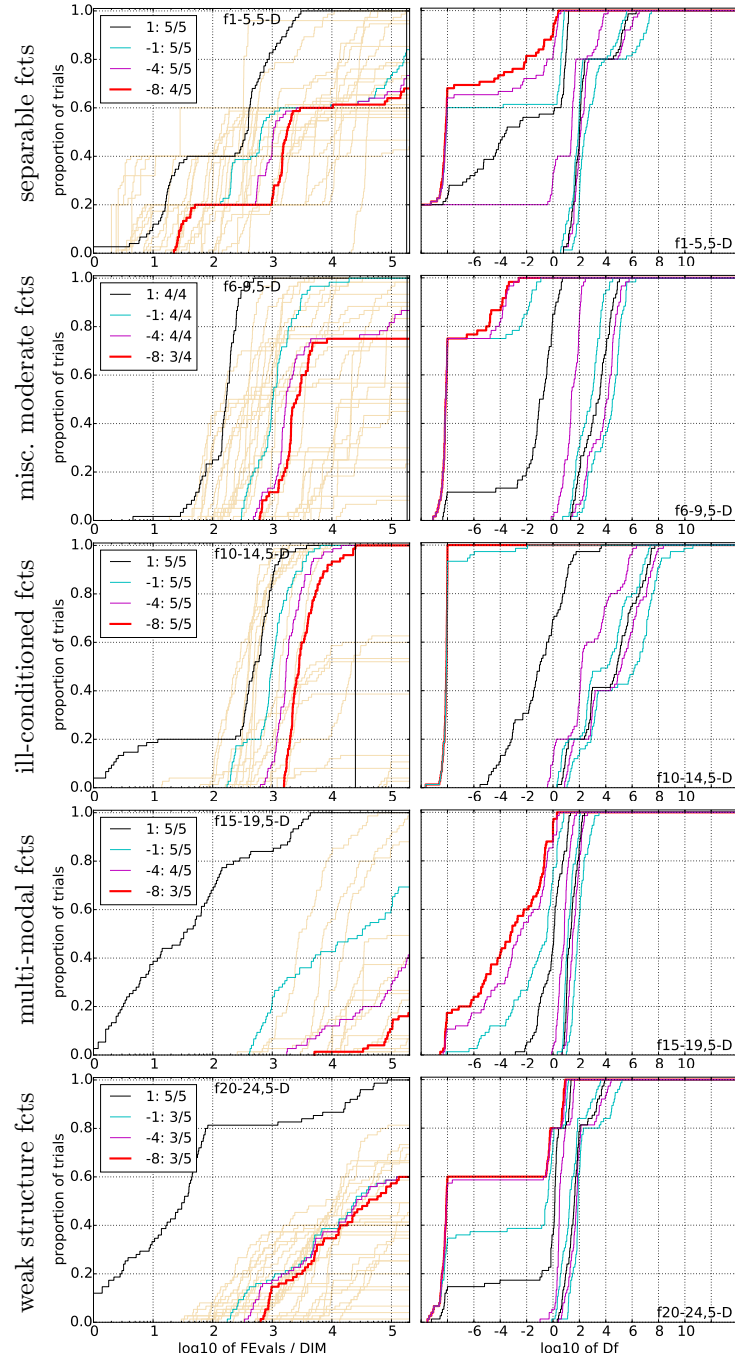
5 Uzyskane wyniki

(DODACĆ TUTAJ)

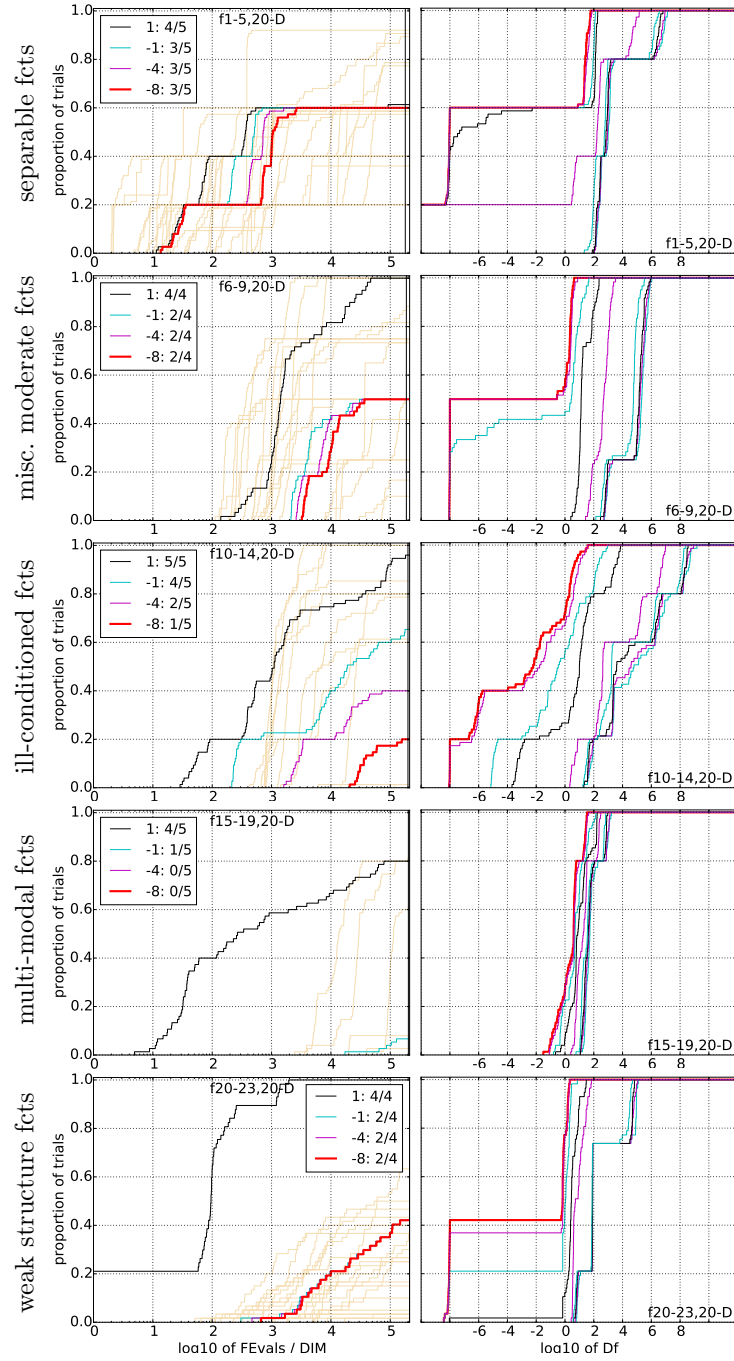
Results from experiments according to [?] on the benchmark functions given in [?, ?] are presented in Figures 1, 2, 3, 4, 5, and 6 and Tables 1 and 2. The **expected running time** (ERT), used in the figures and table, depends on a given target function value, $f_t = f_{\text{opt}} + \Delta f$, and is 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 [?, ?]. **Statistical significance** is tested with the rank-sum test for a given target Δf_t 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 if available.



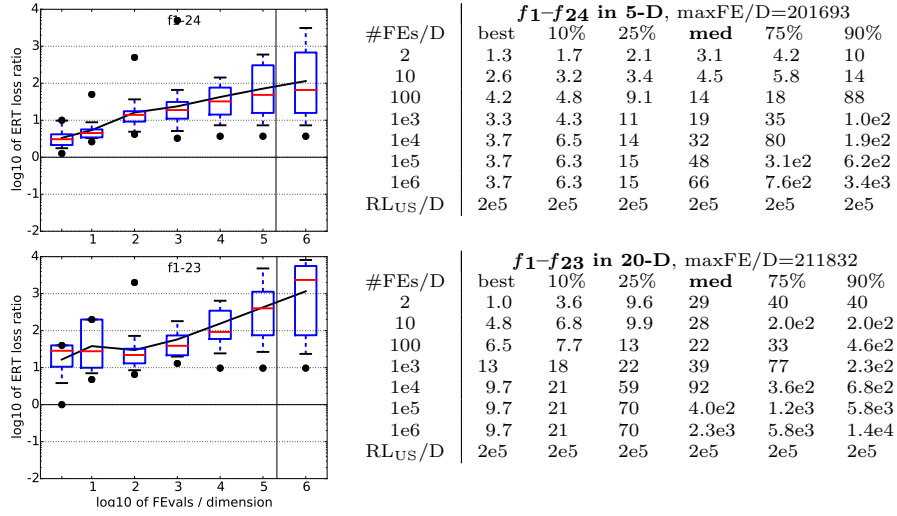
Rysunek 1: Empirical cumulative distribution functions (ECDF), plotting the fraction of trials with an outcome not larger than the respective value on the x -axis. Left subplots: ECDF of the number of function evaluations (FEvals) divided by search space dimension D , to fall below $f_{\text{opt}} + \Delta f$ with $\Delta f = 10^k$, where k is the first value in the legend. The thick red line represents the most difficult target value $f_{\text{opt}} + 10^{-8}$. Legends indicate for each target the number of functions that were solved in at least one trial within the displayed budget. Right subplots: ECDF of the best achieved Δf for running times of $0.5D, 1.2D, 3D, 10D, 100D, 1000D, \dots$ function evaluations (from right to left cycling cyan-magenta-black...) and final Δf -value (red), where Δf and Df denote the difference to the optimal function value. Light brown lines in the background show ECDFs for the most difficult target of all algorithms benchmarked during BBOB-2009. The top row shows results for 5-D and the bottom row for 20-D.



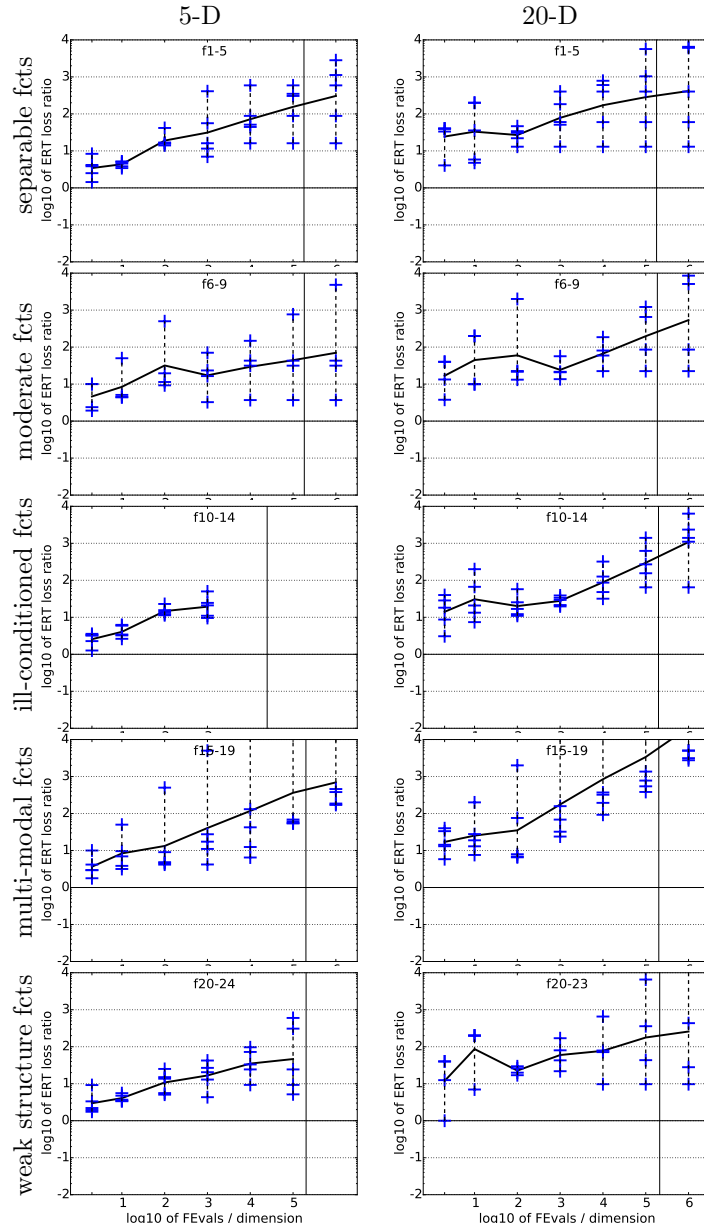
Rysunek 2: Subgroups of functions 5-D. See caption of Figure 1 for details.



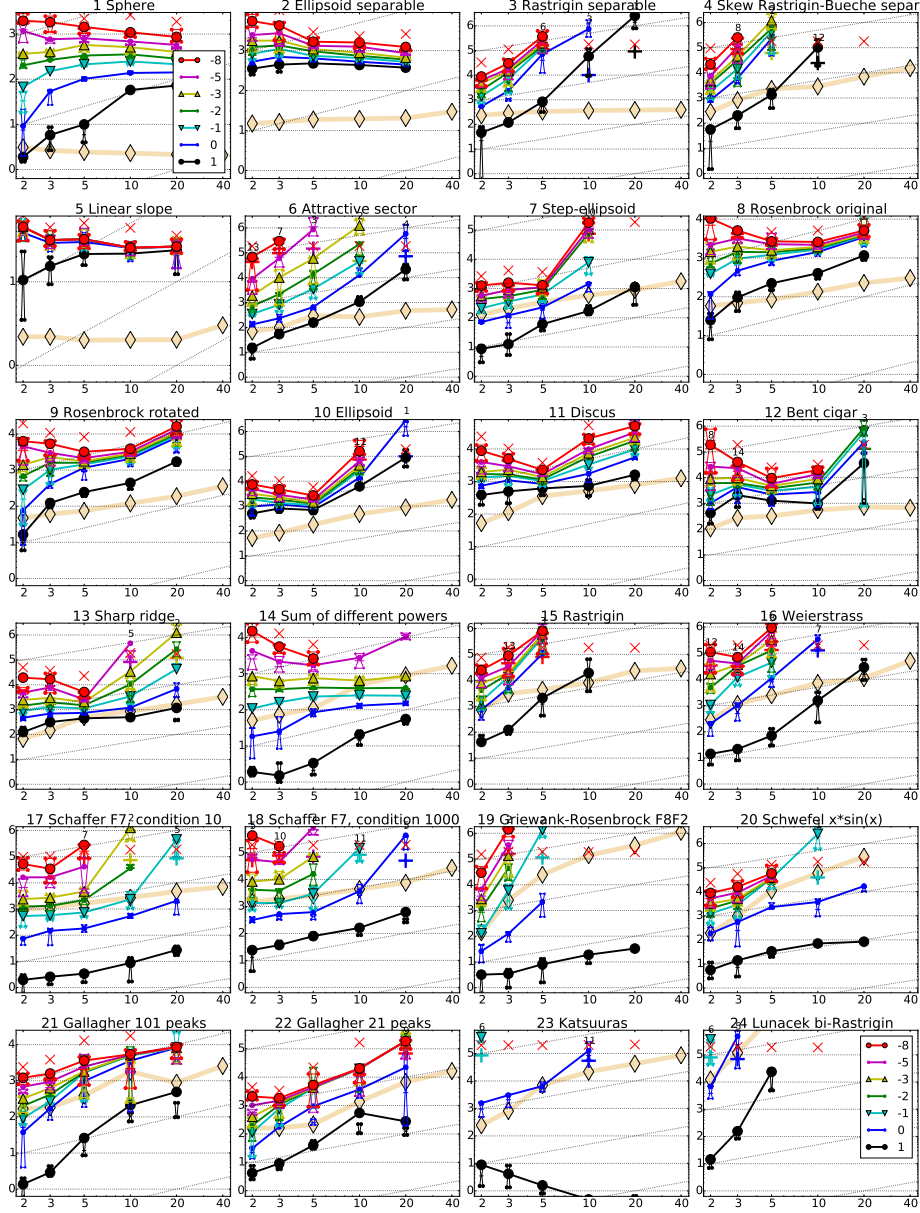
Rysunek 3: Subgroups of functions 20-D. See caption of Figure 1 for details.



Rysunek 4: ERT loss ratio. Left: plotted versus given budget FEvals = #FEs in log-log display. Box-Whisker plot shows 25-75%-ile (box) with median, 10-90%-ile (caps), and minimum and maximum ERT loss ratio (points). The black line is the geometric mean. The vertical line gives the maximal number of function evaluations. Right: tabulated ERT loss ratios in 5-D (top table) and 20-D (bottom table). maxFE/D gives the maximum number of function evaluations divided by the dimension. RL_{US}/D gives the median number of function evaluations for unsuccessful trials.



Rysunek 5: ERT loss ratio versus given budget FEvals divided by dimension in log-log display. Crosses give the single values on the indicated functions, the line is the geometric mean. The vertical line gives the maximal number of function evaluations in the respective function subgroup.



Rysunek 6: Expected number of f -evaluations (ERT, lines) to reach $f_{\text{opt}} + \Delta f$; median number of f -evaluations (+) to reach the most difficult target that was reached not always but at least once; maximum number of f -evaluations in any trial (\times); interquartile range with median (notched boxes) of simulated runlengths to reach $f_{\text{opt}} + \Delta f$; all values are 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 (if appearing) indicate the number of trials reaching the respective target. 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.

Δf	1e+1	1e+0	1e-1	1e-2	1e-3	1e-5	1e-7	#succ
f₁	11 4.5(4)	12 42(8)	12 87(17)	12 137(78)	12 235(59)	12 330(164)	12 538(217)	15/15 15/15
f₂	83 29(4)	87 37(15)	88 42(19)	89 54(43)	90 61(10)	92 72(41)	94 83(11)	15/15 15/15
f₃	716 5.8(2)	1622 219(189)	1637 405(213)	1642 452(376)	1646 604(826)	1650 782(982)	1654 900(864)	15/15 6/15
f₄	809 8.3(6)	1633 712(986)	1688 1297(1780)	1758 1645(1871)	1817 3361(3709)	1886 ∞	1903 $\infty 8.9e5$	15/15 0/15
f₅	10 11(3)	10 15(4)	10 16(4)	10 16(6)	10 16(7)	10 16(5)	10 16(6)	15/15 15/15
f₆	114 6.8(2)	214 15(15)	281 63(119)	404 173(154)	580 507(764)	1038 4163(3929)	1332 $\infty 9.0e5$	15/15 0/15
f₇	24 13(6)	324 3.5(1.0)	1171 2.7(2)	1451 3.1(2)	1572 3.5(2)	1572 3.5(2)	1597 3.9(4)	15/15 15/15
f₈	73 15(6)	273 15(11)	336 18(8)	372 19(7)	391 20(11)	410 28(12)	422 30(16)	15/15 15/15
f₉	35 36(23)	127 46(42)	214 34(16)	263 30(21)	300 29(25)	335 32(12)	369 38(13)	15/15 15/15
f₁₀	349 10(3)	500 9.0(2)	574 9.5(1)	607 10(2)	626 11(2)	829 11(5)	880 14(5)	15/15 15/15
f₁₁	143 22(19)	202 21(10)	763 6.8(3)	977 6.0(2)	1177 5.8(2)	1467 5.7(2)	1673 6.4(2)	15/15 15/15
f₁₂	108 59(9)	268 41(14)	371 42(25)	413 46(18)	461 51(31)	1303 22(14)	1494 31(28)	15/15 15/15
f₁₃	132 17(8)	195 19(6)	250 22(9)	319 23(9)	1310 7.7(3)	1752 7.4(3)	2255 11(2)	15/15 15/15
f₁₄	10 1.7(2)	41 10(5)	58 20(6)	90 23(10)	139 27(4)	251 34(29)	476 25(19)	15/15 15/15
f₁₅	511 20(20)	9310 57(34)	19369 118(197)	19743 150(249)	20073 147(192)	20769 194(287)	21359 189(193)	14/15 3/15
f₁₆	120 2.9(5)	612 90(95)	2662 82(80)	10163 62(156)	10449 154(150)	11644 291(231)	12095 385(332)	15/15 3/15
f₁₇	5.2 3.3(3)	215 4.2(2)	899 4.1(2)	2861 3.8(2)	3669 6.4(3)	6351 33(33)	7934 127(133)	15/15 7/15
f₁₈	103 3.8(3)	378 8.2(4)	3968 3.7(3)	8451 10(5)	9280 37(51)	10905 382(427)	12469 $\infty 9.3e5$	15/15 0/15
f₁₉	1 40(47)	1 10659(5637)	242 27837(49148)	1.0e5 ∞	1.2e5 ∞	1.2e5 ∞	1.2e5 $\infty 9.5e5$	15/15 0/15
f₂₀	16 10(7)	851 13(6)	38111 4.7(3)	51362 3.7(3)	54470 3.5(2)	54861 4.1(4)	55313 4.5(4)	14/15 15/15
f₂₁	41 3.2(3)	1157 4.2(5)	1674 5.0(7)	1692 5.2(5)	1705 5.3(7)	1729 7.1(5)	1757 8.5(12)	14/15 15/15
f₂₂	71 2.8(2)	386 12(27)	938 22(11)	980 22(54)	1008 21(16)	1040 22(31)	1068 24(50)	14/15 15/15
f₂₃	3.0 2.6(2)	518 63(55)	14249 ∞	27890 ∞	31654 ∞	33030 ∞	34256 $\infty 1.0e6$	15/15 0/15
f₂₄	1622 73(83)	2.2e5 ∞	6.4e6 ∞	9.6e6 ∞	9.6e6 ∞	1.3e7 ∞	1.3e7 $\infty 9.5e5$	3/15 0/15

Tabela 1: Expected running time (ERT in number of function evaluations) divided by the best ERT measured during BBOB-2009. The ERT and in braces, as dispersion measure, the half difference between 90 and 10%-tile of bootstrapped run lengths appear in the second row of each cell, the best ERT in the first. The different target Δf -values are shown in the top row. #succ is the number of trials that reached the (final) target $f_{\text{opt}} + 10^{-8}$. The median number of conducted function evaluations is additionally given in *italics*, if the target in the last column was never reached. **Bold** entries are statistically significantly better (according to the rank-sum test) compared to the best algorithm in BBOB-2009, with $p = 0.05$ or $p = 10^{-k}$ when the number $k > 1$ is following the \downarrow symbol, with Bonferroni correction by the number of functions. Results of PSO DE withResets in 5-D.

Δf	1e+1	1e+0	1e-1	1e-2	1e-3	1e-5	1e-7	#succ
f₁	43	43	43	43	43	43	43	15/15
	34 ₍₅₎	66 ₍₇₎	100 ₍₁₂₎	133 ₍₁₅₎	187 ₍₈₎	267 ₍₁₅₄₎	368 ₍₂₀₇₎	15/15
f₂	385	386	387	388	390	391	393	15/15
	20 ₍₃₎	23 ₍₅₎	27 ₍₄₎	31 ₍₃₎	34 ₍₃₎	44 ₍₂₎	54 ₍₂₃₎	15/15
f₃	5066	7626	7635	7637	7643	7646	7651	15/15
	10080 ₍₇₁₂₀₎	∞	∞	∞	∞	∞	∞ <i>3.5e6</i>	0/15
f₄	4722	7628	7666	7686	7700	7758	1.4e5	9/15
	∞	∞	∞	∞	∞	∞	∞ <i>3.6e6</i>	0/15
f₅	41	41	41	41	41	41	41	15/15
	12 ₍₃₎	13 ₍₄₎	13 ₍₄₎	13 ₍₃₎	13 ₍₅₎	13 ₍₄₎	13 ₍₅₎	15/15
f₆	1296	2343	3413	4255	5220	6728	8409	15/15
	334 ₍₂₅₈₎	4934 ₍₈₃₂₈₎	∞	∞	∞	∞	∞ <i>3.7e6</i>	0/15
f₇	1351	4274	9503	16523	16524	16524	16969	15/15
	17 ₍₃₀₎	∞	∞	∞	∞	∞	∞ <i>3.7e6</i>	0/15
f₈	2039	3871	4040	4148	4219	4371	4484	15/15
	11 ₍₃₎	18 ₍₁₃₎	18 ₍₁₂₎	19 ₍₁₆₎	19 ₍₁₈₎	21 ₍₆₎	22 ₍₁₂₎	15/15
f₉	1716	3102	3277	3379	3455	3594	3727	15/15
	20 ₍₉₎	51 ₍₆₇₎	55 ₍₄₂₎	59 ₍₇₇₎	64 ₍₄₉₎	74 ₍₃₈₎	82 ₍₅₈₎	15/15
f₁₀	7413	8661	10735	13641	14920	17073	17476	15/15
	240 ₍₈₅₎	6468 ₍₈₄₆₅₎	∞	∞	∞	∞	∞ <i>3.9e6</i>	0/15
f₁₁	1002	2228	6278	8586	9762	12285	14831	15/15
	32 ₍₂₀₎	50 ₍₃₀₎	31 ₍₂₃₎	42 ₍₂₁₎	46 ₍₃₀₎	53 ₍₃₈₎	57 ₍₉₎	15/15
f₁₂	1042	1938	2740	3156	4140	12407	13827	15/15
	704 ₍₁₇₇₄₎	2442 ₍₆₁₄₈₎	4350 ₍₃₀₅₉₎	5462 ₍₃₀₃₅₎	∞	∞	∞ <i>3.7e6</i>	0/15
f₁₃	652	2021	2751	3507	18749	24455	30201	15/15
	36 ₍₆₇₎	67 ₍₈₂₎	310 ₍₄₈₇₎	1537 ₍₁₄₅₀₎	1315 ₍₁₇₇₈₎	∞	∞ <i>3.4e6</i>	0/15
f₁₄	75	239	304	451	932	1648	15661	15/15
	15 ₍₅₎	13 ₍₃₎	16 ₍₁₎	18 ₍₂₎	18 ₍₃₎	129 ₍₄₆₎	∞ <i>4.0e6</i>	0/15
f₁₅	30378	1.5e5	3.1e5	3.2e5	3.2e5	4.5e5	4.6e5	15/15
	∞	∞	∞	∞	∞	∞	∞ <i>3.6e6</i>	0/15
f₁₆	1384	27265	77015	1.4e5	1.9e5	2.0e5	2.2e5	15/15
	417 ₍₄₈₃₎	∞	∞	∞	∞	∞	∞ <i>4.0e6</i>	0/15
f₁₇	63	1030	4005	12242	30677	56288	80472	15/15
	8.6 ₍₅₎	40 ₍₂₀₎	2268 ₍₁₇₁₄₎	∞	∞	∞	∞ <i>3.7e6</i>	0/15
f₁₈	621	3972	19561	28555	67569	1.3e5	1.5e5	15/15
	20 ₍₂₆₎	2171 ₍₁₆₀₇₎	∞	∞	∞	∞	∞ <i>3.8e6</i>	0/15
f₁₉	1	1	3.4e5	4.7e6	6.2e6	6.7e6	6.7e6	15/15
	661 ₍₂₉₃₎	∞	∞	∞	∞	∞	∞ <i>3.7e6</i>	0/15
f₂₀	82	46150	3.1e6	5.5e6	5.5e6	5.6e6	5.6e6	14/15
	20 ₍₄₎	7.4 ₍₆₎	∞	∞	∞	∞	∞ <i>3.5e6</i>	0/15
f₂₁	561	6541	14103	14318	14643	15567	17589	15/15
	17 ₍₂₇₎	25 ₍₃₉₎	12 ₍₂₅₎	12 ₍₁₆₎	11 ₍₁₈₎	11 ₍₁₂₎	10 ₍₇₎	15/15
f₂₂	467	5580	23491	24163	24948	26847	1.3e5	12/15
	12 ₍₄₎	79 ₍₆₆₎	159 ₍₁₁₃₎	155 ₍₁₃₁₎	150 ₍₂₀₀₎	140 ₍₁₁₀₎	28 ₍₃₀₎	9/15
f₂₃	3.2	1614	67457	3.7e5	4.9e5	8.1e5	8.4e5	15/15
	1.8 _(0.9)	∞	∞	∞	∞	∞	∞ <i>4.2e6</i>	0/12

Tabela 2: Expected running time (ERT in number of function evaluations) divided by the best ERT measured during BBOB-2009. The ERT and in braces, as dispersion measure, the half difference between 90 and 10%-tile of bootstrapped run lengths appear in the second row of each cell, the best ERT in the first. The different target Δf -values are shown in the top row. #succ is the number of trials that reached the (final) target $f_{\text{opt}} + 10^{-8}$. The median number of conducted function evaluations is additionally given in *italics*, if the target in the last column was never reached. **Bold** entries are statistically significantly better (according to the rank-sum test) compared to the best algorithm in BBOB-2009, with $p = 0.05$ or $p = 10^{-k}$ when the number $k > 1$ is following the \downarrow symbol, with Bonferroni correction by the number of functions. Results of PSO DE withResets in 20-D.