Kacper Ptaszek

Classic evolutionary algorithm with tournament selection

I am optimizing functions number 2 and 13 from CEC 2017 in 10 dimensions. The space cube limits are -100, 100.

Below I will include measurement data from experiments for different population sizes and sigma parameter values, where the average was calculated each time based on 100 experiments.

Table for sigma=0.5 and different population sizes
Function f2

population size	min	max	average	standard deviation
2	230,04	21889,94	2326,72	4168,11
4	203,78	471,33	275,33	57,66
6	201,87	596,74	225,14	42,13
8	201,23	6514,3	279,29	627,65
12	200,65	5201,21	295,12	458,35
16	200,89	123532,65	1471,09	12270,68
25	200,34	2957198	296234	2342245

Looking at the average of the obtained function values from 100 experiments, the best value (lowest) was obtained by a population consisting of 6 individuals. While obtaining the lowest standard deviation of 42.13. This proves the stability of the results for this population size, which means that smaller dispersion of the function values indicates greater predictability for this number of individuals.

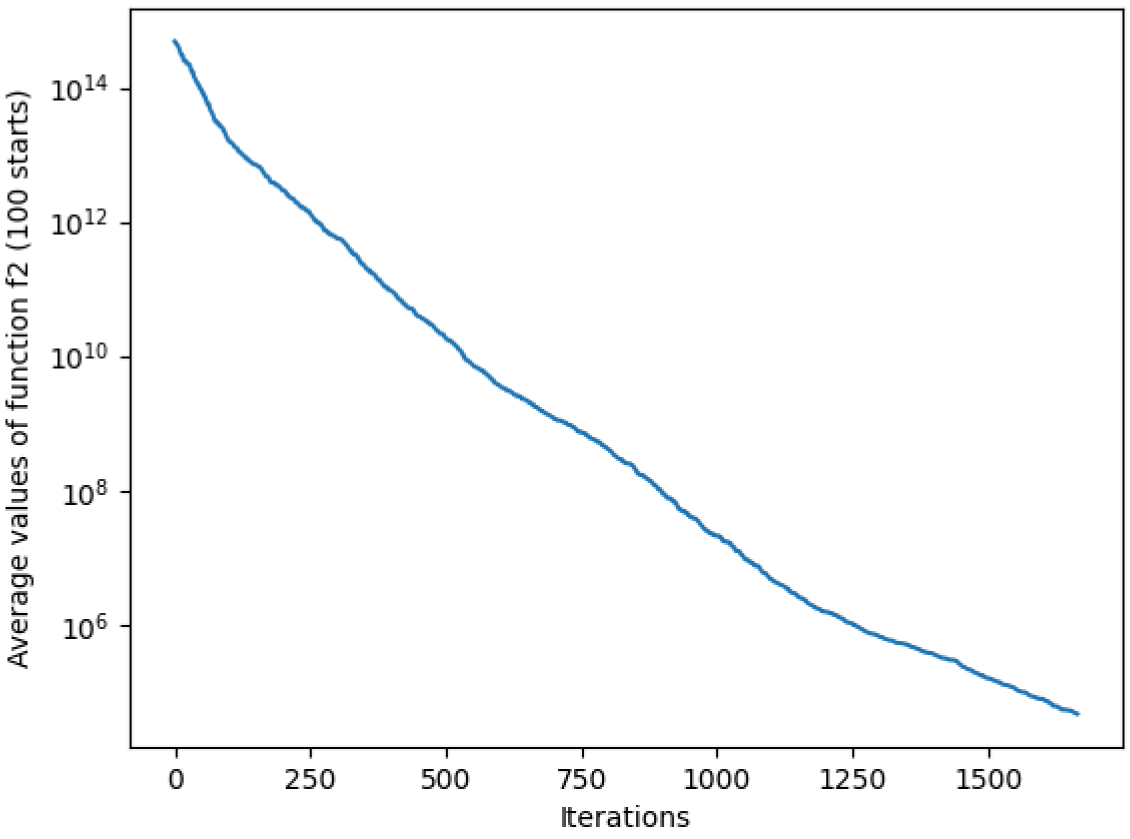
I then examine the effect of changing the mutation strength for the best population found, i.e. 6 individuals.

Table for a fixed number of individuals of 6 and different sigma values
Function f2

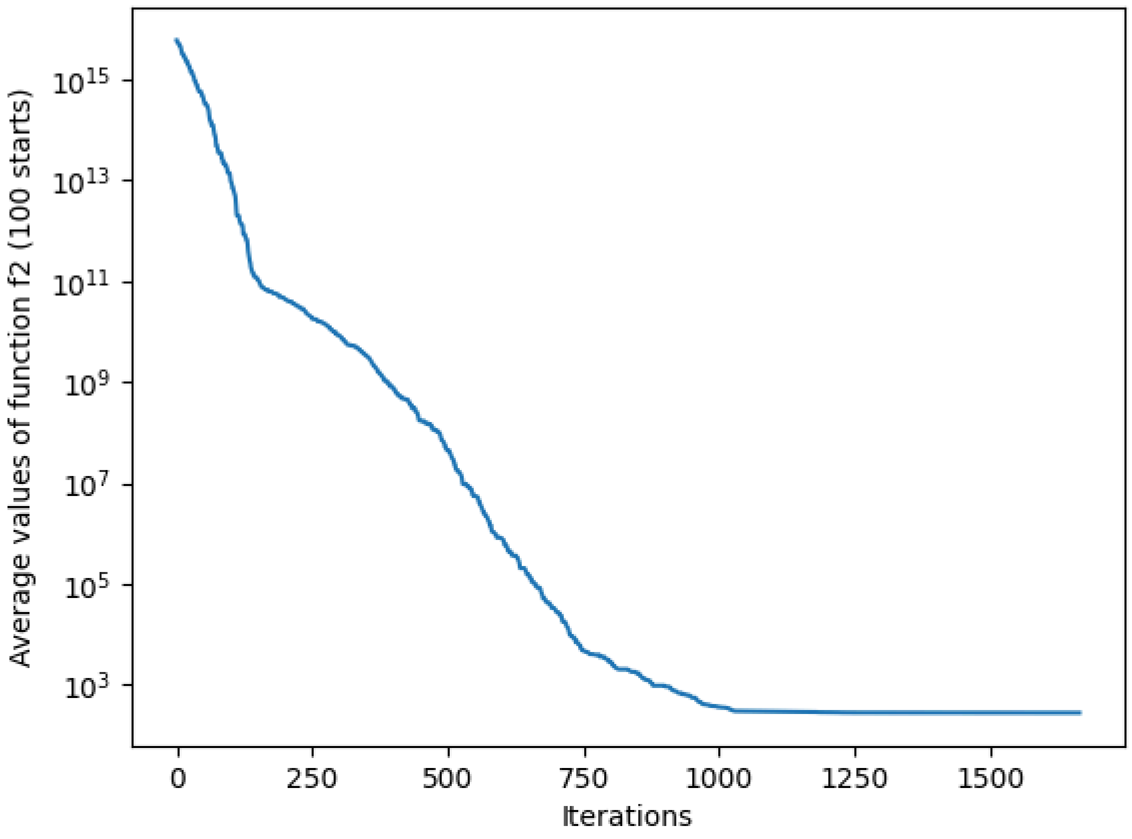
sigma	min	max	average	standard deviation
0,2	201,01	4103164,21	43203,05	40810,6
0,5	201,87	596,74	225,14	42,13
1	251,11	23218,54	5017,21	5914,3
2	958	483026,71	73363,09	9846,21
3	10378	13194762	467322	1353254

For a sigma parameter of 0.5, the population best finds the minimum of f2. Additionally, I include charts showing the best values of an individual in the population from iteration depending on the given sigma.

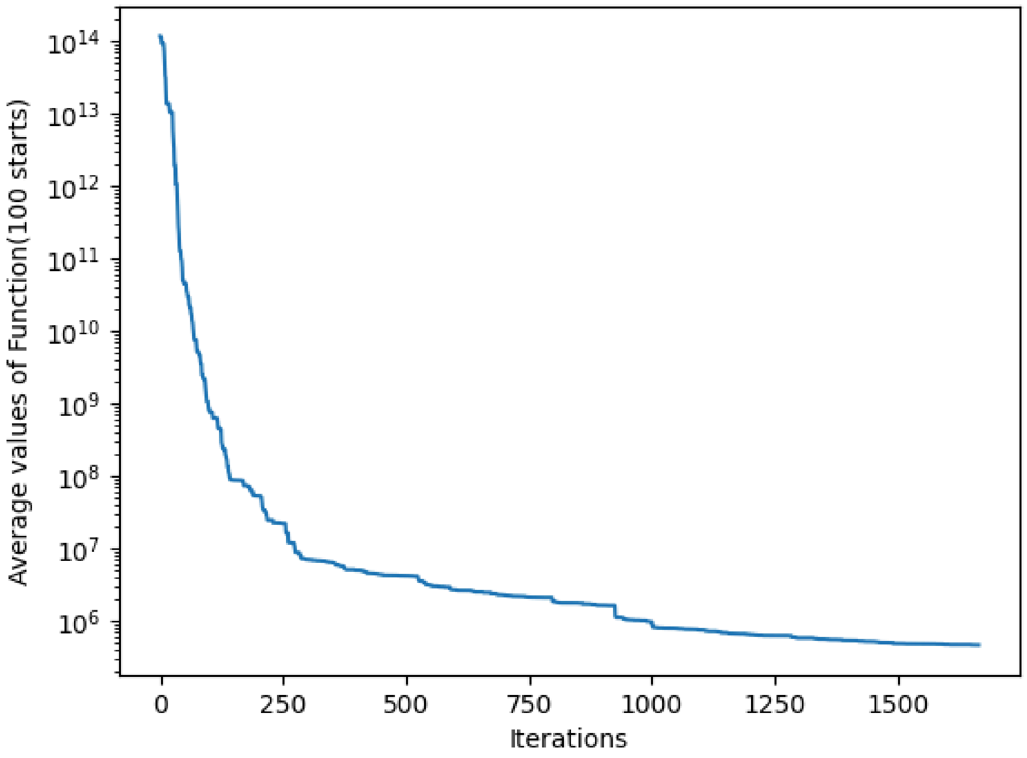
Graph for sigma=0.2



Graph for sigma=0.5



Graph for sigma=3



The graphs show that an increase in the sigma parameter significantly affects the rate of decrease in the value of the function for individuals at the beginning of the algorithm. But too high a sigma value makes it more difficult for the population to ultimately reach the global minimum of the function.

Table for sigma=0.5 and different population sizes
Function f13

population size	min	max	average	standard deviation
2	2403,98	43843,33	15251,21	9590,34
4	2707,21	59275,54	21606,33	13410,31
6	2881,27	55291,53	18022,54	12484,71
8	3412,44	55726,22	22171,12	12993,63
12	1988,72	34423,32	5621,52	5145,29
16	3348,32	37698,76	20574,32	9735,25
25	1784,21	27836,34	10360,29	7873,87
40	4370,12	16929,81	8333,45	2775,21
50	3680,98	18533,62	8925,71	3831,13

The population consisting of 12 individuals obtained the lowest average value. However, the minimum value of the function from all measurements was obtained by a population with 25 individuals. Despite this, the average value for 12 individuals is almost twice as small as for 25 individuals
.

I then examine the effect of changing the mutation strength for the best population found, i.e. 12 individuals.

Table for a fixed number of individuals of 12 and different sigma values
Function f13

sigma	min	max	average	standard deviation
0,05	30401,38	5624807118	56282468	559657784
0,1	9702,75	15334,43	12081,34	1356,65
0,2	1943,32	3777,12	2623,82	427,22
0,5	1988,72	34423,32	5621,52	5145,29
1	2060,32	54775,53	17532,32	9604.32
2	3177,15	84358,32	32070,43	17362,22
3	5029,44	86182,21	31683,98	18456,83

For a population of 12 individuals, the best results were obtained for a sigma parameter of 0.2. For increasingly larger parameter values (greater than 0.2), the algorithm obtained results with increasingly larger standard deviations and the value of the minimum found increased.

Conclusions:
For an appropriately selected number of populations and the value of the sigma parameter, responsible for the mutation strength of individuals, the algorithm can find a solution close to the global minimum of the function, which was achieved for the function f2, where the global minimum is equal to 200. Compared to the simple gradient method, it works much better for more complicated functions. The algorithm itself is not difficult to implement, especially since we do not use the mechanism of crossing between individuals, but only mutation and tournament selection. The results obtained for single launches may vary significantly, so when comparing different startup parameters, you should mainly look at average results from several dozen experiments.