Optimization

Cheng Su

February 26, 2020

1 Introduction

This report introduced several algorithm for optimization, including genetic algorithm and differential evolution algorithm. 18 functions are used as bench mark to test the performance of each algorithm.
The experiment is written in C++14, compiled with Clang 11.0.0. Random numbers are generated using Mersenne Twister(MT) pseudo-random number generator by Makoto Matsumoto.

System	macOS Catalina 10.15.3
Processor	2.5 GHz Dual-Core Intel Core i7
Memory	16GB 2133 MHz LPDDR3

Table 1: Hardware information

2 Algorithms

2.1 Genetic Algorithm

Genetic algorithm(GA) are the heuristic search and optimization techniques that mimic the process of natural evolution.
Basic steps of genetic algorithm consists of selection, crossover and mutation for each generation. Size of the population is kept constant by replacing the bad solution with a better solution. In selection step, parent with high fitness value are selected for crossover. In crossover step, both parents will be cut-off at a same randomly chosen point, and combining the inherited first half with the second half from the other parent to form two new individuals. Then the two individuals are mutated based on criteria.
For this experiment, genetic algorithm uses roulette wheel method during selection step, and differential evolution algorithm uses 10 different strategies. Each strategy shown below follows the convention of DE/x/y/z. x represents a string denoting the vector to be perturbed, y is the number of difference vectors considered for perturbation of x, and z is the type of crossover being used.

2.2 Differential Evolution algorithm

Differential evolution consist of three steps for each generation, including mutation, crossover and selection. During each generation, each vector generates a potential solution. During mutation step, a new parameter vector is generated by adding the weighted difference vector between two randomly selected population member to a third member. In exponential crossover, the crossover iterate through the loop of the vector within the CR bound. In binomial crossover, the crossover is performed on each variable with a probability determined by a random number compared against CR value. The potential solution is selected based on the cost compare to the current solution. Different strategies can be adopted depending on the type of problem applied.

Strategy	Formulation
Strategy 1: DE/best/1/exp	$v = x_{best}^G + F \cdot \left(x_{r2}^G - x_{r3}^G\right)$
Strategy 2: DE/rand/1/exp	$v = x_{r1}^G + F \cdot \left(x_{r2}^G - x_{r3}^G\right)$
Strategy 3: DE/rand-to-best/1/exp	$v = x_i^G + \lambda \cdot \left(x_{best}^G - x_i^G\right) + F \cdot \left(x_{r1}^G - x_{r2}^G\right)$
Strategy 4: DE/best/2/exp	$v = x_{best}^G + \lambda \cdot \left(x_{r1}^G + x_{r2}^G - x_{r3}^G - x_{r4}^G\right)$
Strategy 5: DE/rand/2/exp	$v = x_{r5}^G + \lambda \cdot \left(x_{r1}^G + x_{r2}^G - x_{r3}^G - x_{r4}^G\right)$
Strategy 6: DE/best/1/bin	$v = x_{best}^G + F \cdot \left(x_{r2}^G - x_{r3}^G\right)$
Strategy 7: DE/rand/1/bin	$v = x_{r1}^G + F \cdot \left(x_{r2}^G - x_{r3}^G\right)$
Strategy 8: DE/rand-to-best/1/bin	$v = x_i^G + \lambda \cdot \left(x_{best}^G - x_i^G\right) + F \cdot \left(x_{r1}^G - x_{r2}^G\right)$
Strategy 9: DE/best/2/bin	$v = x_{best}^G + \lambda \cdot \left(x_{r1}^G + x_{r2}^G - x_{r3}^G - x_{r4}^G\right)$
Strategy 10: DE/rand/2/bin	$v = x_{r5}^G + \lambda \cdot \left(x_{r1}^G + x_{r2}^G - x_{r3}^G - x_{r4}^G\right)$

Table 2: Differential Evolution strategies

2.3 Particle Swarm Optimization

Particle Swarm Optimization(PSO) is inspired by flocking and schooling patterns of bird and fish. Over a number of iterations, a group of variables have their value adjusted closer to the member who's value is closest to the target at any given moment. The algorithm keeps track of each particles best solution in its history, a global best solution found by particles in the population. For each iteration, a new velocity is calculated for each particle influenced by local optimal solution and global optimal solution so far. The current particle will be updated with the new velocity, local optimal and global optimal will also be updated if needed.

2.4 Moth-flame Optimization

Moth-flame Optimization(MFO) is inspired by navigation method of moth called traverse orientation, that it flies by maintaining a fixed angle with respect to the moon. A logarithmic spiral is the main update mechanism for this algorithm. Initial point of the spiral start from the moth, the final point ends at the position of the target, which is the flame. The moth can converge to any point in the neighbourhood of the flame, and the frequency of position update on the flame is increased as the moth get closer to the flame. For each iteration, position of the moth, position and number of flames are updated. Number of flame decreases linearly, removing the worst flame in each generation.

2.5 Cuckoo Search Optimization

Cuckoo Search Optimization(CS) is inspired by Cuckoo birds' reproductive strategy. For each iteration, one egg is altered based on Levy Flight random walk. Fitness of this egg will be compared with another egg randomly chosen from the nests, if the fitness of the new egg is higher, it will replace the other egg. Then a fraction p_a of worse nests will be abandoned and replaced with new eggs. Four algorithms used in this experiment are the basic p_a and three variants of p_a.p_{aCi} is the p_a for current iteration, p_{aMax} is the maximum value of p_a, C_i is current iteration and T_i is total number of iteration.

Strategy	Formulation	Equation
Strategy 0: CSCo	Constant	0.25
Strategy 1: CSCLI	Linear increasing	$p_{ac_i} = (p_{aMax}) * (C_i/T_i)$
Strategy 2: CSEI	Constant	$p_{ac_i} = (p_{aMax}) * Exp(C_i/T_i)$
Strategy 3: CSPI	Constant	$p_{ac_i} = (p_{aMax}) * (C_i/T_i)^3$

Table 3: Cuckoo Search switching paramter p_a

3 Benchmarks

$f_i(x^*)$ represents the optimal value for each function.

3.1 Schwefel's function

$$f_1(x) = (418.9829 \cdot n) - \sum_{i=1}^{n} -x_i \cdot sin(\sqrt{|x_i|}), -512 \leq x_i \leq 512 \quad (1)$$
$$f_1(x^*) = 0$$

3.2 1st De Jong's function

$$f_2(x) = \sum_{i=1}^{n} x_i^2, -100 \leq x_i \leq 100 \quad (2)$$
$$f_2(x^*) = 0$$

3.3 Rosenbrock

$$f_3(x) = \sum_{i=1}^{n-1} 100(x_i^2 - x_{i+1})^2 + (1 - x_i)^2, -100 \leq x_i \leq 100 \quad (3)$$
$$f_3(x^*) = 0$$

3.4 Rastrigin

$$f_4(x) = 10 \cdot n + \sum_{i=1}^{n} (x_i^2 - 10 \cdot cos(2\pi \cdot x_i)), -30 \leq x_i \leq 30 \quad (4)$$
$$f_4(x^*) = 0$$

3.5 Griewangk

$$f_5(x) = 1 + \sum_{i=1}^{n} x_i^2 - \prod_{i=1}^{n} cos\left(\frac{x_i}{\sqrt{i}}\right), -500 \leq x_i \leq -500 \quad (5)$$
$$f_5(x^*) = 0$$

3.6 Sine Envelope Sine Wave

$$f_6(x) = -\sum_{i=1}^{n-1} 0.5 + \frac{sin(x_i^2 + x_{i+1}^2 - 0.5)^2}{(1 + 0.001(x_i^2 + x_{i+1}^2))^2}, -30 \leq x_i \leq 30 \quad (6)$$
$$f_6(x^*) = -1.495(n-1)$$

3.7 Stretched V Sine Wave

$$f_7(x) = \sum_{i=1}^{n-1} \left(\sqrt[4]{x_i^2 + x_{i+1}^2} \cdot \left(sin50 \left(\sqrt[10]{x_i^2 + x_{i+1}^2} \right) \right)^2 + 1 \right), -30 \leq x_i \leq 30 \quad (7)$$
$$f_7(x^*) = 0$$

3.8 Ackley's One

$$f_8(x) = \sum_{i=1}^{n-1} \frac{1}{e^{0.2}} \sqrt{x_i^2 + x_{i+1}^2} + 3(cos(2x_i) + sin(2x_{i+1})), -32 \leq x_i \leq 32 \quad (8)$$
$$f_8(x^*) = -7.54276 - 2.91867(n-3)$$

3.9 Ackley's Two

$$f_9(x) = \sum_{i=1}^{n-1} \frac{1}{e^{0.2}} \sqrt{x_i^2 + x_{i+1}^2} + 3(cos(2x_i) + sin(2x_{i+1})), -32 \leq x_i \leq 32 \quad (9)$$
$$f_9(x^*) = 0$$

3.10 Egg Holder

$$f_10(x) = \sum_{i=1}^{n-1} -x_i \cdot sin\left(\sqrt{|x_i - x_{i+1} - 47|}\right) - (x_{i+1} + 47) \cdot sin\left(\sqrt{|x_{i+1} + 47 + \frac{x_i}{2}|}\right), -500 \leq x_i \leq 500 \tag{10}$$

3.11 Rana

$$f_11(x) = \sum_{i=1}^{n-1} x_i \cdot sin\left(\sqrt{|x_{i+1} - x_i + 1|}\right) \cdot cos\left(\sqrt{|x_{i+1} + x_i + 1|}\right) + (x_{i+1} + 1) \cdot cos\left(\sqrt{|x_{i+1} - x_i + 1|}\right) \cdot sin\left(\sqrt{|x_{i+1} + x_i + 1|}\right), -500 \leq x_i \leq 500 \tag{11}$$

3.12 Pathological

$$f_12(x) = \sum_{i=1}^{n-1} 0.5 + \frac{sin(\sqrt{100x_i^2 + x_{i+1}^2})^2 - 0.5}{1 + 0.001(x_i^2 - 2x_i \cdot x_{i+1} + x_{i+1}^2)^2}, -100 \leq x_i \leq 100 \tag{12}$$

3.13 Michalewicz

$$f_13(x) = \sum_{i=1}^{n} sin(x_i) \cdot \left(sin\left(\frac{i \cdot x_i^2}{\pi}\right)\right)^{20}, 0 \leq x_i \leq \pi \tag{13}$$
$$f_13(x^*) = 0.966n$$

3.14 Master's Cosine Wave

$$f_14(x) = -\sum_{i=1}^{n-1} e^{-\frac{1}{8}(x_i^2 + x_{i+1}^2 + 0.5x_{i+1} \cdot x_i)} cos\left(4\sqrt{x_i^2 + x_{i+1}^2 + 0.5x_{i+1} \cdot x_i}\right), -30 \leq x_i \leq 30 \tag{14}$$
$$f_14(x^*) = 1 - n$$

3.15 Quartic

$$f_15(x) = \sum_{i=1}^{n} (i \cdot x_i^4), -100 \leq x_i \leq 100 \tag{15}$$
$$f_15(x^*) = 0$$

3.16 Levy

$$f_16(x) = sin^2(\pi w_i) + \sum_{i=1}^{n-1} (w_i - 1)^2[1 + 10 \cdot sin^2(\pi w_i + 1)] + (w_i - 1)^2[1 + sin^2(2\pi w_n)] \text{ , where: } w_i = 1 + \frac{x_i - 1}{4}, -10 \leq x_i \leq 10 \tag{16}$$
$$f_16(x^*) = 0$$

3.17 Step

$$f_17(x) = \sum_{i=0}^{n-1} (|x_i| + 0.5)^2, -100 \leq x_i \leq 100 \tag{17}$$
$$f_17(x^*) = 0$$

3.18 Alpine

$$f_18(x) = \sum_{i=0}^{n-1} |x_i \cdot sin(x_i) + 0.1 \cdot x_i|, -100 \leq x_i \leq 100$$
$$f_18(x^*) = 0 \tag{18}$$

4 Experimentation

Below are the parameters for experimentation. Random numbers are generated using Mersenne Twister. Results includes average, standard deviation, range, and median of the function value, and the execution time for each experiment in microseconds.

4.1 GA and DE

Parameters	Values
Experimentation	50
Population size	200
Generation	100
Dimension	30

Table 4: GA and DE parameters

4.2 PSO and MFO

Parameters used for PSO are $c1 = 0.5$, $c2 = 2.0$.

Parameters	Values
Experimentation	50
Population size	500
Generation	1000
Dimension	30

Table 5: PSO and MFO parameters

4.3 CS

Parameters used for CS are $alpha0 = 0.01$, $pa_max = 0.25$.

Parameters	Values
Experimentation	50
Population size	200
Generation	2500
Dimension	30

Table 6: PSO and MFO parameters

5 Result

Below are the results for genetic algorithm and differential evolution algorithm for each benchmark. Result of the strategy with lowest mean value is selected as the optimal result for each function in differential algorithm.

f	Mean	Median	Std	Range(low)	Range(high)	Time(mus)
1	1541.2	1571.67	333.191	747.995	2239.77	145.48
2	2211.76	2201.46	1000.77	569.597	5496.91	117.08
3	2.05158e+08	1.59969e+08	1.57839e+08	2.10619e+07	8.79938e+08	136.68
4	306.938	313.772	87.7215	157.944	565.927	140.3
5	10.9869	9.68638	4.04782	3.70623	20.6534	145.84
6	-37.5568	-37.5092	1.08383	-39.7491	-34.3093	143.16
7	30.9784	30.938	0.61566	29.6639	32.5812	209
8	-21.4299	-22.2676	14.125	-47.3761	26.3798	191.76
9	144.976	143.509	21.4298	103.591	200.278	210.28
10	-16992.3	-16685	877.951	-19060.9	-14869.1	167.36
11	-10552.6	-10593	426.656	-11362.8	-9464.1	190.9
12	-10.6553	-10.3169	2.01158	-15.2122	-6.89124	156.96
13	-26.2436	-26.2669	0.62964	-27.6397	-24.539	206.8
14	-11.9625	-11.967	1.9795	-18.3732	-7.56829	191.38
15	4.20667e+07	3.35967e+07	2.87514e+07	2.97705e+06	1.14974e+08	147.34
16	5.97079	4.92521	4.51866	1.44701	31.6831	213.04
17	1893.74	1744.83	842.046	623.461	4666.41	120.94
18	27.1616	26.1614	5.29169	18.627	40.7951	131.9

Table 7: Result for genetic algorithm

f	Mean	Median	Std	Range(low)	Range(high)	Time(mus)
1	2429.76	2138.5	1123.87	692	5533	378.42
2	0.28	0	0.448999	0	1	353.86
3	6.93525e+07	6.94453e+07	1.83696e+07	3.52724e+07	1.37235e+08	452.94
4	140.5	132.5	57.3631	45	248	512.92
5	1.02	1	0.14	1	2	351
6	-28.08	-28	0.716659	-30	-27	362.68
7	43.58	44	1.31286	41	46	619.1
8	-53.46	-54	12.64	-74	-25	530.52
9	37.5	30	12.3422	30	69	564.26
10	-13261.8	-14341.5	3136.38	-17814	-5806	568.14
11	-6067.54	-6026	396.64	-7467	-5340	425.58
12	2.16	2	0.966644	0	4	378.12
13	-13.72	-14	0.775629	-16	-12	398.7
14	-4.08	-4	0.483322	-5	-3	393.66
15	0.06	0	0.237487	0	1	353.08
16	0.44	0	0.535164	0	2	400.06
17	8.38	8	0.485386	8	9	521.1
18	30.58	20.5	26.8612	3	130	530.08

Table 8: Result for differential evolution algorithm

Function	Mean	Median	Std	Range(low)	Range(high)	Time(mus)
1	3228.34	3326.44	490.507	2150.59	4595.97	1443.26
2	1200.38	0.0118296	3249.47	1.4927e-08	10000	1234.94
3	2.07458e+09	2.14748e+09	2.63196e+08	6.98833e+08	2.14748e+09	1289.04
4	370.326	196.503	358.22	101.485	1117.89	1522.4
5	8.89763	0.0994725	21.7757	8.24674e-13	63.3424	1609.78
6	-31.2185	-31.4953	1.71111	-35.2914	-27.0298	1574.76
7	30.5886	29.8405	1.72267	29.0012	34.7174	2418.4
8	39.9739	40.3441	33.8953	-17.2951	114.908	2142.6
9	245.261	253.663	52.0166	111.107	339.365	2263.4
10	-20560.4	-20581.2	1677.94	-24400.5	-17279	1669.42
11	-13658.1	-13661.6	94.6517	-14060.4	-13475.1	1944.52
12	-0.228768	-0.483867	2.49542	-7.8833	4.58436	1612.36
13	-15.3963	-15.5315	1.4753	-18.3317	-11.8207	1849.16
14	-1.8779	-1.77067	0.410949	-2.94956	-1.27895	1988.1
15	1.42329e+08	4.34527e+07	1.89165e+08	1784.66	9e+08	1619.82
16	22.6544	22.7147	7.05471	9.72472	43.1741	2494.64
17	1222.49	9.21536	3281.02	7.50411	10107.6	1251.5
18	47.2483	40.6376	48.2386	0.000912548	162.547	1545.42

Table 9: Result for particle swarm optimization algorithm

Function	Mean	Median	Std	Range(low)	Range(high)	Time(mus)
1	3362.68	3331.31	66.2482	3213.1	3442.61	2733.64
2	5747.58	5683.5	1702.59	2385.87	10405.1	2513.8
3	5.22249e+08	4.67845e+08	2.61434e+08	9.48906e+07	1.0774e+09	1860.66
4	834.404	819.938	165.179	586.477	1357.83	1915.22
5	36.3898	34.4532	11.3639	17.9603	59.2523	2022.12
6	-22.3977	-22.304	0.481364	-23.9005	-21.5423	2003.74
7	60.7445	61.1031	2.03632	56.1954	64.4033	2929.2
8	123.431	124.565	24.4519	72.4132	179.634	2451.32
9	311.113	313.429	26.9556	262.363	391.099	2810.86
10	-18347.3	-18400	1304.22	-21656.4	-14536.2	2528.98
11	-13539.2	-13554.7	93.6786	-13686.9	-13278.3	2687.1
12	-8.7762	-8.95505	1.2941	-11.794	-6.09565	2209.22
13	-9.94114	-9.91784	0.391355	-10.9376	-9.19078	2803.18
14	-2.81039	-2.62756	0.616388	-5.09268	-2.15578	2679.4
15	1.12282e+08	9.41429e+07	6.48114e+07	2.50417e+07	3.25568e+08	2107.16
16	41.2083	38.5155	14.4798	18.9525	79.6705	2876.66
17	5743.79	5510.27	1621.83	2525.78	9091.88	1686.48
18	297.308	299.744	32.2621	225.295	398.598	1858.2

Table 10: Result for moth-flame optimization algorithm

Function	Mean	Median	Std	Range(low)	Range(high)	Time(mus)
1	6367.89	6312.22	669.198	5206.11	7446.51	457.1
2	6303.49	6249.3	1607.84	3404.9	8998.93	409.5
3	3.78296e+08	2.93369e+08	2.18587e+08	1.69398e+08	8.37215e+08	410.8
4	879.077	828.984	141.1	697.017	1232.19	493.7
5	39.4563	38.0311	8.73215	28.3238	57.9285	453.9
6	-24.1341	-24.1496	0.702241	-25.4218	-22.785	557.5
7	49.7245	49.4185	1.90357	45.8168	52.7129	553.7
8	124.088	119.377	16.3176	93.2002	148.237	537.6
9	366.261	363.994	26.491	317.096	430.775	555.1
10	-7881.06	-7968.45	768.083	-9165.44	-6763.61	491.1
11	-7794.09	-7907.32	697.305	-8940.42	-6422.15	547.8
12	-6.38171	-6.18596	2.46735	-11.1153	-3.13929	367.8
13	-12.074	-12.0536	0.410743	-12.7058	-11.4866	555.9
14	-2.84104	-2.65219	0.601934	-4.41615	-2.35834	644.5
15	4.81384e+07	4.34677e+07	2.10456e+07	2.04608e+07	8.46703e+07	449.9
16	52.4811	46.306	11.918	42.5047	81.3921	547.9
17	6161.41	5976.5	1255.92	4622.96	8164.82	407.5
18	181.914	186.362	20.1549	155.861	215.467	450.2

Table 11: Result for cuckoo search optimization algorithm

5.1 Schwefel's

Strategy	Mean	Median	Std	low	high	Time(mus)
GA	1541.2	1571.67	333.191	747.995	2239.77	145.48
DE1	2429.76	2138.5	1123.87	692	5533	378.42
DE2	6716.66	6718.5	256.213	5993	7187	538.44
DE3	6977.38	6974.5	350.304	5775	7659	381.36
DE4	6313.64	6313.5	331.01	5619	7101	626.02
DE5	6862.48	6881.5	273.613	6089	7359	789.46
DE6	2236.5	2225	473.109	1402	3554	519.6
DE7	8015.14	8115.5	317.602	7166	8549	796.28
DE8	7813.76	7811.5	449.894	6479	8709	559.68
DE9	7002.08	7074	562.477	5737	8067	926.12
DE10	7995.68	8050.5	269.776	7325	8430	1241.2
PSO	3228.34	3326.44	490.507	2150.59	4595.97	1443.26
MFO	3362.68	3331.31	66.2482	3213.1	3442.61	2733.64
CS	6367.89	6312.22	669.198	5206.11	7446.51	457.1

Table 12: Results for Schwefel

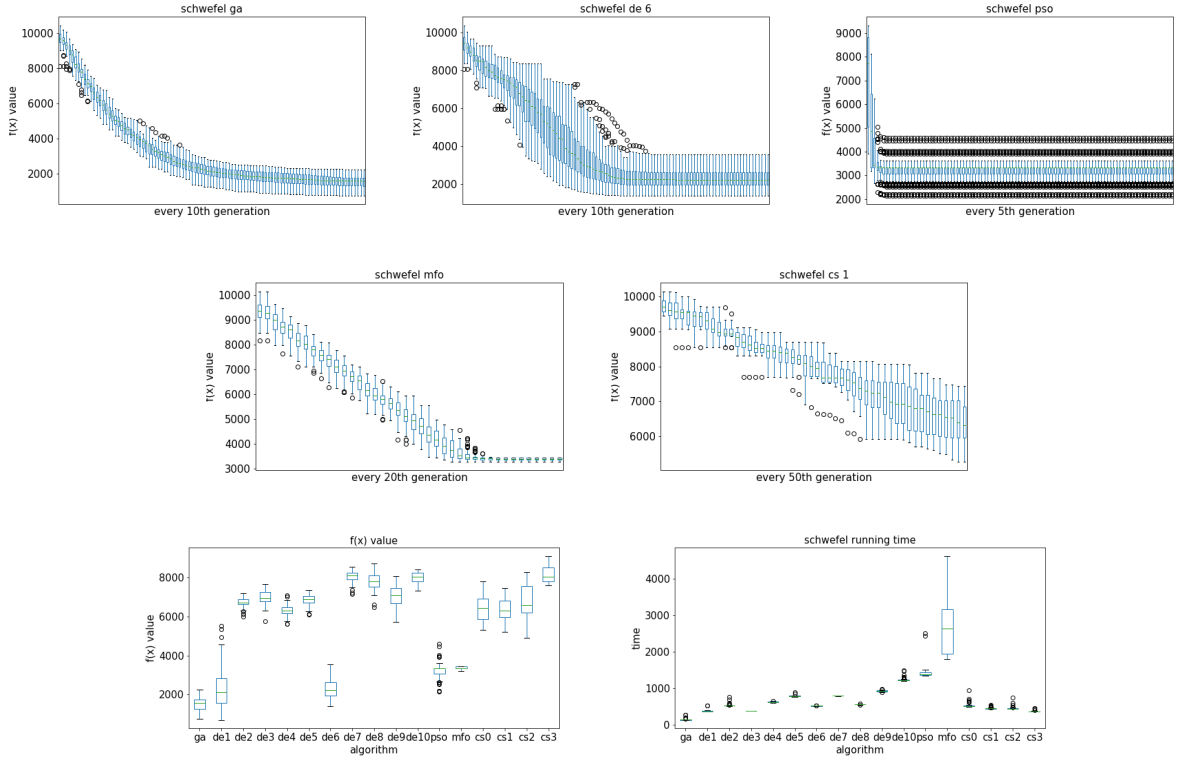

Figure 1: Results for Schwefel's function

5.2 1st De Jong's

Strategy	Mean	Median	Std	low	high	Time(mus)
GA	2211.76	2201.46	1000.77	569.597	5496.91	117.08
DE1	0.28	0	0.448999	0	1	353.86
DE2	1514.64	1504.5	176.132	1071	1802	448.8
DE3	0	0	0	0	0	347.88
DE4	852.16	878.5	182.892	434	1211	485.72
DE5	7038.38	6949.5	742.115	5591	8697	640.62
DE6	0.84	1	0.611882	0	3	494.56
DE7	2225.4	2192	260.237	1681	2760	685.1
DE8	0	0	0	0	0	516.94
DE9	835.54	812	298.862	296	1698	743.86
DE10	13432.5	13517.5	1230.12	9896	16983	1004.6
PSO	1200.38	0.0118296	3249.47	1.4927e-08	10000	1234.94
MFO	5747.58	5683.5	1702.59	2385.87	10405.1	2513.8
CS	6303.49	6249.3	1607.84	3404.9	8998.93	409.5

Table 13: Results for 1st De Jong

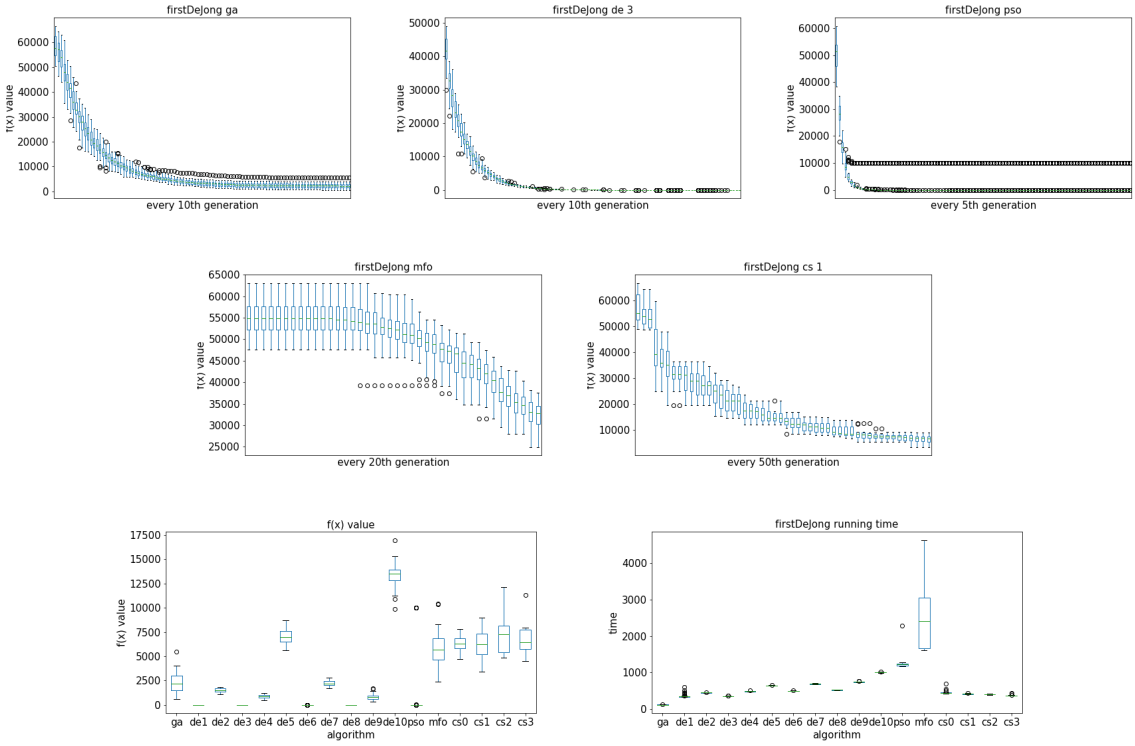

Figure 2: Results for 1st De Jong's function

5.3 Rosenbrock

Strategy	Mean	Median	Std	low	high	Time(mus)
GA	2.05158e+08	1.59969e+08	1.57839e+08	2.10619e+07	8.79938e+08	136.68
DE1	1.14159e+09	2.14748e+09	1.04928e+09	1159	2.14748e+09	339
DE2	6.93525e+07	6.94453e+07	1.83696e+07	3.52724e+07	1.37235e+08	452.94
DE3	8.45684e+08	2.45048e+08	9.85743e+08	258	2.14748e+09	351.04
DE4	8.10328e+08	4.50003e+08	7.42972e+08	3.88069e+07	2.14748e+09	507.46
DE5	6.67499e+08	6.50786e+08	1.32776e+08	4.26236e+08	9.47807e+08	646.78
DE6	2.06159e+09	2.14748e+09	4.20815e+08	181	2.14748e+09	501.24
DE7	2.0798e+08	2.10397e+08	5.68643e+07	7.57484e+07	3.17689e+08	693.84
DE8	2.14748e+09	2.14748e+09	2.32896e+08	2.14748e+09	2.14748e+09	519.74
DE9	1.91844e+09	2.14748e+09	5.93588e+08	9.46526e+07	2.14748e+09	771.3
DE10	1.8042e+09	1.82414e+09	3.17115e+08	1.06765e+09	2.14748e+09	1016.94
PSO	2.07458e+09	2.14748e+09	2.63196e+08	6.98833e+08	2.14748e+09	1289.04
MFO	5.22249e+08	4.67845e+08	2.61434e+08	9.48906e+07	1.0774e+09	1860.66
CS	3.78296e+08	2.93369e+08	2.18587e+08	1.69398e+08	8.37215e+08	410.8

Table 14: Results for Rosenbrock

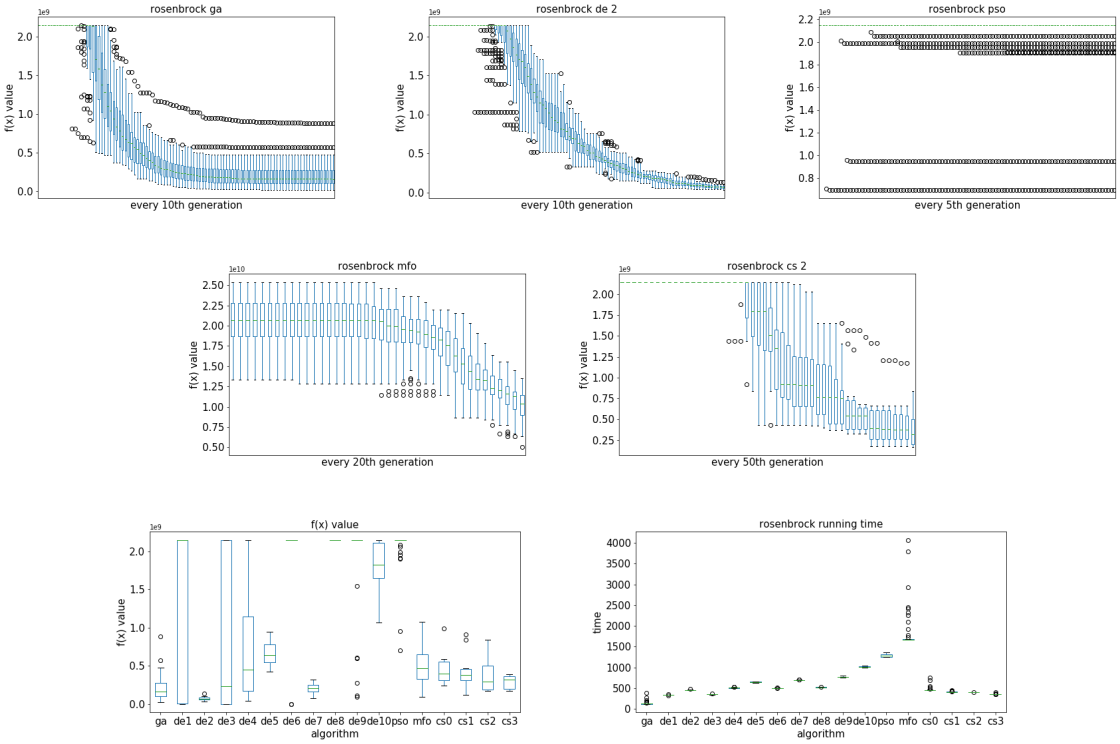

Figure 3: Results for Rosenbrock function

5.4 Rastrigin

Strategy	Mean	Median	Std	low	high	Time(mus)
GA	306.938	313.772	87.7215	157.944	565.927	140.3
DE1	183.4	183	16.1009	144	222	359.68
DE2	387.72	388	25.1619	304	439	467.5
DE3	168.14	170	9.81837	142	188	366.12
DE4	366.72	368	30.9277	301	458	504.22
DE5	910.76	924.5	93.45	641	1052	660.12
DE6	140.5	132.5	57.3631	45	248	512.92
DE7	480.08	479	31.6239	412	528	711.54
DE8	195.52	195	16.0975	165	234	545.52
DE9	390.44	388.5	46.3462	302	554	770.3
DE10	1430.82	1426.5	152.914	1088	1823	1037.76
PSO	370.326	196.503	358.22	101.485	1117.89	1522.4
MFO	834.404	819.938	165.179	586.477	1357.83	1915.22
CS	879.077	828.984	141.1	697.017	1232.19	493.7

Table 15: Results for Rastrigin

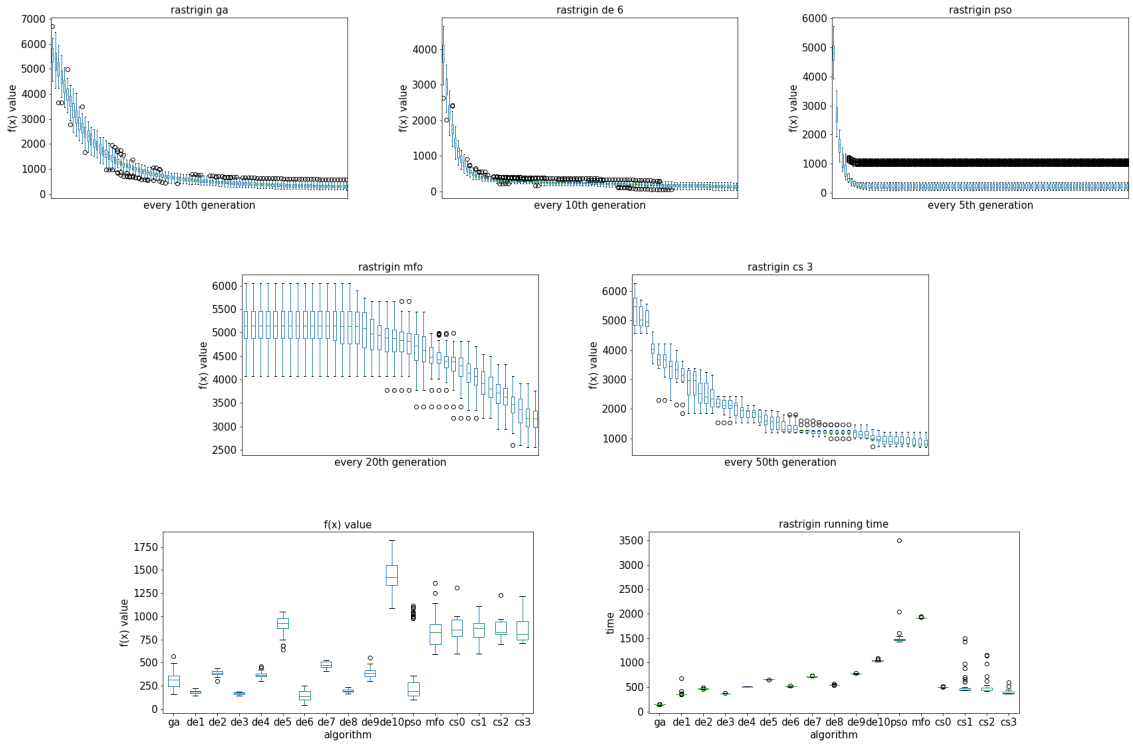

Figure 4: Results for Rastrigin function

5.5 Griewangk

Strategy	Mean	Median	Std	low	high	Time(mus)
GA	10.9869	9.68638	4.04782	3.70623	20.6534	145.84
DE1	1.02	1	0.14	1	2	351
DE2	9.6	9	1.2	7	12	467.32
DE3	1	1	0	1	1	363.26
DE4	5.74	6	1.30859	3	8	506.96
DE5	44.88	45	5.01055	30	56	660.76
DE6	1.84	2	0.578273	1	3	514.4
DE7	14.38	15	1.49519	10	17	705.3
DE8	1	1	0	1	1	532.8
DE9	4.88	5	1.53154	2	8	763.4
DE10	83.1	84	8.44097	60	100	1023.44
PSO	8.89763	0.0994725	21.7757	8.24674e-13	63.3424	1609.78
MFO	36.3898	34.4532	11.3639	17.9603	59.2523	2022.12
CS	39.4563	38.0311	8.73215	28.3238	57.9285	453.9

Table 16: Results for Griewangk

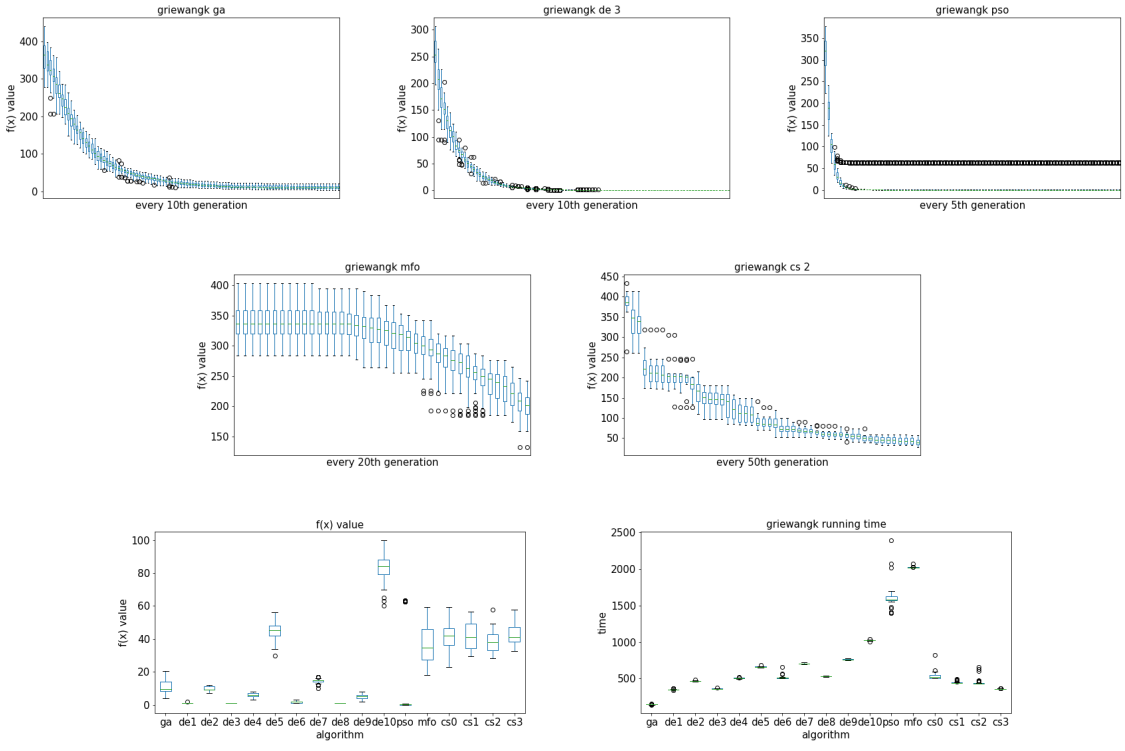

Figure 5: Results for Greiwangk function

5.6 Sine Envelope Sine Wave

Below are results for Sine Envelope Sine Wave.

Strategy	Mean	Median	Std	low	high	Time(mus)
GA	-37.5568	-37.5092	1.08383		-34.3093-39.7491	143.16
DE1	-26.82	-27	0.864639	-25	-28	366.06
DE2	-26.68	-27	0.581034	-26	-28	496.24
DE3	-28.08	-28	0.716659	-27	-30	362.68
DE4	-26.7	-26.5	0.8544	-26	-30	560.06
DE5	-26.34	-26	0.586856	-25	-28	738.82
DE6	-23.6	-23.5	0.959166	-22	-27	538.86
DE7	-23.54	-23	0.753923	-22	-26	759.08
DE8	-25	-25	0.959166	-23	-28	538.06
DE9	-23.14	-23	0.774855	-22	-25	868.98
DE10	-23.34	-23	0.681469	-22	-25	1158.52
PSO	-31.2185	-31.4953	1.71111	-35.2914	-27.0298	1574.76
MFO	-22.3977	-22.304	0.481364	-23.9005	-21.5423	2003.74
CS	-24.1341	-24.1496	0.702241	-25.4218	-22.785	557.5

Table 17: Results for Sine Envelope Sine Wave

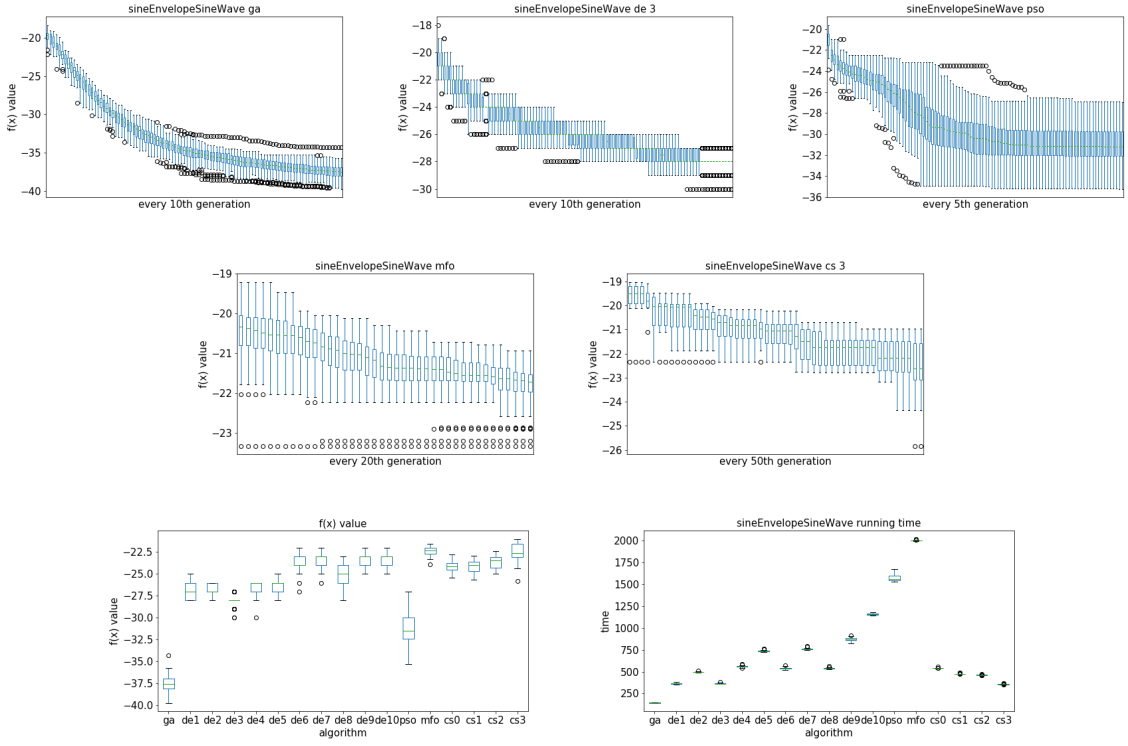

Figure 6: Results for Sine Envelope Sine Wave function

5.7 Stretched V Sine Wave

Strategy	Mean	Median	Std	low	high	Time(mus)
GA	30.9784	30.938	0.61566	29.6639	32.5812	209
DE1	45.72	45	1.9083	42	51	470.58
DE2	46.02	46.5	1.59361	42	49	556.1
DE3	43.58	44	1.31286	41	46	619.1
DE4	46.8	47	1.35647	43	49	831.22
DE5	46	46	1.78885	42	49	772.34
DE6	53.82	54	3.88685	46	63	581.42
DE7	54.16	54	1.79287	50	58	797.12
DE8	50.3	50	2.46779	46	56	583.34
DE9	55.72	56	2.18211	48	59	906.66
DE10	55.36	56	2.42289	49	59	1186.78
PSO	30.5886	29.8405	1.72267	29.0012	34.7174	2418.4
MFO	60.7445	61.1031	2.03632	56.1954	64.4033	2929.2
CS	49.7245	49.4185	1.90357	45.8168	52.7129	553.7

Table 18: Results for Stretched V Sine Wave

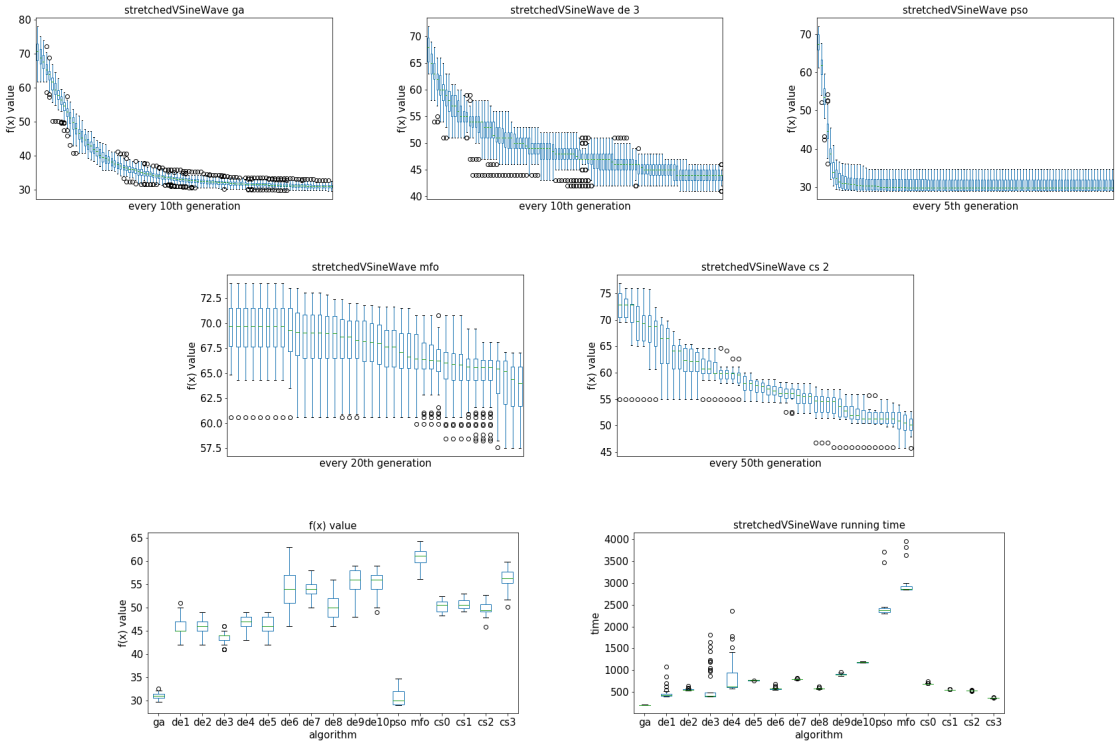

Figure 7: Results for Stretched V Sine Wave function

5.8 Ackley's One

Strategy	Mean	Median	Std	low	high	Time(mus)
GA	-21.4299	-22.2676	14.125	-47.3761	26.3798	191.76
DE1	-43.58	-44	16.5917	-73	-13	368.88
DE2	50.3	51	7.07177	33	62	490.38
DE3	-20.48	-21.5	5.84205	-33	2	380.88
DE4	51.56	52	9.51874	26	66	522.92
DE5	122.3	125	12.2723	79	141	680.7
DE6	-53.46	-54	12.64	-74	-25	530.52
DE7	77.06	78	9.40725	50	92	727.12
DE8	-44.12	-44.5	15.7704	-75	-10	550.42
DE9	74.94	73	14.5484	34	106	782.22
DE10	181.16	183	13.4883	151	215	1048.6
PS0	39.9739	40.3441	33.8953	-17.2951	114.908	2142.6
MFO	123.431	124.565	24.4519	72.4132	179.634	2451.32
CS	124.088	119.377	16.3176	93.2002	148.237	537.6

Table 19: Results for Ackley's One

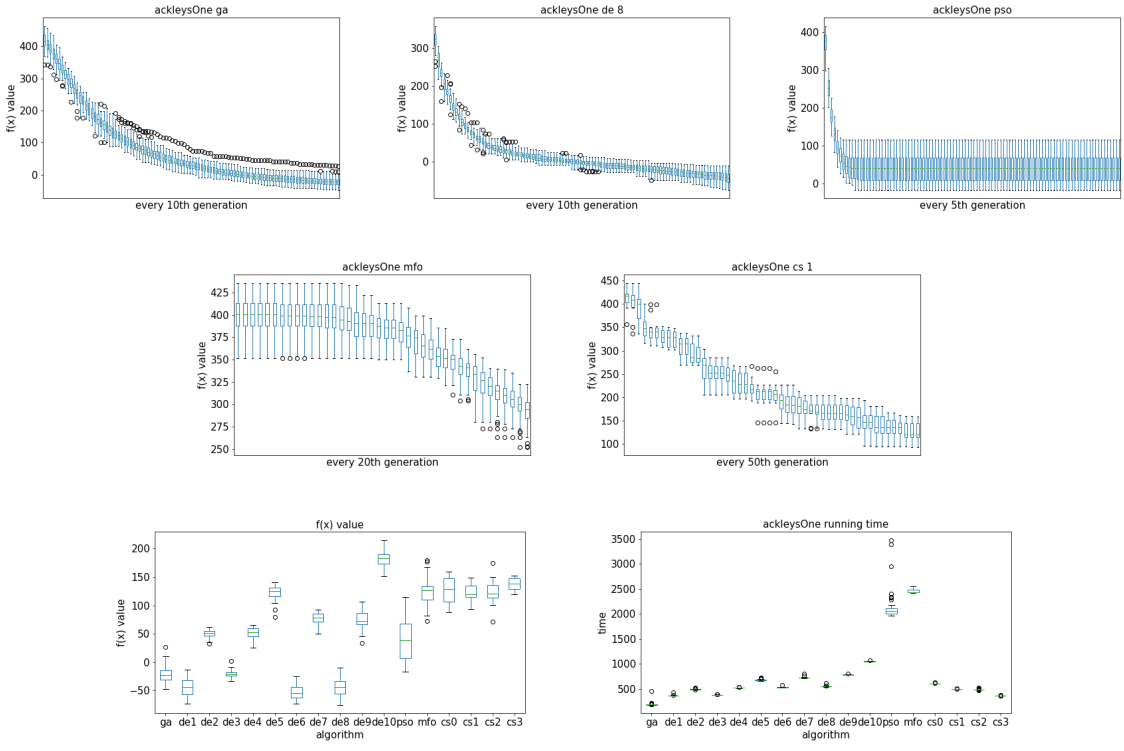

Figure 8: Results for Ackley's One function

5.9 Ackley's Two

Strategy	Mean	Median	Std	low	high	Time(mus)
GA	144.976	143.509	21.4298	103.591	200.278	210.28
DE1	35.98	31	12.7365	30	88	384.2
DE2	208.28	210	10.837	178	228	504.8
DE3	30.74	31	0.558928	30	32	395.28
DE4	216.26	217	14.7048	184	247	546.46
DE5	336.96	337	11.0054	300	363	701.58
DE6	66.24	71.5	33.2407	30	185	537.92
DE7	270.7	271	11.411	251	301	743.46
DE8	37.5	30	12.3422	30	69	564.26
DE9	243.62	245.5	28.7965	180	316	818.06
DE10	405.22	406.5	12.3374	364	433	1090.66
PS0	245.261	253.663	52.0166	111.107	339.365	2263.4
MFO	311.113	313.429	26.9556	262.363	391.099	2810.86
CS	366.261	363.994	26.491	317.096	430.775	555.1

Table 20: Results for Ackley's Two

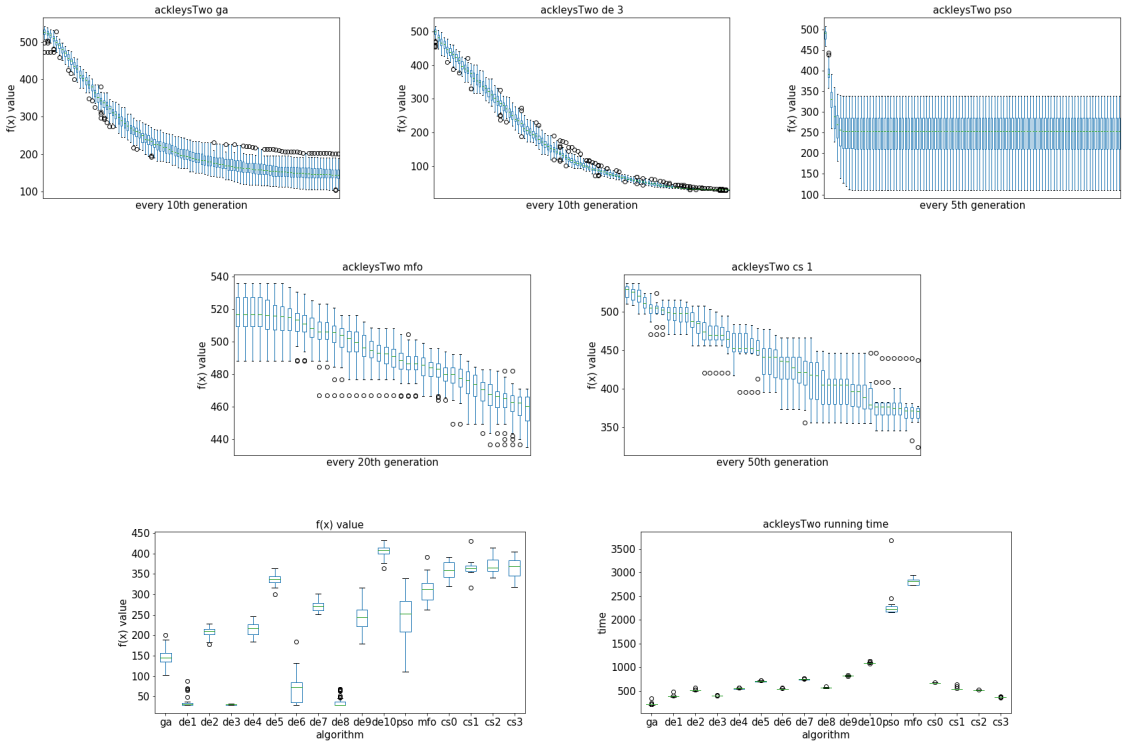

Figure 9: Results for Ackley's Two function

5.10 Egg Holder

Strategy	Mean	Median	Std	low	high	Time(mus)
GA	-16992.3	-16685	877.951	-19060.9	-14869.1	167.36
DE1	-10530.2	-10266	1222.07	-13567	-8605	416.48
DE2	-8298.82	-8240.5	493.525	-9963	-7372	545.86
DE3	-7980.2	-7847	555.938	-9784	-6966	403.16
DE4	-8940.38	-8971	457.441	-9801	-7999	655.68
DE5	-8345.98	-8295.5	465.153	-9561	-7518	804.7
DE6	-13261.8	-14341.5	3136.38	-17814	-5806	568.14
DE7	-6148.84	-5971.5	586.491	-8862	-5320	817.52
DE8	-5922	-5979	455.404	-7133	-4832	585.78
DE9	-6977.12	-6992.5	649.352	-8720	-5703	968.18
DE10	-6253.26	-6250	452.354	-7485	-5346	1244.88
PSO	-20560.4	-20581.2	1677.94	-24400.5	-17279	1669.42
MFO	-18347.3	-18400	1304.22	-21656.4	-14536.2	2528.98
CS	-7881.06	-7968.45	768.083	-9165.44	-6763.61	491.1

Table 21: Results for Egg Holder

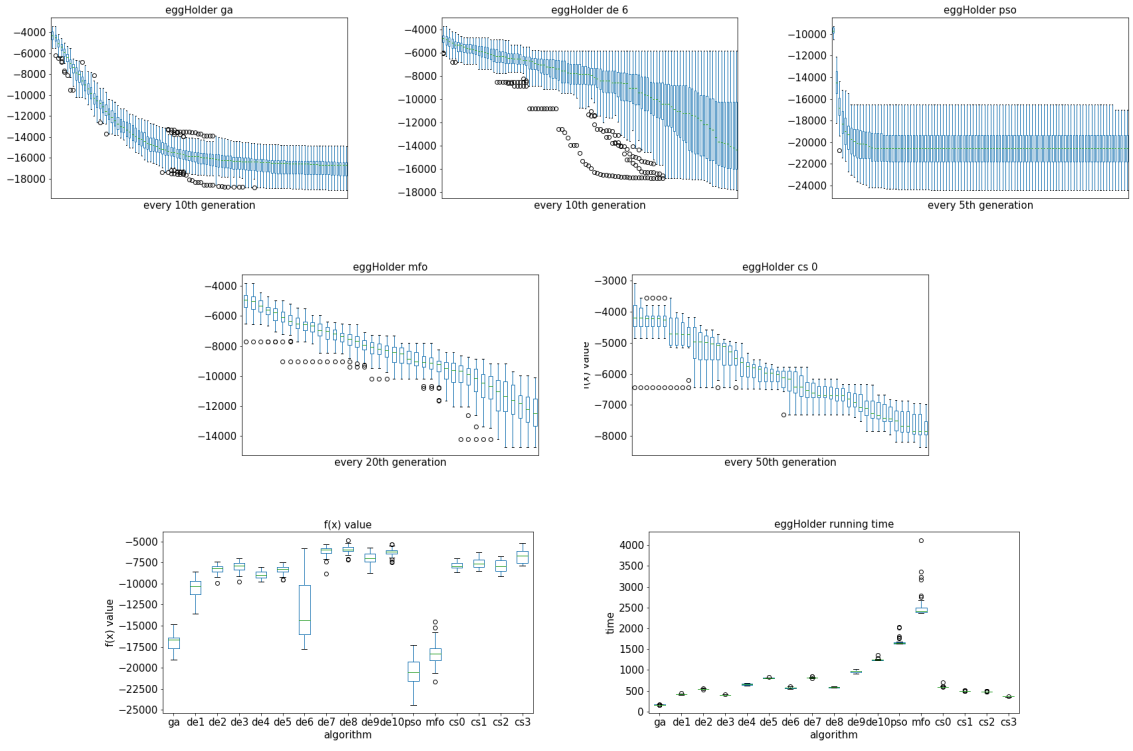

Figure 10: Results for Egg Holder function

5.11 Rana

Strategy	Mean	Median	Std	low	high	Time(mus)
GA	-10552.6	-10593	426.656	-11362.8	-9464.1	190.9
DE1	-6067.54	-6026	396.64	-7467	-5340	425.58
DE2	-5538.16	-5488	306.079	-6641	-5075	553
DE3	-5340.36	-5316	346.355	-6061	-4698	412.64
DE4	-5840.76	-5845.5	263.624	-6537	-5318	662.66
DE5	-5589.44	-5546.5	301.843	-6319	-4828	813.06
DE6	-4914.24	-4822.5	450.765	-6238	-4117	602.5
DE7	-4117.96	-4055.5	270.088	-4743	-3605	856.36
DE8	-3992.12	-3894.5	375.701	-5033	-3472	601.52
DE9	-4417.9	-4406	251.964	-5157	-3932	988.34
DE10	-4159.18	-4085	301.408	-5024	-3758	1252.78
PSO	-13658.1	-13661.6	94.6517	-14060.4	-13475.1	1944.52
MFO	-13539.2	-13554.7	93.6786	-13686.9	-13278.3	2687.1
CS	-7794.09	-7907.32	697.305	-8940.42	-6422.15	547.8

Table 22: Results for Rana

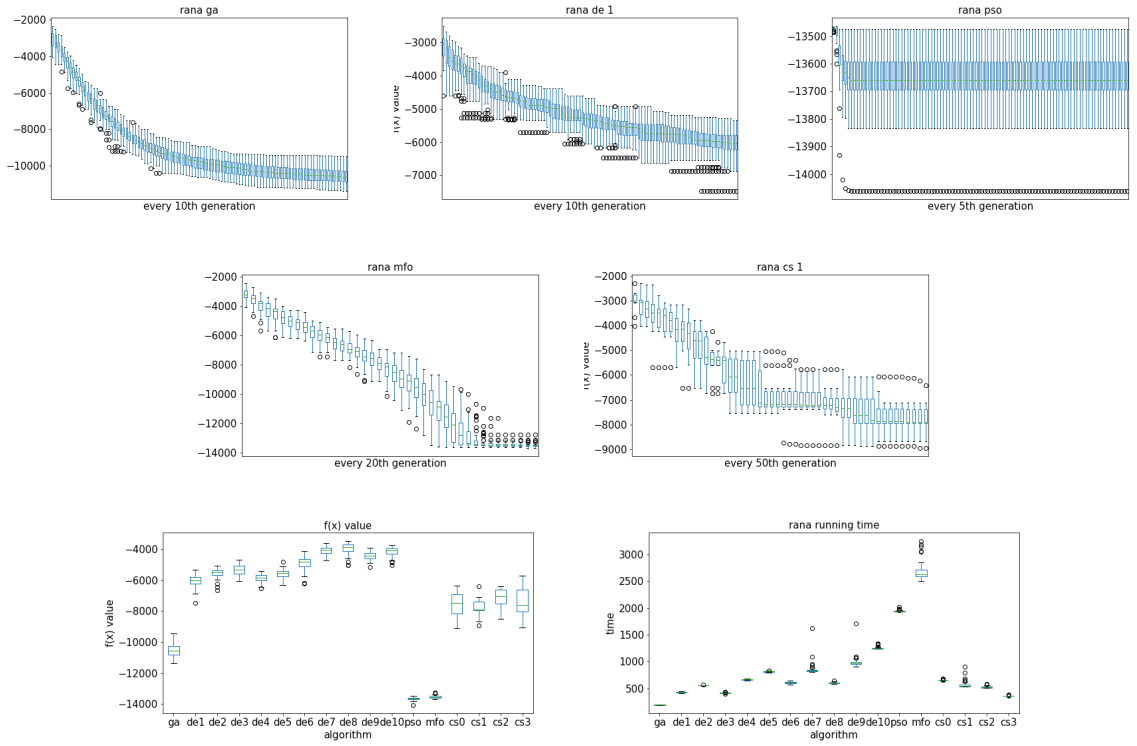

Figure 11: Results for Rana function

5.12 Pathological

Strategy	Mean	Median	Std	low	high	Time(mus)
GA	-10.6553	-10.3169	2.01158	-15.2122	-6.89124	156.96
DE1	2.16	2	0.966644	0	4	378.12
DE2	3.54	4	0.698856	1	5	510.2
DE3	2.46	3	0.698856	1	4	375.92
DE4	3.44	3.5	0.778717	2	5	583.44
DE5	3.38	3	0.66	2	4	756.3
DE6	5.02	5	1.31894	1	8	541.46
DE7	6.44	7	0.668132	5	7	776.14
DE8	5.6	6	0.774597	4	7	556.38
DE9	6.32	6	0.545527	5	7	893.96
DE10	6.58	7	0.602993	5	8	1183.9
PSO	-0.228768	-0.483867	2.49542	-7.8833	4.58436	1612.36
MFO	-8.7762	-8.95505	1.2941	-11.794	-6.09565	2209.22
CS	-6.38171	-6.18596	2.46735	-11.1153	-3.13929	367.8

Table 23: Results for Pathological

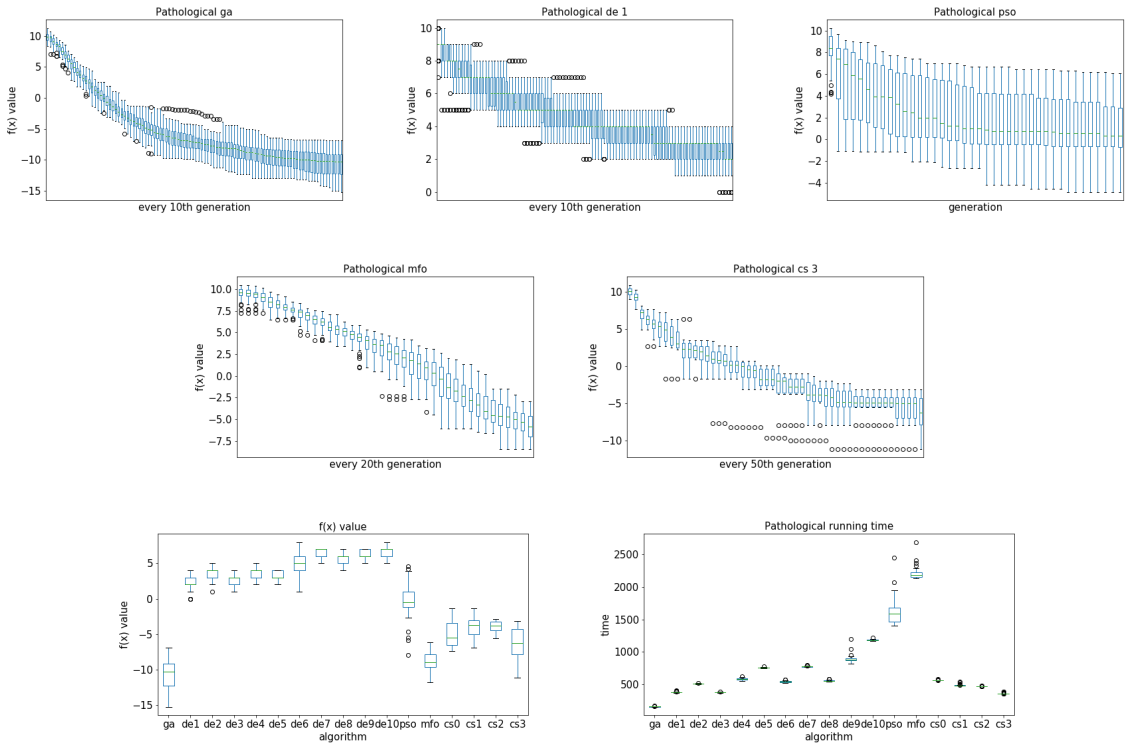

Figure 12: Results for Pathological function

5.13 Michalewicz

Strategy	Mean	Median	Std	low	high	Time(mus)
GA	-26.2436	-26.2669	0.62964	-24.539	-27.6397	206.8
DE1	-13.72	-14	0.775629	-12	-16	398.7
DE2	-13.34	-13	0.62	-12	-16	523.54
DE3	-14.08	-14	0.56	-13	-15	397.52
DE4	-12.92	-13	0.688186	-12	-14	593.3
DE5	-12.8	-13	0.565685	-12	-14	760.46
DE6	-12.02	-12	0.734575	-11	-14	565.12
DE7	-11.12	-11	0.65238	-10	-13	784.42
DE8	-12.42	-12	0.750733	-11	-15	572.8
DE9	-10.78	-11	0.729109	-10	-13	896.58
DE10	-10.66	-11	0.586856	-10	-12	1169.34
PSO	-15.3963	-15.5315	1.4753	-18.3317	-11.8207	1849.16
MFO	-9.94114	-9.91784	0.391355	-10.9376	-9.19078	2803.18
CS	-12.074	-12.0536	0.410743	-12.7058	-11.4866	555.9

Table 24: Results for Michalewicz

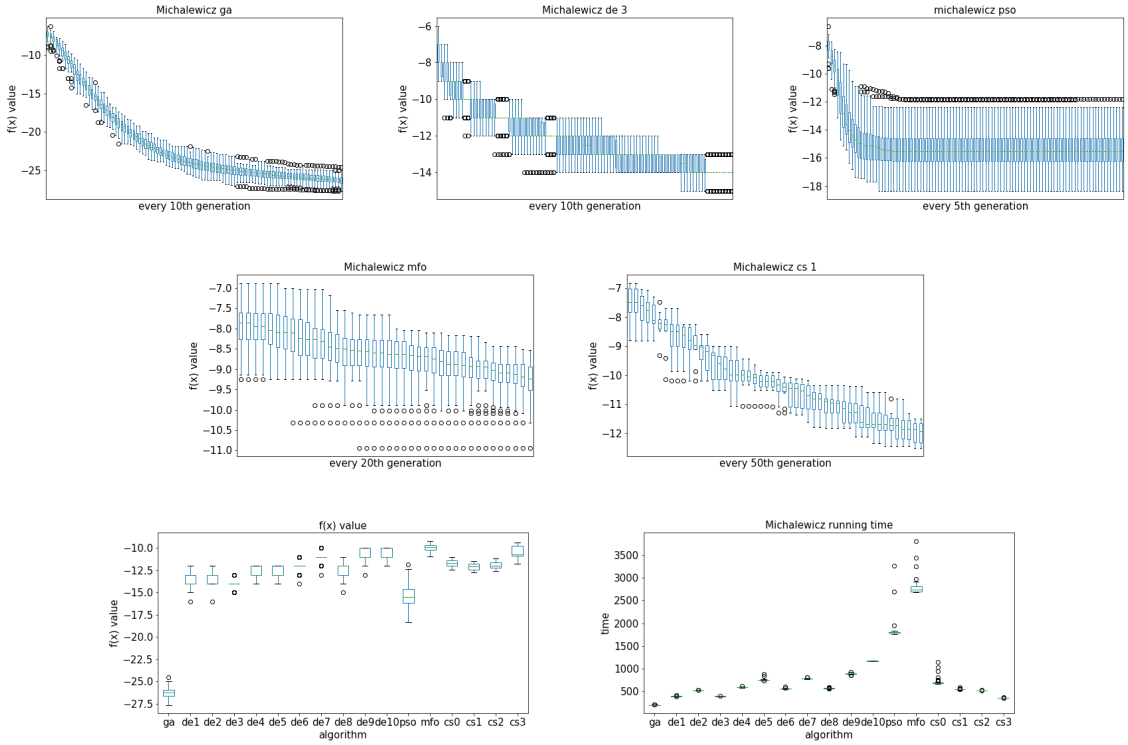

Figure 13: Results for Michalewicz function

5.14 Master's Consine Wave

Strategy	Mean	Median	Std	low	high	Time(mus)
GA	-11.9625	-11.967	1.9795	-7.56829	-18.3732	191.38
DE1	-4.08	-4	0.483322	-3	-5	393.66
DE2	-3.36	-3	0.557136	-3	-5	515.78
DE3	-4.2	-4	0.489898	-3	-6	388.26
DE4	-3.14	-3	0.490306	-2	-4	591.62
DE5	-3.06	-3	0.310483	-2	-4	753.34
DE6	-2.48	-2	0.574108	-2	-4	580.3
DE7	-2.14	-2	0.346987	-2	-3	784.76
DE8	-2.96	-3	0.397995	-2	-4	561.78
DE9	-2.06	-2	0.369324	-1	-3	918.34
DE10	-2.08	-2	0.271293	-2	-3	1178.4
PSO	-1.8779	-1.77067	0.410949	-2.94956	-1.27895	1988.1
MFO	-2.81039	-2.62756	0.616388	-5.09268	-2.15578	2679.4
CS	-2.84104	-2.65219	0.601934	-4.41615	-2.35834	644.5

Table 25: Results for Master's Consine Wave

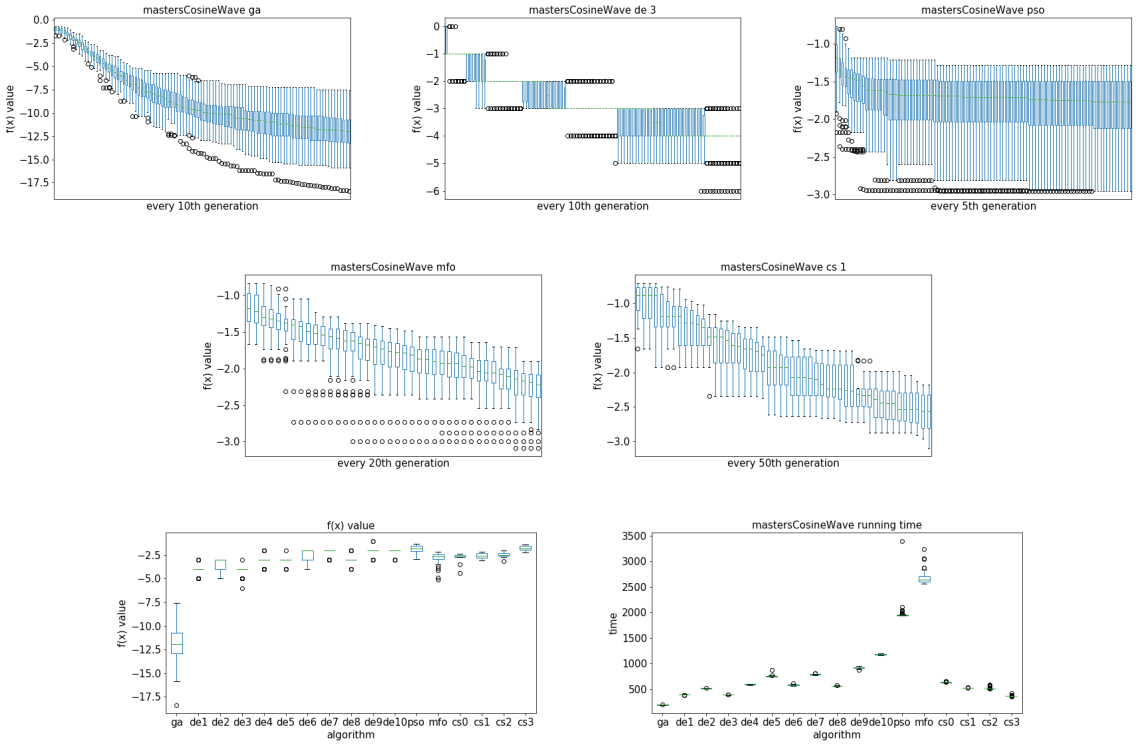

Figure 14: Results for Master's Cosine Wave function

5.15 Quartic

Strategy	Mean	Median	Std	low	high	Time(mus)
GA	4.20667e+07	3.35967e+07	2.87514e+07	2.97705e+06	1.14974e+08	147.34
GE1	0.06	0	0.237487	0	1	353.08
GE2	9.66941e+06	9.59357e+06	2.35018e+06	4.92095e+06	1.56536e+07	470
GE3	0	0	0	0	0	365.5
GE4	4.58521e+06	4.43637e+06	2.21819e+06	1.0841e+06	1.23345e+07	508.1
GE5	8.63252e+07	8.83244e+07	1.95273e+07	4.05153e+07	1.24361e+08	666.46
GE6	0.24	0	1.14123	0	8	512.08
GE7	2.56414e+07	2.43054e+07	7.11827e+06	8.90872e+06	3.93474e+07	710.44
GE8	0.06	0	0.237487	0	1	533.96
GE9	5.6424e+06	4.4291e+06	4.2819e+06	594595	2.0805e+07	768.96
GE10	2.45363e+08	2.48694e+08	4.61119e+07	1.19716e+08	3.87334e+08	1042.16
PSO	1.42329e+08	4.34527e+07	1.89165e+08	1784.66	9e+08	1619.82
MFO	1.12282e+08	9.41429e+07	6.48114e+07	2.50417e+07	3.25568e+08	2107.16
CS	4.81384e+07	4.34677e+07	2.10456e+07	2.04608e+07	8.46703e+07	449.9

Table 26: Results for Quartic

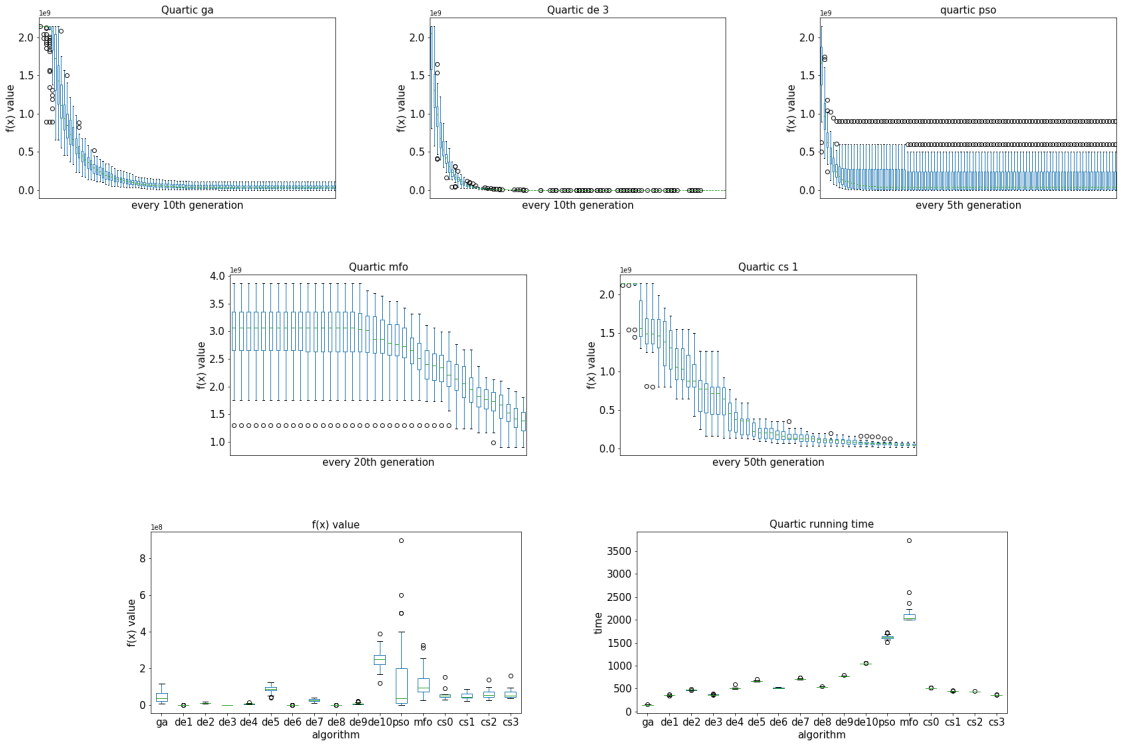

Figure 15: Results for Quartic function

5.16 Levy

Strategy	Mean	Median	Std	low	high	Time(mus)
GA	5.97079	4.92521	4.51866	1.44701	31.6831	213.04
DE1	3.7	3	4.30465	1	31	385.2
DE2	11.36	11.5	1.91583	6	15	508.52
DE3	0.44	0	0.535164	0	2	400.06
DE4	12.78	12	3.36624	7	20	549.36
DE5	32.52	32.5	4.39654	23	41	716.4
DE6	11.46	10	9.32354	2	58	544.94
DE7	20.24	20.5	2.55781	15	25	752.5
DE8	0.88	1	0.790949	0	4	567.9
DE9	22.86	22	7.24157	9	40	816.2
DE10	55.22	55.5	6.70609	43	71	1099.96
PSO	22.6544	22.7147	7.05471	9.72472	43.1741	2494.64
MFO	41.2083	38.5155	14.4798	18.9525	79.6705	2876.66
CS	52.4811	46.306	11.918	42.5047	81.3921	547.9

Table 27: Results for Levy

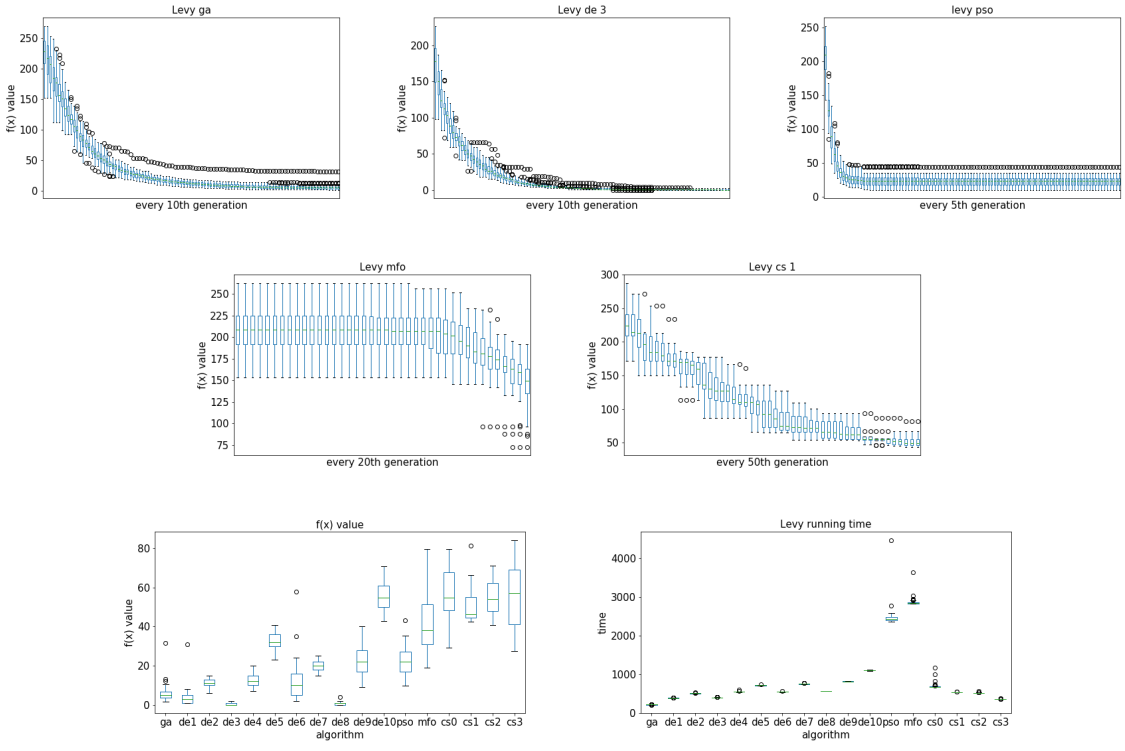

Figure 16: Results for Levy function

5.17 Step

Strategy	Mean	Median	Std	low	high	Time(mus)
GA	1893.74	1744.83	842.046	623.461	4666.41	120.94
DE1	9.14	9	0.490306	8	10	334.24
DE2	1588.68	1603	222.705	1110	2302	450.34
DE3	8.98	9	0.14	8	9	349.8
DE4	1024.32	1011	199.421	684	1572	491.28
DE5	7113	7145.5	975.698	5260	8964	641.8
DE6	10.02	10	1.14	8	15	496.7
DE7	2409.6	2381.5	301.718	1506	3039	688.18
DE8	8.38	8	0.485386	8	9	521.1
DE9	943.16	896	281.37	475	1647	748.22
DE10	13559.6	13703.5	1569.48	8770	16087	1008.32
PSO	1222.49	9.21536	3281.02	7.50411	10107.6	1251.5
MFO	5743.79	5510.27	1621.83	2525.78	9091.88	1686.48
CS	6161.41	5976.5	1255.92	4622.96	8164.82	407.5

Table 28: Results for Step

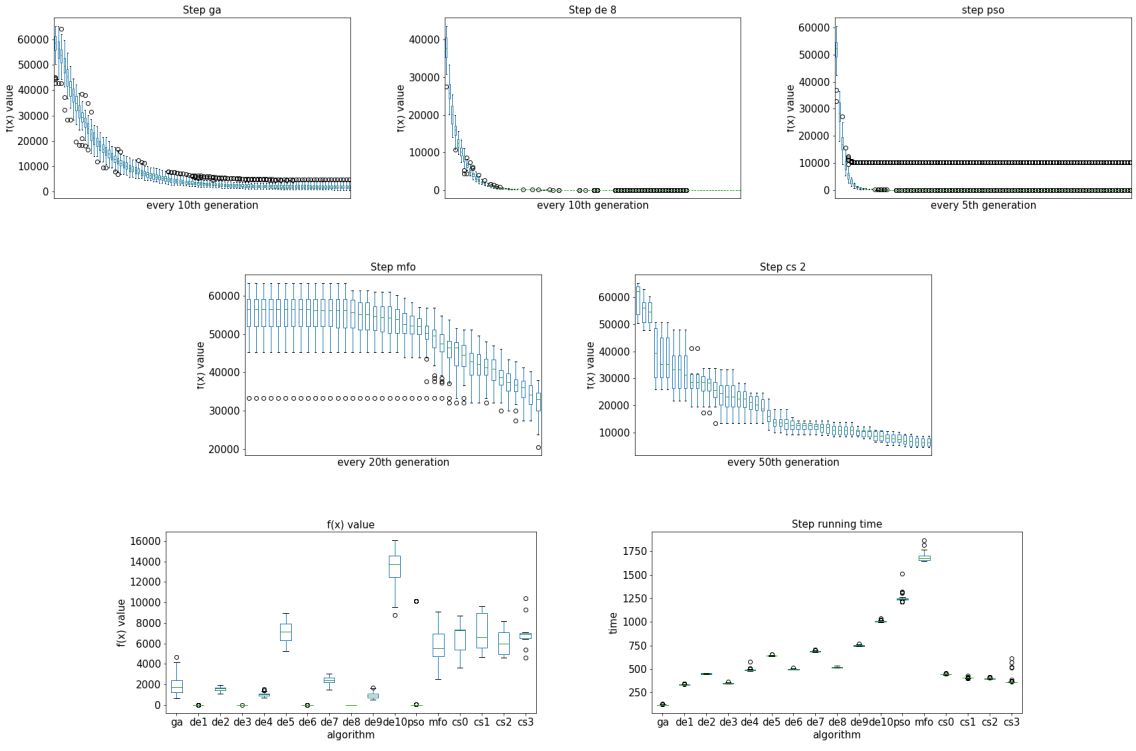

Figure 17: Results for Step function

5.18 Alpine

Strategy	Mean	Median	Std	low	high	Time(mus)
GA	27.1616	26.1614	5.29169	18.627	40.7951	131.9
DE1	80.46	77.5	29.5738	29	143	345.14
DE2	154.22	155	10.0405	130	172	465.1
DE3	44.76	46	10.5822	21	83	356.76
DE4	179.14	179.5	19.307	138	220	518.52
DE5	209.74	207.5	15.3712	158	236	677.12
DE6	136.44	124	56.1936	35	274	509.98
DE7	200.92	202.5	19.2716	140	238	710.52
DE8	30.58	20.5	26.8612	3	130	530.08
DE9	251.9	247	35.0184	193	356	793.42
DE10	295.82	299.5	20.6975	237	331	1057.64
PSO	47.2483	40.6376	48.2386	0.000912548	162.547	1545.42
MFO	297.308	299.744	32.2621	225.295	398.598	1858.2
CS	181.914	186.362	20.1549	155.861	215.467	450.2

Table 29: Results for Alpine

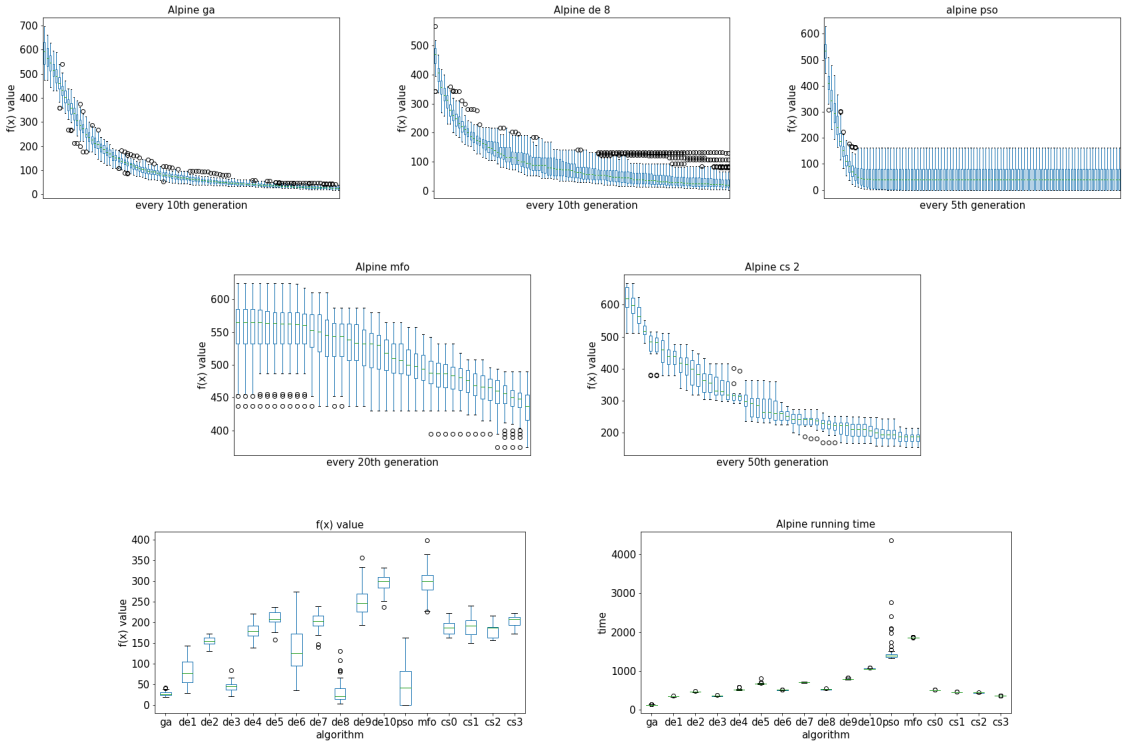

Figure 18: Results for Alpine function

6 Conclusion

6.1 GA and DE

Overall differential evolution algorithm performs better for most of the functions, while genetic algorithm has a better execution time. Most commonly seen better strategies for differential algorithms uses the best or rand-to-best for vector to be perturbed which produced better fitness, and choosing 1 difference vector for perturbation which both produced better fitness and lower execution time. Binomial crossover has a comparatively lower execution time compare to exponential crossover due to the time cost for random number generation.Execution time of differential evolution algorithm can be improved by implementing parallelism.

Function	Strategy(Low value Achieved)
Schwefel	DE/best/1/exp
1st De Jong	DE/rand-to-best/./., DE/best/1/bin
Rosenbrock	DE/rand-to-best/1/exp, DE/best/1/bin
Rastrigin	DE/best/1/bin
Griewangk	DE/best/1/., DE/rand-to-best/1/.
Sine Envelope Sine Wave	DE/././bin
Stretched V Sine Wave	DE/././exp
Ackley's One	DE/best/1/., DE/rand-to-best/1/bin
Ackley's Two	DE/best/1/., DE/rand-to-best/1/.
Egg Holder	GA, DE/best/1/., DE/rand-to-best/1/.
Rana	GA, DE/rand/./bin
Michalewicz	DE/rand/1/bin
Master's Consine Wave	DE/best/2/bin
Quartic	DE/best/1/., DE/rand-to-best/1/.
Levy	DE/rand-to-best/1/.
Step	DE/best/1/., DE/rand-to-best/1/.
Alpine	DE/rand-to-best/1/bin

Table 30: Differential Evolution algorithm Strategies

6.2 PSO and MFO

Both algorithms has a low cost for several benchmark functions. Running time of the function is a little high compare to algorithm due to random number generation. Velocity needs to be updated for every dimension for each particle. For each generation, flames are the best positions of the moth from last generation, diversity introduced would be comparatively low.

6.3 CS

Running time and cost of the function doesn't have significant difference compare to the basic cuckoo search and each other. Thus changing switching parameter p_a might not have a significant improvement upon the basic cuckoo search. Time for cuckoo search to reach a stable cost is longer compare to other algorithms done in this experiment. A possible reason is that for each generation, fitness improvement relies on one new solution accepted based on function cost, and the worst solution abandoned are substituted with global random solution. Trade-off between diversification and intensification can be improved for basic cuckoo search. Two possible improvement that can be tried are using Gaussian normal distribution and Gamma distribution instead of levy flight to determine the random walk size. In the basic cuckoo search, u and v chosen for step size calculation is drawn from $Norm(0, \sigma^2)$ and $Norm(0, 1)$ for Levy flight, in Gaussian distribution X_i is updated by drawing from $Gaussian(Best, \sigma)$, which could improve the intensification of the search. The other one is to use greedy search to improve the search ability.