Motor operation inspired optimization for Hierarchical Influence governed Differential Evolution

ABSTRACT

Operational maturity of biological control systems have fuelled the inspiration for a large number of mathematical and logical models for control, automation and optimisation. The human brain represents the most sophisticated control architecture known to us and is a central motivation for several research attempts across various domains. In the present work, we introduce an algorithm for mathematical optimisation that derives its intuition from the hierarchical and distributed operations of the human motor system. The system comprises global leaders, local leaders and an effector population that adapt dynamically to attain global optimisation via a feedback mechanism coupled with the structural hierarchy. The hierarchical system operation is distributed into local control for movement and global controllers that facilitate gross motion and decision making. We present our algorithm as a variant of the classical Differential Evolution algorithm, introducing a hierarchical crossover operation. The discussed approach is tested exhaustively on standard test functions from the CEC 2017 benchmark. Our algorithm significantly outperforms various standard algorithms as well as their popular variants as discussed in the results.

CCS CONCEPTS

•**Computer systems organization** → **Embedded systems;** *Redundancy;* Robotics; •**Networks** → Network reliability;

KEYWORDS

Hierarchical Optimization

ACM Reference format:

. 2017. Motor operation inspired optimization for Hierarchical Influence governed Differential Evolution. In *Proceedings of the Genetic and Evolutionary Computation Conference 2017, Berlin, Germany, July 15–19, 2017 (GECCO '17),* 6 pages.

DOI: 10.475/123_4

1 INTRODUCTION

Evolutionary algorithms are classified as meta-heuristic search algorithms, where possible solution elements span the n-dimensional search space to find the global optimum solution. Over the years, natural phenomena and biological processes have laid the foundation for several algorithms for control and optimization that have highlighted their applicability in solving intricate optimization problems in various fields.At the cellular level in the E.Coli Bacterium,there is sensing and locomotion involved in seeking nourishment and avoid harmful chemicals.These behavioral characterisitics have fueled the inspiration for the bacterial foraging optimization algorithm. Ant Colony Optimization(ACO) deals with behaviour of ants and has been a successful model for solving complex problems. PSO is a swarm intelligence algorithm based on behaviour of birds anf fishes that models these particles as they traverse an n-Dimensional search space and share information in order to obtain global optimum. From a biological control point, the human brain represents one of the most sophesticated architectures and several research attempts seek to mimic its accuracy, precision and efficiency.The brain function activities can be classified into 2 categories:- sensory and motor operations. Sensory cortical functions inspired concept of neural networks and they are being scaled successfully in deep learning to solve vast amount of problems. The human motor function represents a neural distributed and hierarchical control system. It can be classified as having local control functions for movement as well as higher level controllers for gross motion and decision making for planning actions.The optimal execution of motor operation involves distributed brain structures at different levels of hierarchy.These include the pre-frontal cortex, motor cortex, spinal cord, anterior horn cells etc. For generating an actions sequence, a sequence of actions is implemented by a string of subsequences of actions each implemented in a different part of the body. The operational structure has been depicted in Figure 1.[] For optimality of actions, neurons act in unison. The neurons in the motor cortex act like global leaders and send inhibitory or facilitatory influence over the anterior horn cells, local leaders, located in the spinal cord. These local leaders, are connected to the muscle fibres, effectors through a peripheral nerve and neuromuscular junction. Efficient execution of task requires feedback based facilitation and inhibition of the effectors over the anterior horn cells. These sequence of operations consitute the optimal convergence of the system leading to smooth motor execution.

The present paper focusses on introducing an optimisation algorithm modelled intuitively on the distributed and hierarchical operation of the brain motor function. We seek to exploit the ¡¿ inherent structure of the popular Differential Evolution algorithm by extrapolating the ¡Add a Line¿. The algorithm performance is exhaustively compared on the ¡Add here and connect to different algorithms¿.¡Why we have extrapolated DE¿.¡How introducing Hierarchy has ensured faster convergence and better optimization that other adaptive variants that have been discussed like JADE¿.

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim

GECCO '17, Berlin, Germany

DOI: 10.475/123_4

veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

2 CLASSICAL DIFFERENTIAL EVOLUTION

The classical Differential Evolution (DE) algorithm is a population-based global optimization algorithm, utilizing a crossover and mmutation approach to generate new individuals in the population for achieving optimum solutions. For each individual x_i that belongs to the population, DE randomly samples three other individuals from the population a_i, b_i and c_i. Then using the randomly chosen points, a new individual vector is generated using equation (1):

$$u_i = a_i + F(b_i - c_i) \tag{1}$$

Where, F is called the differential weight (Usually lies between $[0, 1]$).
And to obtain the new position of the individual, a crossover operation is implemented between x_i and u_i, controlled by the parameter CR called the crossover probability. The value for CR always lies between $[0, 1]$.

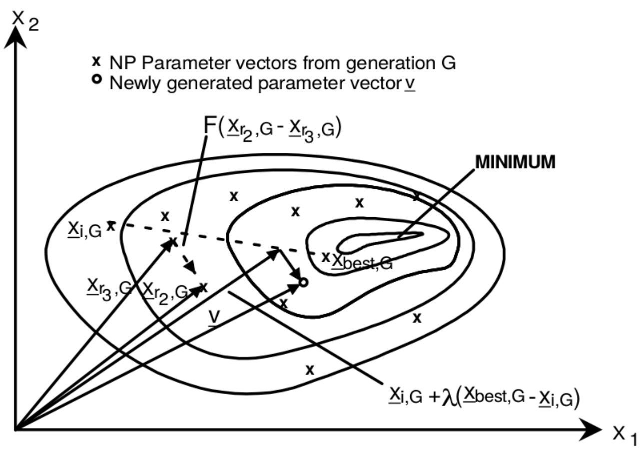


Figure 1: Two dimensional example of an objective function showing its contour lines and the process for generating v in scheme DE2

3 DISTRIBUTED LEADER OPTIMIZATION

Taking inspiration from the human motor system, we model the hierarchical motor operations in our optimization agents, where we define a global leader which influences the action of several distributed local leaders and the particle agents which act as the effectors. The global leader is analogous to the decision making and planning section in the motor system hierarchy whilst, the local leaders correspond to motion generators acting under the influence of the global leader.

The position of each particle in the population is affected by the influence of global leaders and local leaders, while also being affected by a randomly chosen particle from the population to

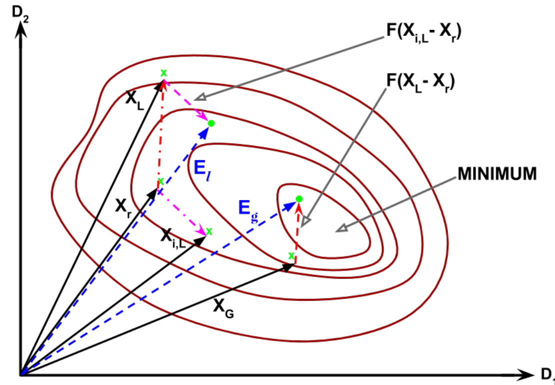


Figure 2: The process for generating generation of E_g and E_l in a 2 dimensional optimization.

induce some stochasticity in the optimization pipeline. We first model the influence of the global leader on the local leaders and the influences of the local leaders on each population element using equation (2) and (3). We introduce a hierarchical crossover between the two influencing equations governed by a hierarchical crossover parameter HC.

Analogously to [step 3] in the brain motor operation, the updation of particle positions requires generating feedback for the leaders as a part of the optimization procedure, and hence the local leaders and the global leader are updated based onn their objective function value generated from the perturbations in population particles. This series of events comprise of one optimization pass (one loop step). On execution of several optimization passes as described, the system is able to converge to an optimal configuration, analogous to the successful execution of the required task as shown in [step 4].

The updated position of the particle x is governed by the hierarchical crossover operation and a mutation operation. The hierarchical operation is affected by the global leader g_L and the local leader l through the parametric equations (2) and (3). Switching between the two is governed by the hierarchical crossover parameter HC. The given equations are discussed as follows:

$$E_g = g_L + F(l - c) \tag{2}$$

$$E_l = l + F(x - c) \tag{3}$$

In the algorithm 1, The Hierarchical crossover is controlled by the conditional equation $i_c < (HC * i_N)$. According to this equation, during the initial phases (HC fraction of total generations) of the optimization procedure, only the local leader is responsible for the motion of the agents, and after a certain amount of time has passed, the global leader also takes part in the motion generation process, signifying the motor control operation. Additionaly, The hierarchical crossover parameter HC also influences the mutation

Algorithm 1 Distributed Leader Optimization

procedure START
  Initialize parameters (HC, P, N_l, N).
  Generate initial global leader g_L as a random point.
  Generate N_l local leader points around g_L.
  Using a Normal distribution, generate N points for population P around the local leaders.
  while termination criteria is not met **do**
    for each individual x_i in P **do**
      compute the corresponding local leader l based on nearest position.
      Let $u = 0$ be an empty vector.
      Let i_c and i_N be the current generation and total generations of the procedure.
      if $i_c < (HC * i_N)$ **then**
        $u = E_g$ from (2).
      else
        $u = E_l$ from (3).
      end if
      for each dimension i **do**
        Generate $r_i = U(0, 1)$, a random number between 0 and 1.
        if $r_i < HC$ **then**
          Set $x_i^{'} = x_i$.
        else
          Set $x_i^{'} = u_i$
        end if
      end for
      if $f(x^{'}) < f(x)$ **then**
        Replace x with $x^{'}$ in the population.
      end if
    end for
    Alter local leaders in each population cluster based on objective function value.
    Compute updated global leader g_L.
  end while
end procedure

process wherein the degree of final mutation is decided based on the probability HC.

4 RESULTS AND DISCUSSIONS

All evaluations were performed using Python 2.7.12 with Scipy and Numpy for numerical computations and Matplotlib package for graphical representation of the result data. This section is divided into two sub-sections: Section A provides description about the problem set used for analysis of algorithmic efficiency and accuracy, and section B comprises of tabular and graphical data to support the claim of eminence of the proposed approach.

4.1 Problem Set Description

The set of objective functions considered for testing the proposed algorithm and compare it's performance against DE, Particle Swarm Optimization Differential Evolution (PSODE) and Joint Adaptive Differential Evolution (JADE) has been taken from the CEC 2017 [] set of benchmark functions.

Table 1: Test Functions

F_{id}	Problem Function
1	Shifted and Rotated Bent Cigar Function
2	Shifted and Rotated Sum of Different Power Function
3	Shifted and Rotated Zakharov Function
4	Shifted and Rotated Rosenbrockfis Function
5	Shifted and Rotated Rastriginfis Function
6	Shifted and Rotated Expanded Scafferfis F6 Function
7	Shifted and Rotated Lunacek Bi_Rastrigin Function
8	Shifted and Rotated Non-Continuous Rastriginfis Function
9	Shifted and Rotated Levy Function
10	Shifted and Rotated Schwefelfis Function
11	Hybrid Function 1 (N=3)
12	Hybrid Function 2 (N=3)
13	Hybrid Function 3 (N=3)
14	Hybrid Function 4 (N=4)
15	Hybrid Function 5 (N=4)
16	Hybrid Function 6 (N=4)
17	Hybrid Function 7 (N=5)
18	Hybrid Function 8 (N=5)
19	Hybrid Function 9 (N=5)
20	Hybrid Function 10 (N=6)
21	Composition Function 1 (N=3)
22	Composition Function 2 (N=3)
23	Composition Function 3 (N=4)
24	Composition Function 4 (N=4)
25	Composition Function 5 (N=5)
26	Composition Function 6 (N=5)
27	Composition Function 7 (N=6)
28	Composition Function 8 (N=6)
29	Composition Function 9 (N=3)
30	Composition Function 10 (N=3)
Search Range: $[-100,100]^D$	

4.2 Results

5 CONCLUSION

Differential Evolution is one of the most popular and widely used evolutionary meta-heuristic for the task of optimization. In this work, we have proposed a new variant of the same called "Hierarchical Motor Differential Evolution", inspired from the hierarchical structure of the brain motor function. This approach enables the population to follow two distince motion patterns, one governed by their local leaders and one by the global leader. Based on these two influences, the individuals try to achieve the global optimum, and have shown to outperform the algorithms taken under consideration by a appreciable factor, as is clearly depicted through the

Table 2: Objective Function Value for Dimension: 10

ID	DE		JADE		PSO-DE		Ours	
	best	mean	best	mean	best	mean	best	mean
1	100.000051	100.011085	100.0	100.0	100.000712	185.975885	100.0	100.0
2	200.0	200.1	200.0	200.0	200.0	200.0	200.0	200.0
3	300.00134	300.214502	300.0	300.0	300.000006	300.000985	300.0	300.0
4	400.042617	403.674837	400.0	400.409399	400.064644	404.307763	400.0	400.000003
5	566.661791	604.867489	523.908977	541.521084	525.868824	575.61616	533.803201	539.483815
6	621.914237	634.807962	620.878276	636.034759	603.187964	635.865001	613.730565	629.293758
7	724.831278	739.129935	717.016542	723.983312	725.44788	733.15638	720.345706	725.233785
8	818.904202	829.749207	821.914433	826.321588	820.8941	830.246691	821.064763	823.160987
9	900.0	908.104383	900.0	1084.478253	900.0	1124.102561	900.0	903.454324
10	1911.510092	2447.443751	1760.956867	2162.648588	2049.644727	2518.241095	1694.437597	2049.074266
11	1102.985708	1113.423105	1105.661676	1117.509748	1105.97013	1120.192974	1101.769749	1108.863598
12	2531.746305	6509.743078	1438.605713	5430.674683	4089.006352	10810.387667	1308.438341	1327.405881
13	1313.130226	1404.903601	1304.681558	1328.755262	1319.839199	1453.340785	1302.682039	1304.282241
14	1409.949612	1426.571937	1412.934432	1428.169439	1420.91065	1434.112884	1404.928993	1410.000769
15	1504.131392	1521.446614	1502.496189	1508.31154	1501.389515	1518.310358	1500.08137	1503.169264
16	1958.42062	2104.555728	1958.857997	2094.630816	1958.411527	2048.156879	1958.433511	2012.385949
17	1728.194973	1743.155244	1730.715318	1748.129878	1727.80039	1791.607742	1723.853972	1746.589077
18	1801.586012	1838.840555	1804.298538	1825.091639	1817.154641	1840.546923	1800.235516	1804.014301
19	1901.195482	1903.604767	1900.399786	1902.152965	1902.71174	1906.252333	1900.005632	1901.014116
20	2204.55412	2289.226577	2148.538938	2178.313173	2140.561308	2261.038768	2135.915527	2152.816519
21	2337.772994	2387.230357	2314.421135	2338.688719	2337.207339	2351.898856	2320.496212	2334.61612
22	2300.805852	2304.132879	2300.0	2300.093485	2300.684181	2301.710478	2300.000015	2301.095975
23	3070.177083	3145.772296	3003.678563	3091.22041	2773.372859	3060.022519	2657.020036	2851.982305
24	2500.0	2500.0	2500.0	2500.0	2500.0	2500.0	2500.0	2500.0
25	2899.584968	2933.249812	2899.584968	2930.266506	2897.742869	2921.27479	2897.833388	2919.976511
26	2800.0	4117.597033	2800.0	2956.064173	2800.0	3367.60765	2800.0	3161.548079
27	3113.157656	3358.806434	3072.439023	3178.509645	3078.873134	3240.501812	3071.203569	3107.268539
28	3184.75565	3230.921422	3184.75565	3195.113042	3184.755652	3198.370691	3100.0	3195.411961
29	3148.587115	3266.979786	3172.400194	3233.707677	3191.348193	3244.892638	3189.211417	3292.420474
30	3442.555095	11927.404685	3207.766942	4615.591316	4573.358512	16415.162901	3205.740954	3249.710975

Table 3: Objective Function Value for Dimension: 30

ID	DE		JADE		PSO-DE		Ours	
	best	mean	best	mean	best	mean	best	mean
1	100.001508	4334.438478	100.001338	100.056201	364.295574	4236.363207	100.0	100.0
2	40412441.0	5.129601e+19	200.0	1535352368.6	332899.0	9.590679e+11	200.0	159855.5
3	17926.872873	22131.542719	69304.926091	74080.700372	15792.547575	21683.209092	3679.811599	8999.947269
4	481.255055	519.422652	403.633939	442.206911	468.341175	479.341966	400.004163	441.016156
5	689.041352	737.79326	667.50756	735.204027	715.904429	746.548906	617.40454	688.842184
6	643.626307	652.582714	651.39169	655.142819	642.724237	655.106996	644.701241	652.002395
7	883.347367	962.591129	779.907693	818.344111	790.014281	854.285524	812.923573	856.90477
8	923.37426	967.251501	931.500175	957.362003	915.414882	960.486239	930.288539	964.11663
9	5652.483961	7878.781444	4953.05469	5146.600953	6018.417197	9042.410178	4003.118072	4734.984364
10	3596.63104	4536.989761	4012.723292	4204.18969	3934.606704	4863.741107	3793.781776	4346.741344
11	1162.405965	1184.634006	1152.748529	1174.58813	1165.144993	1189.171787	1149.748499	1171.130409
12	56679.435092	317650.61349	24821.171765	58930.090242	10221.077465	161046.055405	9208.289246	41947.222696
13	3002.029489	18794.835991	4276.907742	13775.816239	3871.279833	10612.26359	1664.06241	2453.606969
14	1773.180798	5502.160382	1496.219858	42868.9158	1555.452763	4029.808535	1462.926848	1504.191515
15	1860.435669	2484.689969	1688.05046	2222.674323	1651.747476	2223.060542	1611.074402	1852.66177
16	2517.439623	2827.004968	2344.19818	2621.618684	2239.242719	2664.114667	2298.041965	2606.674809
17	2321.175936	2604.529778	2062.898023	2546.995596	2107.43677	2457.34021	1820.806639	2418.723829
18	38987.282456	94156.328505	11841.60813	184888.162181	62294.853257	118430.289122	12578.003784	23024.111937
19	2043.469888	3010.235379	1959.71819	2156.957875	3049.52231	6840.408394	1949.271714	1987.866761
20	2625.539158	2864.832611	2706.314441	2805.600064	2619.996493	2895.107238	2753.806213	2966.035793
21	2412.081757	2504.777775	2414.52134	2456.718982	2431.740293	2478.841357	2200.0	2442.734316
22	2300.481796	5655.569322	2300.0	4157.698784	2307.721358	6811.069162	2300.009985	6795.24842
23	3050.654508	3572.965066	2772.002023	2946.749322	2764.922461	3199.874364	2883.276891	3543.839343
24	3104.623692	3290.698756	2891.557648	2965.225566	2911.63347	2983.772932	2500.0	2940.75997
25	2916.180657	2946.711753	2875.106846	2881.091389	2875.498843	2889.943671	2874.171109	2877.484904
26	4043.691403	6756.3724	2900.0	3266.510982	2800.007809	3273.128769	2900.0	3298.490539
27	3200.005857	3998.876498	3145.810354	3189.82261	3145.425231	3639.634132	3132.816283	3284.28897
28	3290.744025	3326.263983	3100.0	3131.027315	3195.486838	3225.594053	3100.0	3115.505829
29	3720.314598	4115.185803	3305.310139	3626.887552	3535.952295	3867.593068	3352.845055	3709.102375
30	3359.030768	3900.826662	3263.496536	3749.610722	3312.635025	3524.714477	3298.704645	3421.715322

numerical results and performance plots. however, since the proposed algorithm fizzles on a small fraction of the objective functions, we shall continue our quest to improve it's performance through continous modifications through our future work, and analyse it's performance on several real-world applications. [1]

Table 4: Objective Function Value for Dimension: 50

ID	DE		JADE		PSO-DE		Ours	
	best	mean	best	mean	best	mean	best	mean
1	5884574.87314	367294248.521	136.072384	3708.75086	5811.218992	154233.646744	106.072862	3665.419272
2	4.718137e+24	3.364977e+44	2635725.0	5.023747e+26	2.212101e+19	2.544543e+23	2.279950e+17	1.00729e+31
3	45520.966376	62237.296819	143481.793147	156166.762356	52308.42743	64435.24063	44613.299932	58182.83733
4	574.400328	801.384952	418.580378	470.113207	477.080964	574.528479	400.005049	447.775413
5	816.394775	843.258843	809.899483	834.131266	778.59312	831.066954	791.405194	830.218472
6	652.541914	655.794152	633.217881	654.893828	653.291336	658.183613	645.25633	656.060597
7	1109.02123	1263.038487	889.036574	944.90319	915.153525	1047.43879	989.957862	1186.248741
8	1139.278925	1175.893113	1118.339103	1144.604745	1092.62639	1159.032351	1100.476077	1168.529946
9	22196.387817	29218.775982	11958.280061	13174.662361	24753.040541	32233.95451	10251.476381	24752.7168
10	6228.49289	7289.183679	6054.707691	6833.306317	6207.795302	7055.595231	6050.434374	6609.804567
11	1170.858603	1258.517635	1202.694857	1232.204268	1206.154564	1252.939541	1156.439606	1205.254497
12	677263.079928	16987989.9819	74784.615963	530814.648184	584300.698313	3448448.79067	126908.215793	494471.075675
13	6005.535308	16893.949921	2041.488125	4332.5945	1572.252973	4301.829606	1484.761799	7760.056137
14	38490.532315	174367.45065	2466.047056	238838.470051	16327.42317	67939.000264	2967.818485	26290.316181
15	2278.141229	26989.255509	13553.041864	25636.769611	3443.587343	9167.267098	1938.200405	14976.72189
16	2722.026011	3176.916902	2345.400708	2916.561016	2521.93881	3146.04527	2436.449338	2978.37746
17	2799.949776	3289.61565	2568.383575	2907.869272	2887.281107	3236.957928	2591.370306	2874.965038
18	264037.125702	872072.477397	36176.586779	113941.317657	156965.285126	114846.121366	260540.781819	536454.326476
19	10051.912407	20380.25713	2089.172253	7763.17234	9905.850822	16555.756926	2013.126904	3609.258962
20	2950.923195	3274.334015	3041.81309	3113.289461	2991.589293	3361.823946	2495.031774	3080.137478
21	2596.725663	2689.688363	2526.190898	2597.677199	2555.8788	2642.381597	2447.758274	2570.911014
22	9713.993241	10803.653732	10759.59674	11032.880953	8918.436264	10465.022457	8181.446081	9755.070369
23	3451.104943	4200.174424	2971.160647	3237.778662	2977.554961	3490.639751	2851.650254	3162.313622
24	3434.465028	3682.846708	3103.955173	3185.382676	3036.799607	3158.330504	3136.927747	3284.656095
25	3141.144886	3292.303449	2931.162959	2962.471758	2931.926959	3008.895353	2931.182314	2954.767839
26	4906.132848	7989.490966	2900.0	3346.874039	2900.441895	3653.757741	2900.0	3262.668498
27	3200.010703	3792.645588	3143.038057	3184.646353	3158.178238	3397.130323	3200.010872	3200.011524
28	3300.010827	3431.570911	3240.725865	3288.253039	3263.207144	3300.257609	3243.631996	3294.373237
29	3812.475517	4605.349537	3533.945743	3956.835243	3955.324537	4364.18129	3653.675553	3966.471956
30	3673.711968	5813.173755	3916.725719	4869.089335	3730.309354	5143.078706	3346.483679	4747.88675

Table 5: Objective Function Value for Dimension: 100

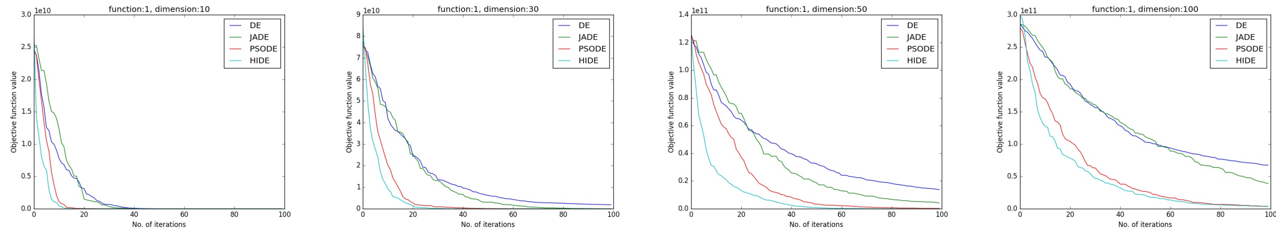
ID	DE		JADE		PSO-DE		Ours	
	best	mean	best	mean	best	mean	best	mean
1	3427212811.79	13807281895.7	141.263356	13516.698933	6067123.52108	29751976.5091	122.398748	11708.823609
2	4.196171e+84	1.547414e+112	8.737524e+74	2.543621e+87	6.153667e+66	3.211842e+73	3.883505e+80	8.891481e+114
3	208808.969094	242699.687639	312244.360944	332179.290693	241427.723667	257462.977885	220765.08386	261901.109331
4	1975.651157	2752.246068	539.386275	677.054657	777.314462	836.965399	531.169819	621.219143
5	1223.536503	1286.153332	1249.195036	1307.110127	1248.410134	1310.887657	1068.11742	1272.47682
6	651.650133	657.84974	654.709342	659.421427	656.877048	662.318417	642.33355	654.132758
7	1614.003864	1920.797726	1367.066537	1536.357878	1311.849757	1534.207764	1562.379772	2076.702502
8	1595.418732	1736.367379	1672.567849	1768.082435	1678.127263	1761.94051	1293.552115	1592.162983
9	59726.514621	71986.043905	28906.90908	30336.745335	63640.331351	74961.220998	23466.575012	27067.029593
10	12005.889721	14725.348334	14227.801909	15355.621891	12937.027857	14972.950738	11153.58683	13298.092101
11	7540.617987	11481.260145	40447.548688	57228.683666	3521.901521	4544.804011	5380.432052	9916.347692
12	529993877.325	1881773956.29	2893556.27222	6415173.6097	26105108.937	41876679.0862	3680108.18151	10059039.6342
13	7943.9249	508209.562668	4622.698553	8892.775994	8246.515295	12675.845535	2976.841354	11376.986338
14	728122.833253	1329183.17224	132194.795253	365560.88163	548410.338286	941547.524763	234045.940166	867160.306892
15	2660.465784	181957.060133	1799.506503	3362.509604	1899.073444	2914.44348	1976.789124	4485.415275
16	4749.254663	5847.826738	4817.483738	5632.3022	3852.700054	5228.663526	3519.494945	4796.802728
17	4397.496352	4958.418182	3842.206015	4450.177422	3790.72056	4730.994585	3582.785882	5463.216947
18	1357845.39305	1938893.27972	146426.273603	763318.822618	1004224.20385	2315010.29868	631040.14635	1335739.59138
19	2482.170159	26455.706954	2098.9496	4767.529535	2263.725158	3927.459947	2071.077067	3664.159878
20	4968.497438	5436.604051	5231.026486	5690.748998	5109.460563	5781.300835	3627.777893	5228.430669
21	3180.746656	3355.4783	2921.900122	3085.692252	2885.574085	3127.356835	2926.350399	3199.986183
22	17808.897744	19562.986646	19213.375668	20278.929093	18695.522312	20167.413741	17548.339053	19597.151245
23	4907.519646	5819.207866	3352.556985	4222.436894	3582.043556	4779.921248	3418.983204	3609.098575
24	5173.249408	5946.12042	4060.951302	4095.429519	3801.368588	4042.426859	3998.054028	4216.824895
25	4089.118918	4548.285768	3153.485413	3236.61784	3348.382262	3407.526581	3176.3038	3264.318532
26	8557.498566	20159.11458	2900.077371	11924.799473	3021.136025	4682.035439	2900.000382	9867.5518
27	3200.023355	3772.409153	3194.809213	3201.670732	3200.024171	3494.618132	3200.023542	3200.023953
28	4947.745152	5948.213156	3295.122914	3340.280383	3456.828432	3542.571307	3300.807691	3354.717338
29	6004.774424	7090.642544	5208.711727	5970.628689	5462.328635	6178.559061	4541.195471	5739.291549
30	7798.106217	202435555.594	3584.974771	10674.217331	3920.327039	7139.460728	3850.317099	15318.554601

REFERENCES

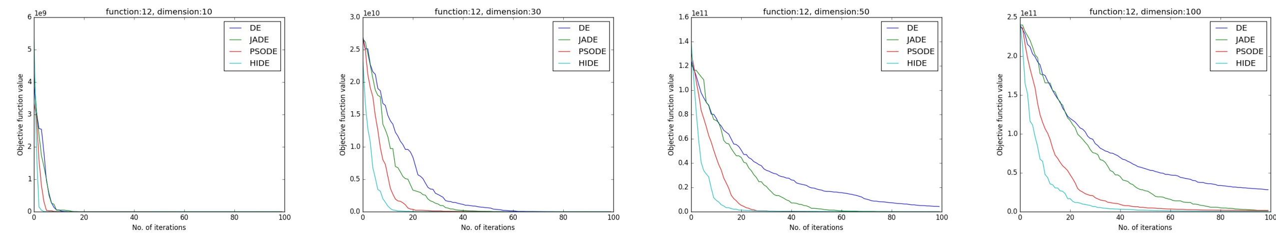
[1] Zhimin Cao, Qi Yin, Xiaoou Tang, and Jian Sun. 2010. Face recognition with learning-based descriptor. In *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*. IEEE, 2707–2714.

REFERENCES

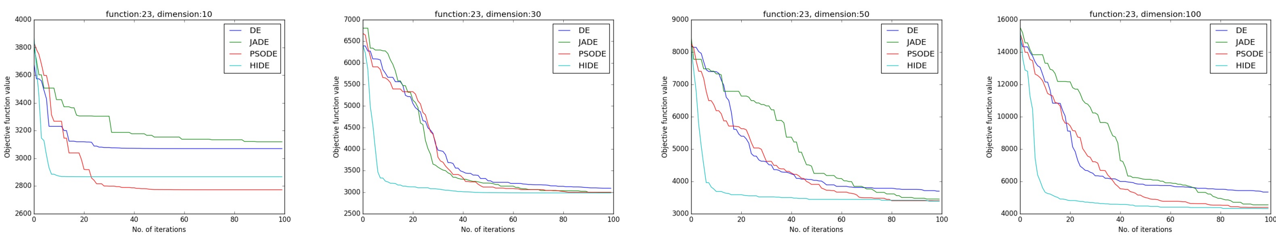
[1] Zhimin Cao, Qi Yin, Xiaoou Tang, and Jian Sun. 2010. Face recognition with learning-based descriptor. In *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*. IEEE, 2707–2714.



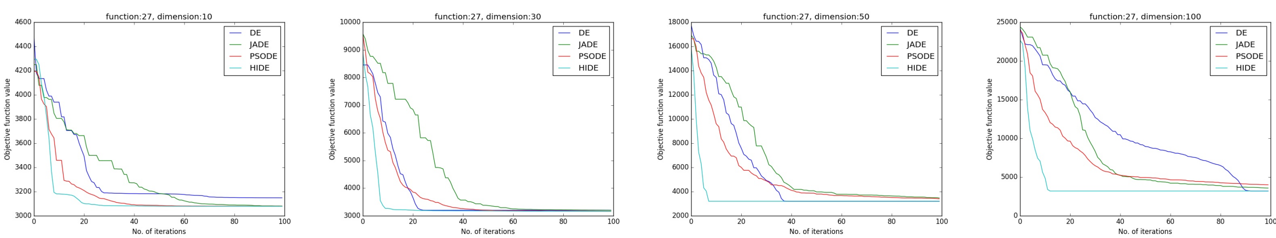
(a) F1D10 (b) F1D30 (c) F1D10 (d) F1D30



(e) F12D10 (f) F12D30 (g) F12D50 (h) F12D100



(i) F23D10 (j) F23D30 (k) F23D50 (l) F23D100



(m) F27D10 (n) F27D30 (o) F27D50 (p) F27D100

Figure 3: Comparision analysis over various functions and dimensions