A New Adaptive and Dynamic Strategy in Cat Swarm Optimization

Maysam Orouskhani, Yasin Orouskhani, and Mohammad Teshnehlab

Abstract—**Cat Swarm Optimization (CSO) is one of the new swarm intelligence algorithms for finding the best global solution. Because of complexity, sometimes the pure CSO takes a long time to converge and cannot achieve the accurate solution. For solving this problem and improving the convergence accuracy level, we propose a new improved CSO namely 'Adaptive Dynamic Cat Swarm Optimization'. First, a new adaptive inertia weight is added to velocity equation and then an adaptive acceleration coefficient is used.**

Second, by using the information of two previous/next steps and applying a new factor, we reach to a new position update equation composing the average of position and velocity information. Experimental results for six test functions show that in comparison with the pure CSO, the proposed CSO can takes a less time to converge and can find the best solution in less iteration.

I. Introduction

Function Optimization is one of the important fields in the computational intelligence theories. There are many algorithms to find the global and local solutions of the problems. Some of these optimization algorithms were developed based on swarm intelligence. These algorithms imitate the creature's swarm behavior and model into algorithm, such as Ant Colony Optimization (ACO) which imitates the behavior of ants [1]-[6], Particle Swarm Optimization (PSO) which imitates the behavior of birds [2], Bee Colony Optimization (BCO) which imitates the behavior of bees [3] and the recent finding, Cat Swarm Optimization (CSO) which imitates the behavior of cats [4]. By simulating the behavior of cats and modeling into two modes, CSO can solve the optimization problems.

In the cases of functions optimization, CSO is one of the best algorithms to find the global solution. In comparison with other heuristic algorithms such as PSO and PSO with weighting factor [7], CSO usually achieves better result. But, because of algorithm complexity, solving the problems and finding the optimal solution may take a long process time and sometimes much iteration is needed.

So in this article, in order to achieve the high convergence accuracy in less iteration, an improved CSO is proposed.

First, an adaptive inertia weight and acceleration coefficient is used. So, the new velocity update equation will be computed in an adaptive formula. Then, our aim is to consider the effect of previous/next steps in order to calculate the current position. So by using a factor namely 'Forgetting Factor', the information values of steps will be different. Finally, an average form of position update equation composing new velocity and position information will be used.

Experimental results indicate that the proposed algorithm rather than pure CSO can improve performance on finding the best global solution and achieves better accuracy level of convergence in less iteration

Maysam Orouskhani, Department of Computer Engineering, Science and Research Branch, Islamic Azad University, Tehran, Iran (Corresponding author, email:orouskhani@ce.sharif.edu)

Yasin Orouskhani, BSc Student of Computer Engineering, Sharif University of Technology, Tehran, Iran.

Mohammad Teshnehlab, Department of Computer Engineering, Science and Research Branch, Islamic Azad University, Tehran, Iran

II. Cat Swarm Optimization

Cat Swarm Optimization is a new optimization algorithm in the field of swarm intelligence [4]. The CSO algorithm models the behavior of cats into two modes: 'Seeking mode' and 'Tracing mode'. Swarm is made of initial population composed of particles to search in the solution space. For example, by simulating the birds, ants and bees, Particle swarm optimization, Ant colony optimization and Bee colony optimization is created respectively. In CSO, cats are used for solving the problems.

In CSO, every cat has its own position composed of D dimensions, which is the size of decision variables, velocities for each dimension, a fitness value, which represents the accommodation of the cat to the fitness function, and a flag to identify whether the cat is in seeking mode or tracing mode. The final solution would be the best position of one of the cats. The CSO keeps the best solution until it reaches the end of the iterations [5].

Cat Swarm Optimization algorithm has two modes in order to solve the problems which are described below:

A. Seeking Mode

For modeling the behavior of cats in resting time and being-alert, we use the seeking mode. This mode is a time for thinking and deciding about next move.

This mode has four main parameters which are mentioned as follow: seeking memory pool (SMP), seeking range of the selected dimension (SRD), counts of dimension to change (CDC) and self-position consideration (SPC) [4].

The process of seeking mode is describes as follow:

Step1: Make j copies of the present position of cat_k, where j = SMP. If the value of SPC is true, let j = (SMP-1), then retain the present position as one of the candidates.

Step2: For each copy, according to CDC, randomly plus or minus SRD percent the present values and replace the old ones.

Step3: Calculate the fitness values (FS) of all candidate points.

Step4: If all FS are not exactly equal, calculate the selecting probability of each candidate point by equation (1), otherwise set all the selecting probability of each candidate point be 1.

Step5: Randomly pick the point to move to from the candidate points, and replace the position of cat_k.

$$P_i = \frac{|SSE_i - SSE_b|}{SSE_{max} - SSE_{min}} \tag{1}$$

If the goal of the fitness function is to find the minimum solution, $FS_b = FS_{max}$, otherwise $FS_b = FS_{min}$ [4].

B. *Tracing Mode*

Tracing mode is the second mode of algorithm. In this mode, cats desire to trace targets and foods. The process of tracing mode can be described as follow:
Step1: Update the velocities for every dimension according to equation (2).

Step2: Check if the velocities are in the range of maximum velocity. In case the new velocity is over-range, it is set equal to the limit.

$$V_{k,d} = V_{k,d} + r_1 c_1 \left(X_{best,d} - X_{k,d}\right) \tag{2}$$

Step 3: Update the position of cat $_k$ according to equation (3).

$$X_{k,d} = X_{k,d} + V_{k,d} \tag{3}$$

$X_{best,d}$ is the position of the cat, who has the best fitness value, $X_{k,d}$ is the position of cat_k , c_1 is an acceleration coefficient for extending the velocity of the cat to move in the solution space and usually is equal to 2.05 and r_1 is a random value uniformly generated in the range of [0,1]. The position equation is similar to the position equation of PSO, but in velocity equation, PSO is including of personal and global terms while CSO only has a global term.

C. *Core Description of CSO*

In order to combine the two modes into the algorithm, a mixture ratio (MR) which indicates the rate of mixing of seeking mode and tracing mode is defined. This parameter decides how many cats will be moved into seeking mode process. For example, if the population size is 50 and the MR parameter is equal to 0.7, there should be 50×0.7=35 cats move to seeking mode and 15 remaining cats move to tracing mode in this iteration [9].

We summarized the CSO algorithm below:

First of all, N cats are created and positions, velocities and the flags for each cat are initialized. In second step, according to the fitness function, evaluate the fitness value of the each cat and keep the best cat into memory (X_{best}). In next step, according to cat's flag, apply cat to the seeking mode or tracing mode process. After finishing the related process, re-pick the number of cats and set them into seeking mode or tracing mode according to MR parameter. At the end, check the termination condition, if satisfied, terminate the program, and otherwise go to second step [9].

III. ADAPTIVE DYNAMIC CAT SWARM OPTIMIZATION

The tracing mode of CSO has two equations: velocity update equation and position update equation. For reaching to an adaptive CSO, some parameters will be changed in velocity equation. Also for computing the current position of cats, the information of previous and next steps is considered by using a special factor. The proposed algorithm is explained in two parts: A and B.

A. *Adaptive Parameters*

In pure CSO, the velocity of each cat has two constants: r_1 and c_1 which are told before.

In the proposed algorithm, an adaptive inertia weight is added to the velocity equation which is updated in each dimension. By using this parameter, the global and local search ability will be balanced. A large inertia weight facilitates a global search while a small inertia weigh facilitates a local search. First we use a large value and it will be reduced gradually to the least value by using equation (4).

$$W(d) = W_s + \frac{(d_{max} - d)}{2 \times d_{max}} \tag{4}$$

Equation (4) indicates that the inertia weight will be updated adaptively, where W_s is the starting weight, d_{max} is the maximum dimension of benchmark and d is the current dimension. So the maximum inertia weight happens in the first dimension of the each iteration and it will be updated decreasingly in every dimension. In the proposed algorithm W_s is equal to 0.6 .

Also, c_1 is an acceleration coefficient for extending the velocity of the cat to move in the solution space. This parameter is a constant value and is usually equal to 2.05, but in the proposed CSO, an adaptive formula is used:

$$C(d) = C_s - \frac{(d_{max} - d)}{2 \times d_{max}} \tag{5}$$

Equation (5) demonstrates that the adaptive acceleration coefficient will be gradually increased in every dimension and the maximum value happens in the last dimension. Here C_s is equal to 2.05.
By using these two adaptive parameters, the velocity equation will be updated by equation (6).

$$V_{k,d} = W(d) \times V_{k,d} + r_1 \times C(d) \times \left(X_{best,d} - X_{k,d}\right) \tag{6}$$

B. *New Dynamic Position Update Equation*

In this part, the position equation is turned to a new form. In the pure CSO, the position of cat is including of current information of velocity and position.

Sometimes in many cases, using of previous information in order to estimate the current position is useful. Also, taking the advantages of next information can be appropriate information for updating the cat's position. So we use the two previous/next steps information of velocity and position by applying a new factor which is called 'Forgetting factor'. By this factor, the values of previous and next steps will be different. So the information value for first previous/next step is higher than second previous/next step. It means that the influence of previous/next step is more important than previous/next second step.
New position update equation is described in equation (7).

$$X_{k,d} = \frac{1}{2}[(\text{Position Information}) + (\text{Velocity Information})] \quad (7)$$

$$\text{Position Information} = X_{k,d} + \frac{\left(\gamma \times (X_{k,d+1}) + (1-\gamma) \times (X_{k,d+2})\right)}{2} + \frac{\left(\gamma \times (X_{k,d-1}) + (1-\gamma) \times (X_{k,d-2})\right)}{2}$$

$$\text{Velocity Information} = V_{k,d} + \frac{\left(\gamma \times (V_{k,d+1}) + (1-\gamma) \times (V_{k,d+2})\right)}{2} + \frac{\left(\gamma \times (V_{k,d-1}) + (1-\gamma) \times (V_{k,d-2})\right)}{2}$$

In the proposed algorithm, γ is the forgetting factor and is equal to 0.6 (It is necessary to use $\gamma > 0.5$). This new position update equation is composing two new dynamic terms, average position information and average velocity information. Here, we use the current and the average information of first and second previous/next steps for both velocity and position by applying a forgetting factor (γ).
Fig. 1 shows the process of position updating for cat_k.

IV. EXPERIMENTAL RESULTS

In this paper, simulation experiments are conducted in GCC compiler on an Intel Core 2 Duo at 2.53GHz with 4G real memory. Our aim is to find the optimal solution of the benchmarks and find the minimum. Rastrigrin, Griewank, Ackley, Axis parallel, Trid10 and Zakharov are six test functions used in this work. In TABLE I (A,B), the parameter values of the pure CSO and Adaptive Dynamic CSO are defined respectively. The limitation range of six test functions is mentioned in TABLE II. Also, the experimental results of benchmarks are shown in TABLE III (A, B, C, D, E, and F).

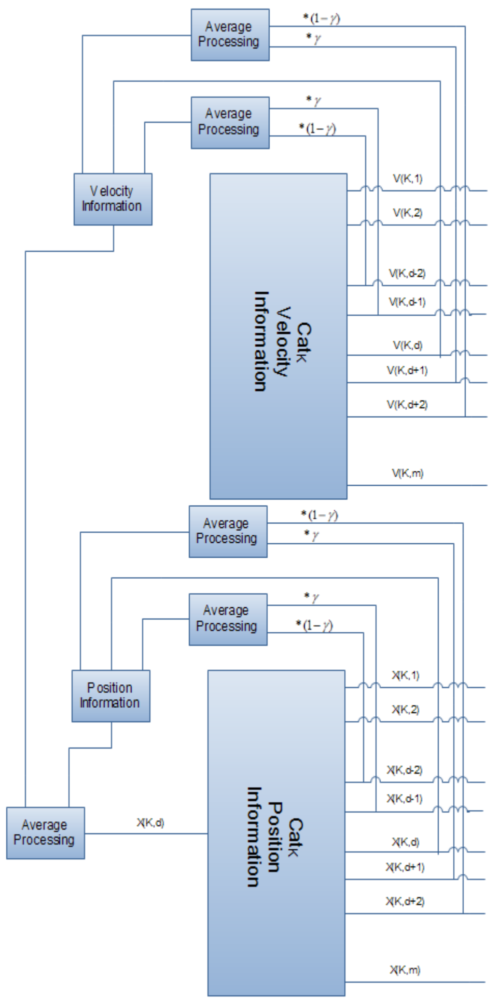

Fig. 1. Process of position updating for cat_k. This figure shows that the position of cat_k depends on the information of second previous/next steps.

TABLE I (A)
PARAMETER SETTINGS FOR PURE CSO

Parameter	Range Or Value
SMP	5
SRD	20%
CDC	80%
MR	2%
C_1	2.05
R_1	[0,1]

TABLE I (B)
PARAMETER SETTINGS FOR ADAPTIVE DYNAMIC CSO

Parameter	Range or Value
SMP	5
SRD	20%
CDC	80%
MR	2%
R_1	[0,1]
Starting Adaptive Acceleration Coefficient (C_s)	2.05
Starting Adaptive Inertia Weight (W_s)	0.6
Forgetting Factor (γ)	0.6

TABLE II
LIMITATION RANGE OF BENCHMARK FUNCTIONS

Function Name	Limitation Range	Minimum	Population Size
Rastrigin	[2.56,5.12]	0	160
Griewank	[300,600]	0	160
Ackley	[-32.678,32.678]	0	160
Axis Parallel	[-5.12,5.12]	0	160
Zokhrov	[-100,100]	0	160
Trid10	[-5,10]	-200	160

TABLE III (A)
EXPERIMENTAL RESULTS OF RASTRIGIN FUNCTION

Algorithm	Average Solution	Best Solution	Best Iteration
Pure CSO	26.64208	1.161127	2000
Adaptive Dynamic CSO	18.08161	0	968

TABLE III (B)
EXPERIMENTAL RESULTS OF GRIEWANK FUNCTION

Algorithm	Average Solution	Best Solution	Best Iteration
Pure CSO	16.34035	12.08197	2000
Adaptive Dynamic CSO	6.005809	2.609684	2000

TABLE III (C)
EXPERIMENTAL RESULTS OF ACKLEY FUNCTION

Algorithm	Average Solution	Best Solution	Best Iteration
Pure CSO	1.031684	0.004589	3500
Adaptive Dynamic CSO	1.019844	0	1264

TABLE III (D)
EXPERIMENTAL RESULTS OF AXIS PARALLEL FUNCTION

Algorithm	Average Solution	Best Solution	Best Iteration
Pure CSO	66.38478	0	665
Adaptive Dynamic CSO	49.3307	0	408

TABLE III (E)
EXPERIMENTAL RESULTS OF ZAKHAROV FUNCTION

Algorithm	Average Solution	Best Solution	Best Iteration
Pure CSO	0.68080	0	475
Adaptive Dynamic CSO	0.442817	0	275

TABLE III (F)
EXPERIMENTAL RESULTS OF TRID10 FUNCTION

Algorithm	Average Solution	Best Solution	Best Iteration
Pure CSO	-125.0846	-178.065182	2000
Adaptive Dynamic CSO	-173.384	-194.000593	2000

TABLE IV
PROCESS TIME OF SIX TEST FUNCTIONS

Algorithm	Function Name	Average Process Time
CSO	Rastrigin	6.4117
Proposed CSO	Rastrigin	3.5074
CSO	Griewank	5.6785
Proposed CSO	Griewank	8.0220
CSO	Ackley	8.6873
Proposed CSO	Ackley	4.7585
CSO	Axis Parallel	3.7814
Proposed CSO	Axis Parallel	2.5203
CSO	Zakharov	2.5907
Proposed CSO	Zakharov	1.6085
CSO	Trid10	2.7905
Proposed CSO	Trid10	2.3448

Process time is an important factor for us to know that whether our proposed algorithm runs as well as we expect or not. So in TABLE IV, we demonstrate the running time of pure CSO and our proposed CSO. It indicates that for six test functions, proposed CSO always runs faster than pure CSO except for Griewank function. For example, the process time of proposed CSO for Rastrigin function is 6.4117 which is less than pure CSO.

Rastrigin's function is based on the function of De Jong with the addition of cosine modulation in order to produce frequent local minima. Thus, the test function is highly multimodal [8]. Function has the following definition

$$F(x) = \sum_{d=1}^{M} [x_d{}^2 - 10\cos(2\pi x_d)^2 + 10] \qquad (8)$$

As shown in Fig. 2(A) and TABLE III (A) by running the pure CSO, the best fitness value for Rastrigin function is equal to 1.161127. But the proposed algorithm can find the best optimal and the result is reached to 0.

Griewangk function has many wide spread local minima regularly distributed [8].Function has the following definition

$$F(x) = \frac{1}{4000}\sum_{d=1}^{M} x_d{}^2 - \prod_{d=1}^{M}\cos\left(\frac{x_d}{\sqrt{d}}\right) + 1 \qquad (9)$$

As demonstrated in Fig. 2(B) and TABLE III (B), improved CSO in comparison with pure CSO, can find the better best solution in a same iteration.

Ackley's function is a widely used multimodal test function [8]. It has the following definition

$$F(x) = -20.\exp\left(-0.2\sqrt{\frac{1}{n}\sum_{i=1}^{n} x_i{}^2}\right) - \exp\left(\frac{1}{n}\sum_{i=1}^{n}\cos(2\pi x_i)\right) - 20 \quad (10)$$

Fig. 2(C) shows that the proposed CSO can find the best solution of Ackley function in less iteration. As shown, proposed CSO can get the optimal solution in 1264th iteration. Meanwhile, the best solution of pure CSO is 0.004 in the 3500th iteration.

The axis parallel hyper-ellipsoid is similar to function of De Jong. It is also known as the weighted sphere model. Function is continuous, convex and unimodal [8]. It has the following general definition

$$F(x) = \sum_{i=1}^{n} i\, x_i^2 \qquad (11)$$

As it is shown in Fig. 3(D) and Table III (D), the optimal solution of axis parallel function is zero and both algorithms can find it. But the proposed CSO, in comparison with pure CSO can find it in less iteration.

The fifth function is Zakharov function. Zakharov equation is shown in equation (12).

$$F(x) = \sum_{i=1}^{n} (x_i)^2 + (\sum_{i=1}^{n} 0.5 i x_i)^2 + (\sum_{i=1}^{n} 0.5 i x_i)^4 \qquad (12)$$

Fig. 3(E) illustrates that the final solution of proposed CSO is better than pure CSO. It shows that proposed CSO can find the best answer in the 275th iteration which is 200 iterations less than pure CSO.

Trid10 function is the last benchmark for making a comparison between pure CSO and proposed CSO. It has no local minimum except the global one. Equation (13) shows the equation of Trid10 function.

$$F(x) = \sum_{i=1}^{10} (x_i - 1)^2 - \sum_{i=2}^{10} x_i x_{i-1} \qquad (13)$$

Table III (F) and Fig. 3(F) show that the best solution of proposed algorithm is equal to -194.000593. In comparison with pure CSO, this value is closer to global minimum of function. (It is necessary to say that, due to many negative fitness values of Trid10 function, fitness values are added by number +194 in order to show the picture in log axis in Fig. 3(F)).

V. Conclusions

In the optimization theory, many algorithms are proposed and some of them are based on swarm intelligence. These types of algorithms imitate the behavior of animals.

Cat Swarm Optimization (CSO) is a new optimization algorithm for finding the best global solution of a function which imitates the behavior of cats and models into two modes. In comparison with Particle Swarm Optimization CSO can find the better global solution. But CSO is more complex than PSO and because of its complexity, sometimes takes a long time to find the best solution. So, by improving the pure CSO, we proposed a new algorithm namely 'Adaptive Dynamic Cat Swarm Optimization'.

First of all, an adaptive inertia weight is added to velocity equation in order to facilitate the global and local search ability adaptively. Also an adaptive formula is applied for updating the acceleration coefficient. Then, a new form of position equation composing the novel position and velocity information is used in proposed CSO. In new position equation, in addition to current information of position and velocity, we used the average information of two previous/next steps by applying a forgetting factor. Experimental results of six benchmarks indicated that the improved CSO in comparison with pure CSO has much better solution and achieves the best fitness value and converge in less iteration. Also by considering the process time, it is clear that the proposed algorithm runs faster than pure CSO.

References

[1] M. Dorigo, L.M. Gambardella, "Ant colony system: a cooperative learning approach to the traveling salesman problem", IEEE Trans. on Evolutionary Computation. 26 (1) (1997) pp 53-66.

[2] R. Eberhart, J. Kennedy, "A new optimizer using particle swarm theory", Sixth International Symposium on Micro Machine and Human Science (1995),pp 39-43.

[3] D. Karaboga, "An idea based on Honey Bee swarm for numerical optimization",TechnicalReport-TR06,ErciyesUniversity,Engineering Faculty, ComputerEngineering Department 2005.

[4] S.C.Chu, P.W.Tsai,and J.S.Pan, "Cat Swarm Optimization," LNAI 4099, 3 (1), Berlin Heidelberg: Springer-Verlag, 2006, pp. 854– 858.

[5] B. Santosa,M. Ningrum, "Cat Swarm Optimization for Clustering", International Conference of Soft Computing and Pattern Recognition, (2009), pp 54-59.

[6] S.C. Chu, J.F. Roddick, J.S. Pan, "Ant colony system with communication strategies",Information Sciences 167 (2004), pp 63-76.

[7] Y. Shi, R. Eberhart, "Empirical study of particle swarm optimization", Congress on Evolutionary Computation. (1999), pp 1945-1950.

[8] M. Molga, C. Smutnicki, "Test functions for optimization needs", 3 kwietnia 2005.

[9] M.Orouskhani, M. Mansouri,and M. Teshnehlab, "Average-Inertia weighted Cat swarm optimization ," LNCS, Berlin Heidelberg: Springer-Verlag, 2011, pp 321– 328.

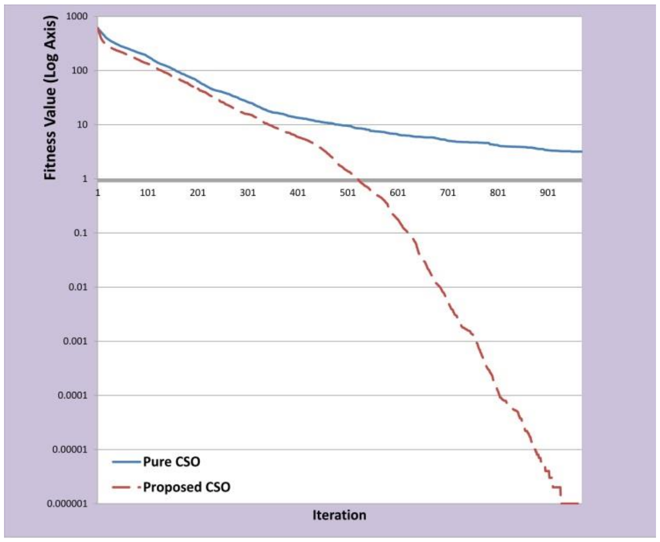

Fig. 2(A). Experimental Results Of Rastrigin Function

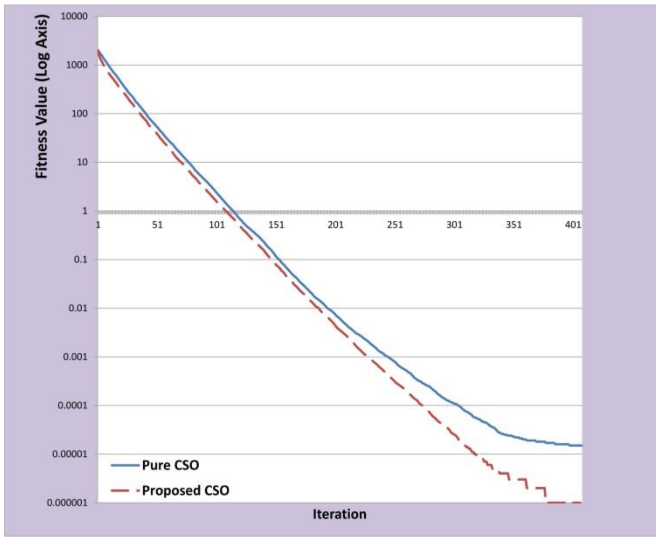

Fig. 2(D). Experimental Results Of Axis Parallel Function

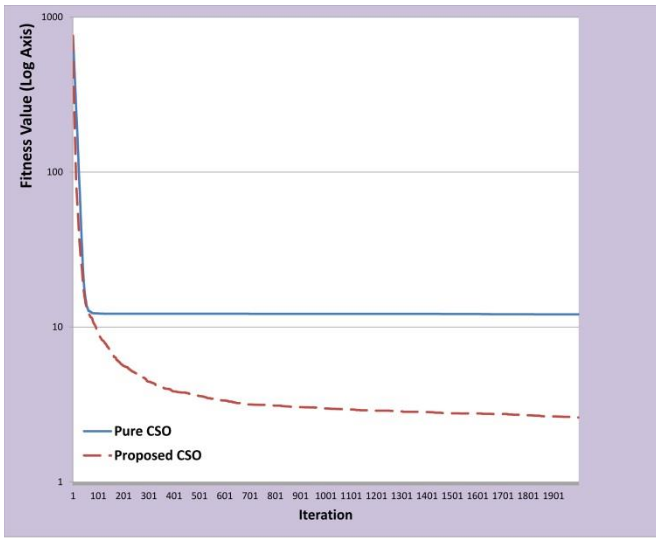

Fig. 2(B). Experimental Results Of Griewank Function

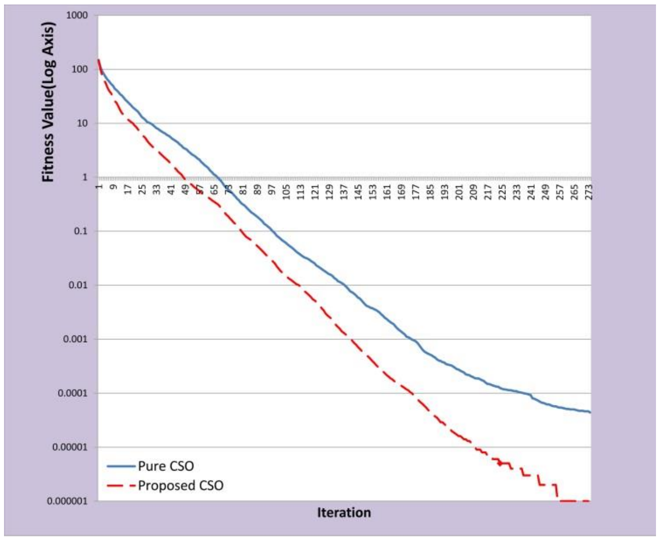

Fig. 2(E). Experimental Results Of Zakharov Function

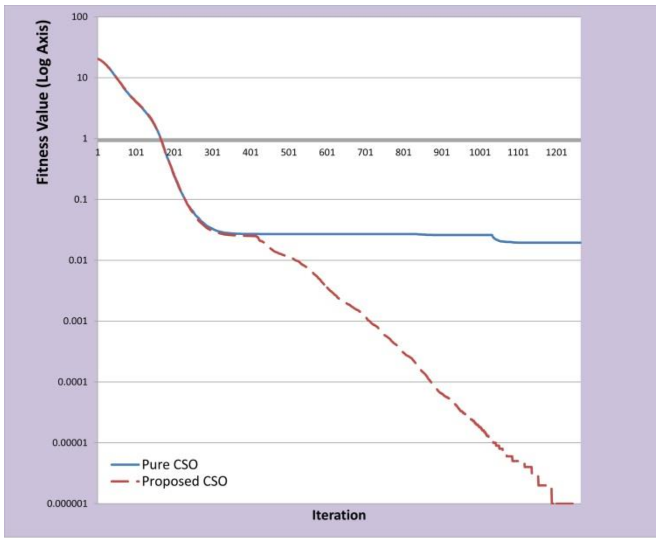

Fig. 2(C). Experimental Results Of Ackley Function

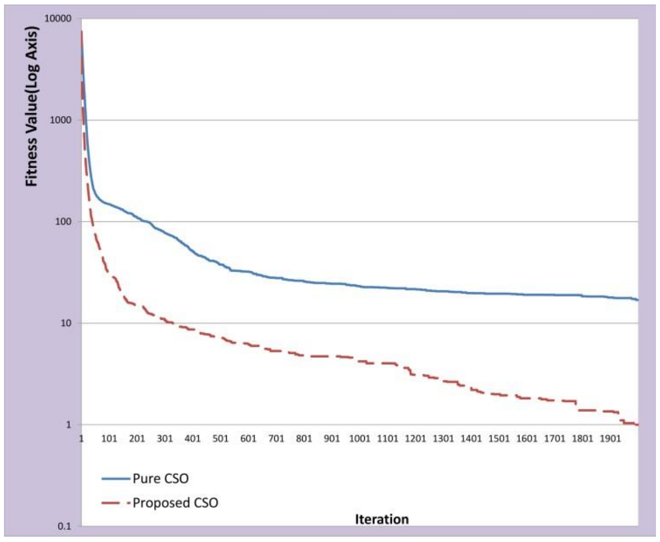

Fig. 2(F). Experimental Results Of Trid 10 Function (For showing the fitness values in log axis, we added up the values with number +194)