An Improved Particle Swarm Optimization Algorithm with Chi-Square Mutation Strategy

Waqas Haider Bangyal[1]
Department of Computer Science
University of Gujrat,Pakistan

Hafsa Batool[3]
Department of Computing and Technology
Iqra University,Islamabad,Pakistan

Jamil Ahmed[5]
Kohat University of Science and Technology
(KUST),Pakistan

Hafiz Tayyab Rauf[2]
Department of Computer Science
University of Gujrat,Pakistan

Saad Abdullah Bangyal[4]
Department of Computer Science
Abasysn University,Islamabad, Pakistan

Sobia Pervaiz[6]
Department of Software Engineering
University of Gujrat, Gujrat, Pakistan

***Abstract*—Particle Swarm Optimization (PSO) algorithm is a population-based strong stochastic search strategy empowered from the inherent way of the bee swarm or animal herds for seeking their foods. Consequently, flexibility for the numerical experimentation, PSO has been used to resolve diverse kind of optimization problems. PSO is much of the time caught in local optima in the meantime taking care of the complex real-world problems.Considering this, a novel modified PSO is introduced by proposing a chi square mutation method. The main functionality of mutation operator in PSO is quick convergence and escapes from the local minima. Population initialization plays a critical role in meta-heuristic algorithm. Moreover, in this work, to improve the convergence, rather applying random distribution for initialization, two quasi random sequences Halton and Sobol have been applied and properly joined with chi-square mutated PSO (Chi-Square PSO) algorithm. The promising experimental result suggests the superiority of the proposed technique. The results present foresight that how the proposed mutation operator influences on the value of cost function and divergence. The proposed mutated strategy is applied for eight (8) benchmark functions extensively used in the literature. The simulation results verify that Chi-Square PSO provide efficient results over other tested algorithms implemented for the function optimization.**

***Keywords*—Particle Swarm Optimization; Chi-Square Mutation; Population Initialization.**

I. Introduction

The term “swarm intelligence” is practiced as to explain the algorithm and distributed problem solvers, motivated by the common actions of colonies of insects and other animal groups. Swarm Intelligence (SI) based systems are normally buildup of simple agents of population that are communicating internally with each other and with their environment [1].

Likewise, other evolutionary algorithms (EAs), the particle swarm optimization (PSO) technique is a population based meta-heuristic search approach that devised from nature aspect. Such type of approaches commonly needs extra objective function evaluations that compared with gradient search techniques. These techniques offer stunning features like easiness in the numerical implementation for both discrete and continuous optimization problems and more powerful solution creations for seeking the global solutions. PSO algorithm seeks the optimum solution inside the population called as flock or swarm. PSO avails the advantage from two type of training: cognitive training focused on particle’s own history while social training concentrated on swarm history of information sharing collected from all the individuals of the swarm.

Kennedy and Eberhart formally introduced PSO, it got enticed notable attraction in last decade. Vast majority of research on this subject is concerned either with the mathematical analysis or to enhance the algorithm for attaining the quicker, robust, and scalable candidate solutions. Main motivation of the posterior, stuck in local optima by the solution or premature convergence. An EA is trapped to the local optima, if it is not capable to investigate all the search space except the explored region, and another area persists that hold a best solution better than to the currently find solution. One of the major causes for the deficiency of diversity is premature convergence.

Diversity is vital important for vigorous searching in the given search space while the mutations are fundamental operators to give the dynamic diversity inside the swarm [2]. Besides the designing of new mutation operator, researchers have placed fewer efforts to explore that how to use the mutation operator [3] and to find what type of diversity supposed to be available in the swarm. Hence, after the detail exploration of diversity concept focused on qualification and quantification studies, this paper presents new mutation strategy and operator to give useful diversity in the swarm. The new proposed technique has been used on the selected benchmark functions. From these test cases, it is shown that proposed technique has given the better results than other variants of PSO. The core objective of this strategy was to find which particle should be mutated and when; it should be mutated. The proposed technique also gives controlled diversity in the swarm.

The rest of the paper is structured like this: Section 2 reviews the technical background of PSO algorithm where Standard PSO is discussed in Section 3. Section 4 carries the proposed algorithm. In Section 5, the computational simulation results and comparison are carried out. Conclusion and future work are presented in Section 6.

II. Literature Review

PSO might have some problems associated to rate of convergence, premature converged solution, poor accuracy and failure of diversity. Several modifications have been introduced till now to overcome these issues [4]. Improved approaches can split into two types concentrating on improvements of hardware and software. The hardware is concerned with parallel computing. While the software improvements can be more categorized like integrating with other heuristic techniques, motivation from other stochastic methods, manipulating and restructuring of velocity update equation, managing the neighborhood topology, tuning the parameters during the update phase and swarm size.

PSO has been incorporated with other metaheuristic algorithm named as hybridization. The standard hybrid application is to employ PSO with other stochastic based technique like GA or other evolutionary computing algorithms. Furthermore, other approach is to utilize the gradient-based approach as incorporated segment. Restructuring the equations (2.2) and (2.3) has the significant importance in the improvements of the algorithm. In such rearranging or manipulations and updation of the equation of velocity is expanded [5] with an additional factor or reducing it rely on the technique.

Standard PSO parameters ω , c_1 and $c_2$can be used as constant, periodically, chaotic, random, adaptive, linear changeable or nonlinear relying on time or other concerns like cost function and velocity measures. Neighborhood topologies created by choosing the position vector of particle from Pi to Pg .In standard algorithm of PSO, individual particle is pulled towards the particle's personal best and global best.

Consequently, none else personal particle's best position vector will take the appealing impact on a single particle itself and also best position vector also will not be modified at each epoch. At present, various neighborhood topologies are presented [6], population of swarm is also explored in many different dimensions.

By selecting swarm size dynamically might be encouraging to find out the solution few optimization real world problems [7]. Motivation from other population based search algorithm like GA, opposition based learning techniques, simulating annealing are the famous approaches used for the fundamental working enhancement of the standard PSO. In addition, GA common operator's selection, crossover, elitism and mutation can incorporated in PSO architecture [8].

Mutation is mainly used in GA operators due to shortcomings of PSO [9], because of deficient diversity that drives the swarm particles to approach the position discovered yet in the swarm cause local minima. Lack of diversity improving techniques, majority of optimization approaches cannot be effectively explore or exploit the given search space. Mutation operator set up the new particles by altering the current particle in the swarm [10], so incorporating diversity in the swarm and probably stopping stagnation of exploration for local minima.

Consequently, the mutation application technique draws some new modification to the algorithm like condition for mutation application, the placement of mutation is to apply and choose the different distribution sequences. In order to find the criterion for mutation applications, description of diversity threshold [11], similarity and mutation probability ration may be used. Similarly, for viable random distribution, sequences are Cauchy distribution, Chaotic distribution, Beta distribution and Gaussian distribution.

Higashi and Iba [12] implemented initial practice of mutated operator by introducing the random number based on Gaussian distribution for changing the particle dimension. Likewise Higashi, a new mutation operator is proposed by Stacey et al [13] by mutilating the particle using a number drawn from cauchy distribution to keep the diversity.

Pant et al. [14] gave a new mutation operator named as Sobol mutation operator that utilizes the quasi-random sequences to explore the search space better than random distribution. To keep the diversity in the swarm Zavala et al. [15] gave two separate perturbation operators known as C-perturbation operator and M-perturbation operator and implemented to personal best position in lieu of perturbing the position vector of particle in the swarm.

Jia et al. [16] used two mutation operator called as Chaotic mutation operator and Gaussian mutation operator. By using chaotic mutation operator, global searching was performed while to solve the issue of local exploitation, Gaussian mutation was integrated into PSO. To avoid from local minima, Wu et al. [17] proposed a novel mutation named as cloud mutation having the features of randomness; and keeping the capability of standard cloud model.

Chen et al. [18] defined a novel mutation operator named as adaptive mutation. In this variant, they contain the potential particle for mutation around the global best discovered by either particle in given search space during preliminary epochs.

Liu et al. [19] defined new mutation operator with names Chaotic PSO by integrating chaos in PSO having adaptive inertia weight. To avoid from premature convergence and keeping the diversity in swarm, Yang et al. [20] proposed a new mutation operator that uses the chaotic probability into the algorithm having inertia weight linear decreasing.

Liu et al. [21] defined an improved version of PSO focused on the concept of collectivity in which resemblance focus on current global best position vector and a particle in the swarm. Biao et al. [22] proposed PSO based on fast position convergence, in order to prevent from unwanted epochs in each local optima, and it is used only when required.

According to [23], by proposing the intelligent PSO called (PSO-IM) based on intelligent mutation. PSO-IM includes two types of mutation, uniform mutation and non uniform mutation. Tournament selection strategy is adopted to select particles randomly in each tournament for mutation operation while mutation probability (p_m) is controlled dynamically with fuzzy controllers. The process of tournament selection continues until a predetermined number of individuals selected.

Uniform mutation replaced selected particles by a random number while non uniform mutation applied on rest of particles to overcome local minima. The performance of PSO-IM was tested on six popular nonlinear global optimization problems and four nonlinear reliability problems .

PSO-IM outperforms the original PSO in all test cases. K. Wang and F. Li [24] brought dynamic chaotic behavior in PSO called (dcmPSO) to gain global exploration at the start of iteration and local exploitation at the end of the iteration. The logistic map turns the mutation process into chaotic state. At the end of iteration a temporary leader is found. If the temporary leader does not improved for a predefined constant number. Than chaotic state turns on to update temporary leader. The proposed technique proved its effectiveness on Park-Ramirez bioreactor dynamic problem.

The mutation operator gives the diversity ability in the swarm. Thus, for mutation application, mutation operator types and their application technique is the major decision portion. Besides the proposing of new mutation operator to avoid from local minima, the researchers put very limited attempts to examine how to use these new deigned mutated operators through the PSO procedure and find which kind of diversity in the swarm should be available.

III. Particle Swarm Optimization

PSO is a relatively new metaheuristic optimization search algorithm that uses the pool of best solutions to find the optimum solution. The search of optimum solution is managed by adopting the social behavior of bees, herds of animal and bird flock [25]. Considering the ants' colonies, bee swarm, flock of birds, animal herds and school of fish revealed that collective venture of a bunch is normally more productive than single effort. Each single entity inside a group has a specific ability to achieve the goal. During working in a group, action of a candidate led not only by the candidate's understanding to accomplish the goal but also through the social action. The entire candidates inside a group share the experience by following the common goal, and each individual discover not only from its own experience as well as from experience of its neighbors. This accelerates the searching process considerably fast. This kind of social interaction was the origination of the PSO algorithm, elaborated in the paper.

PSO works on bunch of candidates. Each candidate termed as a particle which depicts a solution for the optimization problem. For n dimensional problem, a particle depicted by n-dimensional vector, x represents a particle position. Each particle has a fitness value that shows the worth of individual's solution representation. Swarm's particle moves in n dimension given search space. Beside the position vector in search space, each particle has velocity vector that finds the step size and direction of particle motion. Social communication emulated by making neighbors inside the swarm. Each particle saves its own personal best position discovered until now and can inquire neighbor particles for the best position as found by the neighboring particles so far. PSO neighboring swarm information sharing techniques have been discussed in the literature [26].

The architecture and size of neighbor finds the method where information shared between the particles. PSO seeks for an optimum solution by moving the particles over the n dimensional search space. For each step k, position vector of particle X_{id}^kupdated by adding the velocity vector X_{id}^{k+1} of the particle to the prior position vector.

$$X_{id}^{k+1} = X_{id}^k + V_{id}^{k+1} \tag{1}$$

The velocity vector of the particle finds out the step size and direction. The velocity equation is given below:

$$V_{id}^{k+1} = V_{id}^k + c_1 r_1 (p_{id} - x_{id}) + c_2 r_2 (p_{gd} - x_{id}) \tag{2}$$

Where acceleration coefficients c_1 and c_2 applied to measure the effect of cognitive part (second term of equation 2.2) and social component (third term of equation 2.2); r_1 and r_2 are random numbers vectors, where each part taken from uniform distribution in the range between zero and one. In each iteration value of r_1 and r_2 changed. p_{id} is particle personal best position acquired by the particle 'y' yet; where as p_{gd} is the best global position discovered by any particle in the neighbor of particles.

Although in the original form of PSO, none method exists to restrain the velocity. It causes the feeble nature of algorithm, particularly in the area of local minima. In the back adaptations of PSO, this insufficiency was handled by incorporating two new parameters, named as inertia weight proposed by Shi and Eberhart [27] and constriction factor (x) presented by Clerc [28]. Therefore, each particle is approaching to the best position confronted by itself until now, along with entire best position found by the neighborhood particles, so far. A maximum velocity $V_(max)$ occasionally adjusted to restrict particle velocity in each dimension of the search space. Velocity clamping performed to stop the particles from traveling the search space quickly, since extremely big steps stop particles from exploiting good areas of the search space. $V_(max)$ is implemented by stopping V_id^k in each dimension space [29]. Major advancement in PSO is the addition of inertia weight term to restrain the influence of value of older velocity on new velocity is given below:

$$V_{id}^{k+1} = \omega V_{id}^k + c_1 r_1 (p_{id} - x_{id}) + c_2 r_2 (p_{gd} - x_{id}) \tag{3}$$

Where the term ω is named as inertia weight; two positive constants c_1 and c_2 are cognitive and social parameter, respectively.

Currently, Clerc [5] inducted one more parameter named as constriction factor, which may help to anchor the convergence. The constriction factor model clarifies the system by choosing ω , c_1 and c_2 for the guaranteed convergence. By selecting these parameters values accurately , the velocities of all particles are selected in the range $[-V_(max), V_(max)]$. Eberhart and Shi investigate the performance of PSO using velocity clamping $V_(max)$ with constriction factor. The experimental outcomes showed that incorporation of constriction factor [30] boosts the convergence of algorithm. When constriction factor is examined on test benchmark problems, it remained unsuccessful to attain the certain threshold error for the given problem within allotted stipulated number of epochs. Subsequently it was established that as the particles remained away from the given search space, the constriction factor unsuccessful to attain the given number of epochs. After setting the velocity clamping to constriction factor, the performance was enhanced for all benchmark problems.

After adjusting the values of c_1 and c_2 in the equation, it might provide for accelerating the convergence of the algorithm. Choosing the default values $c_1 = c_2 = 2$ was suggested but the simulation results shows that different combination according to the problem nature may give better performance. Latest work [10] reveals that it could be still best to select a smaller social parameter than cognitive parameter. $r_1 and r_2$ are random numbers vectors where each part taken from uniform distribution in the range U(0,1), has been utilized to keep the diversity. Magnitude of the velocities managed by the constriction parameter factor x like$V_(max)$ parameter practiced in the initial version of PSO. The algorithm gives the quicker convergence speed, when x and $V_(max)$ are collectively used.

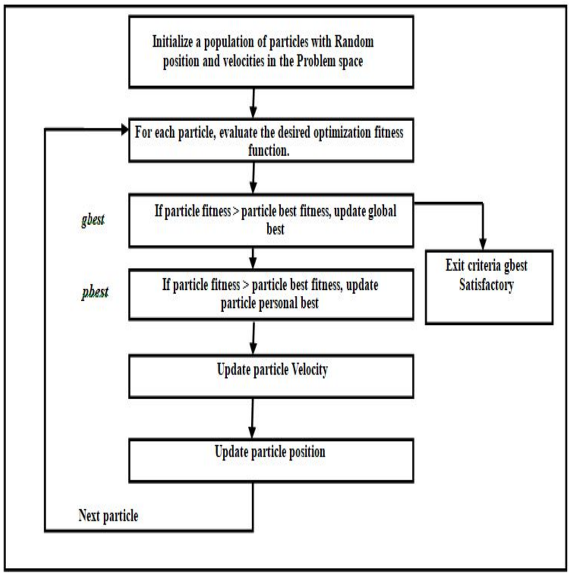


Fig. 1. Standard PSO.

Three primarily operators are normally used in EA approaches: the selection, the recombination and the mutation operator. PSO does not have a recombination operator. Stochastic improvement of a particle towards its past best position, notwithstanding the best particle of the swarm [31], mimics the recombination method in EA. In PSO, transfer of information occur only among the particles of the swarm by their own experience and the experience of best particle in the population, rather choosing the fitness from elite "parent" to the off spring in GA's. Furthermore, in PSO position updating vector matches the mutation operator in GA. PSO relate to the kind of EAs that does not practice the concept of "survival of fittest". It does not use the selection procedure directly. Hence, particle with less fitness can exist during optimization process and possible visit any point of the search space in the swarm. Pseudo code of standard PSO is presented in the Fig. 1:

IV. Proposed Methodology Scheme

PSO is an evolutionary intelligent searching approach relies on the population. It is used to find the optimum solution in the search space based on collective cooperation and contest between the groups. Like other swarm intelligence techniques, PSO algorithm faces the problem of premature convergence. In PSO algorithm, nature of particle is determined by its global best location and previous best location of the particle discovered yet [32]. When these best particles trapped in local minima, the current particle could be fallen in local minima. To recover from this phenomenon, chi-square mutation operator is introduced. If particle is fallen in local minima, proposed mutation operator will support it, recover it from local minima and move this particle to another location far from the local minima. In this proposed technique, global best particle is mutated by Chi -Square. By using chi-square in PSO, PSO takes a long jump to escape from local minima. The chi square distribution is one of the mathematical distributions which have large usage in statistical work. The term $chisquare$ (pronounced as ch). The Greek letter X is used to represent this distribution. The probability density function (pdf) of the chi-squared distribution is given in Equation 4.

$$mut(x) = Zx(1 + Chi - square(\partial) \tag{4}$$

in the Eq 4 (∂) is fixed as 0.1 and Zx refereed as numerical object . Normally in population, based meta heuristic search algorithm like PSO, working performance relies on population initialization, which finds the succeeding methodology evaluations. Standard PSO uses the random uniform distribution numbers for initialization of population [33].

Because of this and the issues of large dimensions, search space exploration is ineffective and non-distribution of swarm particles may take place. It causes the slow convergence and creates the process to escape from local minima. Recently, to avoid these shortcomings, different population initialization methodologies have been developed and utilized in various research fields to prevent from premature convergence in metaheuristic algorithms and enhance the efficient exploration in search space.

Quasi random sequences are not as much random than the pseudo random sequences, however these are more strong for computational techniques who rely on creation of the random numbers. A few famous quasi random numbers are Sobol, Halton, Torus, Faure and Vander Corput.

Such sequences have been employed for initialization. A simulation outcome depicts a remarkable progress over standard PSO that employed uniform distribution. The experimental results revealed that using QRS for initiation of population enhance the performance of meta heuristic algorithms. The flow chart of the proposed technique is presented in Fig. 2. The prime phases of proposed technique are given in Algorithm 1.

V. Experimental Results

The proposed chi-square mutated PSO (Chi-Square PSO) is simulated in C++ and applied on computer with 2.3 GHz Core (M) 2 Duo CPU processor . In order to measure the execution of the proposed chi-square PSO algorithm, a group of benchmark functions has been utilized to do the comparison with many other improved PSO techniques with traditional PSO, Adaptive PSO and different initialization techniques. Eight non-linear test functions are chosen here to examine the optimization outcomes of proposed Chi-Square PSO that are normally applied to investigate the performance of any technique.

A. Experimental Setup

The parameters for simulation used as c1=c2=1.45, inertia weight w is used in the interval [0.9,0.4] and swarm size is 20. For all the simulation, the function dimensions are D=10, 20 and 30 and maximum number of epochs is 3000. For fair comparison, all techniques apply similar parameters. In order to check the performance of each technique, all algorithms tested for 30 runs.

B. Benchmark Functions

This segment contains the eight benchmark functions applied to test the performance evaluation of the proposed algorithm. List of these functions is available in the Table I, D shows the dimensionality of the problem, S represents the interval of the variables and fmin denotes the common global optimum minimum value.

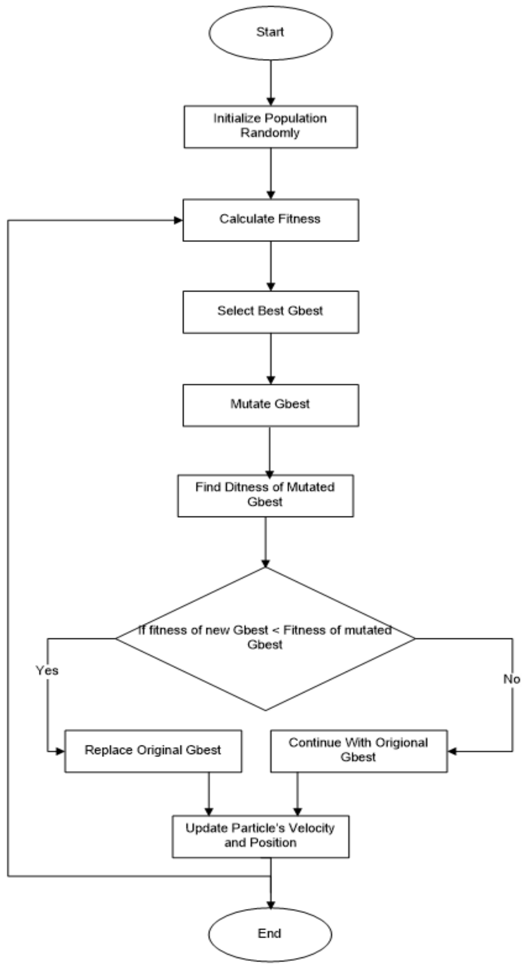


Fig. 2. Flow Chart for proposed PSO

Algorithm 1 Proposed PSO

Step 1: Initialization
Set epoch number I=0, swarm population size NP, the particles dimensions D in the swarm , wmax, wmin, C1, C2, For each particle Pi in the swarm
Step 1.1: Initialize the Particle position Xi
Step 1.2: Initialize the Particle velocity Vi
Step 1.3: Calculate the fitness value fi
Step 1.4: Set global best position gbest with the particle with best-evaluated fitness function computed value in the swarm.
Step 1.5: Set the personal best location pbest of each particle in the swarm as Pi= Xi.
Step 2:Compare the value of current particle's fitness in the swarm and its previous best location pbest . If the particle fitness is better than pbest, then substitute the pbest with its current fitness; else retain the particle Pi unchanged
Step 3:Compare the value of current particle's fitness in the swarm and its global best location gbest . If the particle fitness is belter than gbest, then substitute the global best position gbest with its current fitness ; else retain the particle Pi unchanged
Step 4:Execute chi square mutation to each particle
Step 5:Updating the each particle velocity using the equation
Step 6:Updating the each particle position using the equation
Step 7:Apply chi square mutation on the global best particle by using Eq (4).
If mutated particle is better than current Pi, substitute the current particle Pi with mutated particle; else retain the particle Pi unchanged
Step 8:If the stopping criteria met, stop the epoch process; else go to step 2

C. Discussion

The main objective is to examine the progress produced by the proposed technique. For a fair comparison, the performance of following PSO variants is tested: Standard PSO, PSO with adaption mutation (AMPSO), Chi-Square PSO and also compared the results Chi-Square PSO with Sobol initialization and Chi-Square PSO with Halton initialization using mutation operator and without using the mutation operator. From Fig. 3 to 8, it is shown that not only the Chi-Square PSO provide fast convergence speed over AMPSO and from standard PSO. Simulation Results depicts that proposed technique improve the exploration capability but also provide fast convergence and achieve the global diversities and global optima. Table II shows the comparative results.

VI. CONCLUSION

This paper introduces a new approach of PSO algorithm by proposing a Chi-Square mutation operator and using two different quasi random initialization techniques have been joined with proposed PSO and employed on function optimization problems. The proposed mutation strategy maintains the diversity of the swarm and improves the global searching capability. The simulation results show that the proposed mutated PSO has better convergence accuracy and can escape from premature convergence successfully and compared with other recognized variants of PSO. The future work is to theoretically examine its effects and employ it some real world complex optimization problems. For future research work, it will be exciting to focus on the proposed approach to many real-world engineering applications. Furthermore, it is interestingly important to implement the proposed technique for engineering optimization problem to enhance its practicability and rightness.

TABLE I. EIGHT STANDARD BENCHMARK FUNCTIONS

Function Name	Objective Function	Search space
Sphere	$Minf(x) = \sum_{i=1}^{n} x_i^2$	$-5.12 \leq x_i \leq 5.12$
Grienwank	$Minf(x) = \frac{1}{4000} \sum_{i=1}^{n} x_i^2 - \prod_{i=1}^{n} cos?(\frac{x_i}{?i}) + 1$	$-600 \leq x_i \leq 600$
Rosenbrock	$Minf(x) = \sum_{i=1}^{n-1} [1000(x_{i+1} - x_i^2)^2 + (x_i - 1)^2]$	$-5 \leq x_i \leq 5$
Rastrigin	$Minf(x) = 10n + \sum_{i=1}^{n} [x_i^2 - 10cos?(2\pi x)]_i$	$-5.12 \leq x_i \leq 5.12$
Ackley	$Minf(x) = -20exp(-0.2\sqrt{\frac{1}{n} \sum_{i=1}^{n} [x_i^2} - exp(\frac{1}{n} \sum_{i=1}^{n} cos?(2\pi x_i)) + 20 + e$	$-30 \leq x_i \leq 30$
Schwefel	$Minf(x) = \sum_{i=1}^{n} -x_i \sin(-1\sqrt{\lvert x_i \rvert})$	$-500 \leq x_i \leq 500$
De Jong's	$Minf(x) = \sum_{i=1}^{n} (x_i^2)$	$-5.12 \leq x_i \leq 5.12$
Axis parallel hyper-ellipsoid	$Minf(x) = \sum_{i=1}^{n} (i.x_i^2)$	$-5.12 \leq x_i \leq 5.12$

TABLE II. COMPREHENSIVE RESULTS

				Results						
Sr	**Name**	**DIM**	**Iter**	**PSO**	**AMPSO**	**CPSO**	**HD-CPSO without CPSO**	**HD-CPSO with CPSO**	**SD-CPSO without CPSO**	**SD-CPSO with CPSO**
F1	**Sphere**	10	1000	3.03E-02	9.87E-02	2.40E-03	3.01E-02	5.60E-03	5.65E-02	2.20E-03
		20	2000	5.72E+00	5.25E+00	5.99E-01	2.47E+00	5.89E-01	3.34E+00	4.03E-01
		30	3000	1.98E+01	1.84E+01	2.29E+00	9.17E+00	1.70E+00	9.51E+00	4.56E+00
F2	**Grienwank**	10	1000	5.71E-01	5.18E-01	3.81E-01	8.84E-01	6.60E-01	6.91E-01	4.35E-01
		20	2000	1.39E+01	1.38E+01	2.70E+00	8.14E+00	2.33E+00	9.78E+00	2.03E+00
		30	3000	5.37E+01	4.49E+01	8.86E+00	3.56E+01	6.49E+00	2.77E+01	9.59E+00
F3	**Rosenbrock**	10	1000	2.69E+01	4.30E+00	2.98E+00	7.11E+00	4.24E+00	6.87E+00	2.92E+00
		20	2000	5.85E+01	3.37E+01	2.76E+01	3.69E+01	1.90E+01	2.66E+01	1.95E+01
		30	3000	1.82E+02	1.20E+02	5.20E+01	1.01E+02	5.09E+01	7.27E+01	3.70E+01
F4	**Rastrigin**	10	1000	3.21E+02	3.22E+02	3.27E+02	3.08E+02	3.21E+02	3.21E+02	3.29E+02
		20	2000	3.95E+02	3.94E+02	3.78E+02	3.82E+02	3.51E+02	3.85E+02	3.44E+02
		30	3000	4.75E+02	4.73E+02	4.23E+02	4.24E+02	3.87E+02	4.51E+02	3.82E+02
F5	**Ackley**	10	1000	1.51E+01	1.51E+01	1.51E+01	1.51E+01	1.50E+01	1.51E+01	1.51E+01
		20	2000	1.53E+01	1.53E+01	1.53E+01	1.54E+01	1.54E+01	1.53E+01	1.53E+01
		30	3000	1.53E+01	1.53E+01	1.53E+01	1.54E+01	1.53E+01	1.53E+01	1.53E+01
F6	**Schwefel**	10	1000	8.88E-17	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	1.78E-16
		20	2000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00
		30	3000	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00
F7	**De Jong's**	10	1000	1.08E-02	1.68E-02	3.14E-04	5.06E-02	1.70E-03	5.65E-02	7.00E-04
		20	2000	4.48E+00	3.17E+00	9.17E-01	1.94E+00	2.50E-01	3.34E+00	5.29E-01
		30	3000	1.45E+01	1.52E+01	4.18E+00	6.81E+00	1.50E+00	9.51E+00	2.29E+00
F8	**Axis parallel hyper-ellipsoid**	10	1000	5.27E-01	4.51E-01	5.20E-02	1.52E+00	1.87E-02	6.49E-01	1.89E-01
		20	2000	2.92E+01	3.69E+01	1.20E+01	3.57E+01	4.22E+00	2.08E+01	4.95E+00
		30	3000	1.95E+02	3.69E+01	4.61E+01	1.18E+02	3.30E+01	8.11E+01	3.20E+01

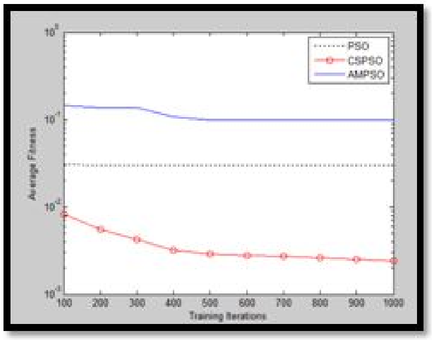


Figure (3.a)

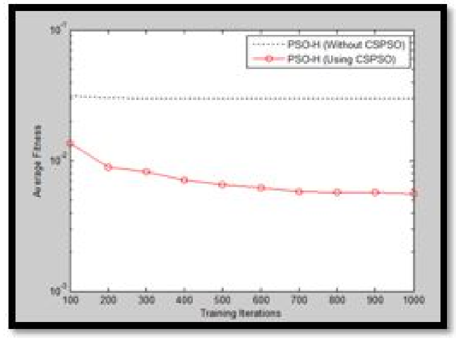


Figure (3.b)

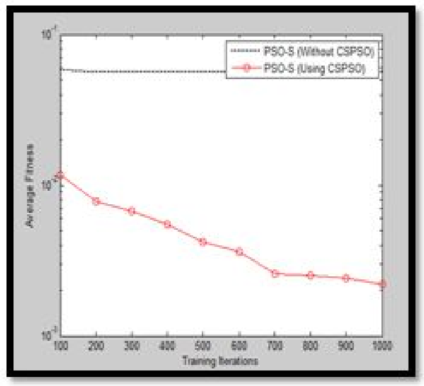


Fig. 3. Function F1 (3.a) using Chi-Square PSO (3.b) PSO-Halton(Using Chi-Square PSO) (3.c) PSO-Sobol(Using Chi-Square PSO)

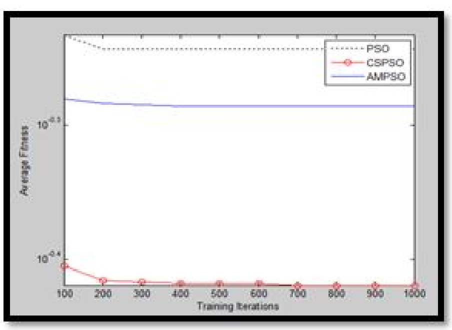


Figure (4.a)

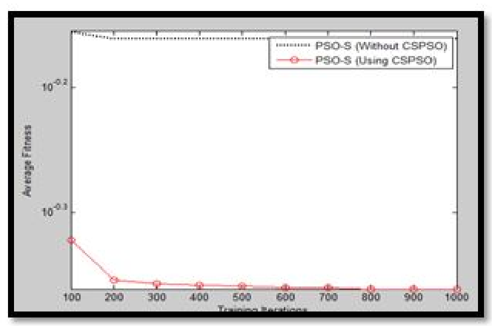


Figure (4.b)

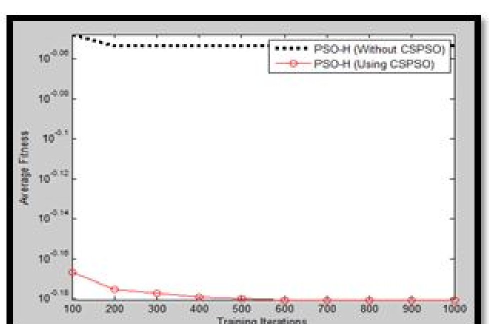
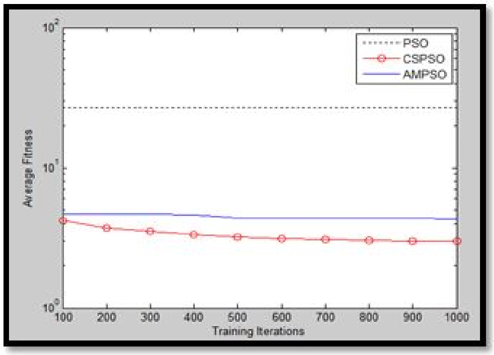


Fig. 4. Function F2 (4.a) using Chi-Square PSO (4.b) PSO-Halton(Using Chi-Square PSO) (4.c) PSO-Sobol(Using Chi-Square PSO))



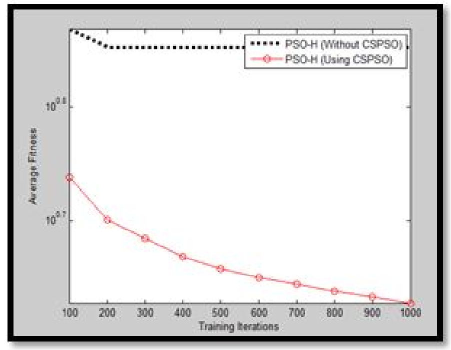


Figure (5.a) **Figure (5.b)**

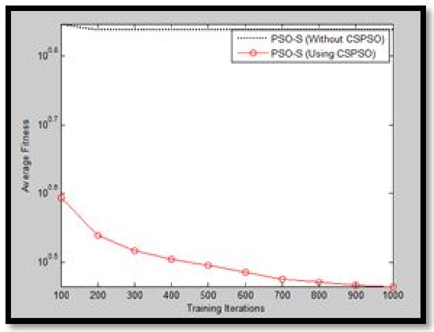
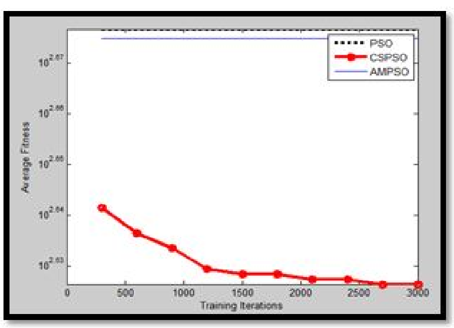


Fig. 5. Function F3 (5.a) using Chi-Square PSO (5.b) PSO-Halton(Using Chi-Square PSO) (5.c) PSO-Sobol(Using Chi-Square PSO)



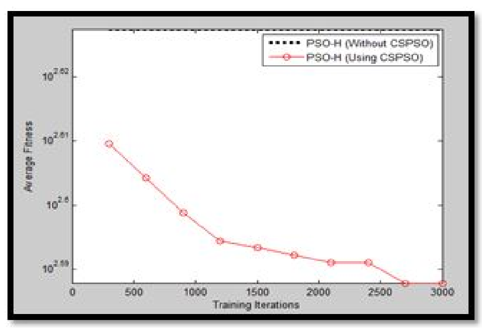


Figure (6.a) **Figure (6.b)**

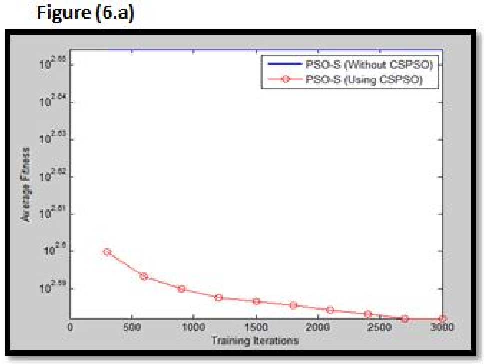
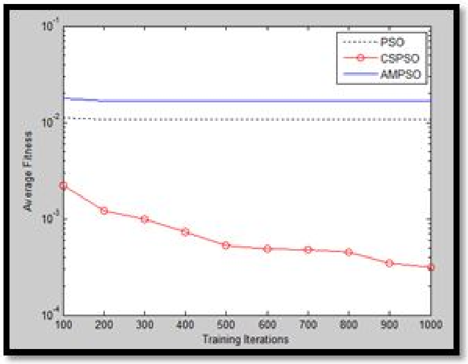


Figure (6.c)

Fig. 6. Function F4 (6.a) using Chi-Square PSO (6.b) PSO-Halton(Using Chi-Square PSO) (6.c) PSO-Sobol(Using Chi-Square PSO)



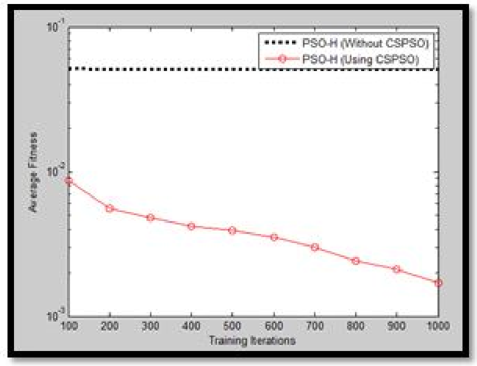


Figure (7.a)

Figure (7.b)

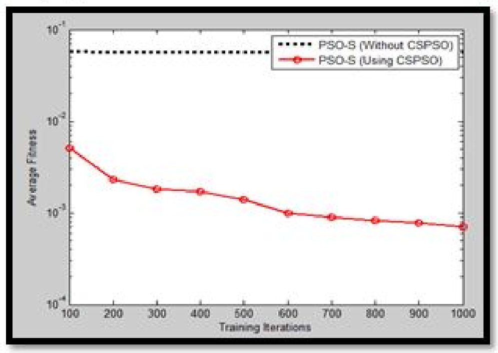
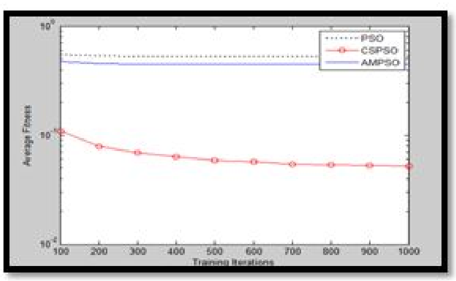


Figure (7.C)

Fig. 7. Function F7 (7.a) using Chi-Square PSO (7.b) PSO-Halton(Using Chi-Square PSO) (7.c) PSO-Sobol(Using Chi-Square PSO)



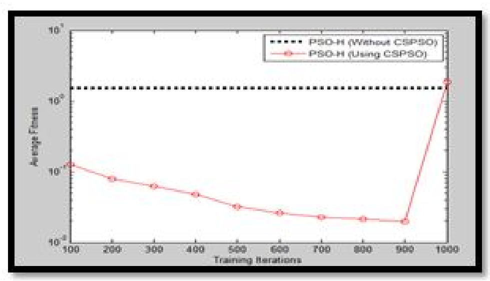


Figure (8.a)

Figure (8.b)

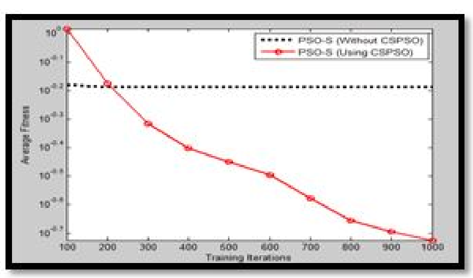


Figure (8.c)

Fig. 8. Function F8 (8.a) using Chi-Square PSO (8.b) PSO-Halton(Using Chi-Square PSO) (8.c) PSO-Sobol(Using Chi-Square PSO)

REFERENCES

[1] W. H. Bangyal, J. Ahmad, H. T. Rauf, and S. Pervaiz, "An improved bat algorithm based on novel initialization technique for global optimization problem," *International journal of advanced computer science and applications*, vol. 9, no. 7, pp. 158–166, 2018.

[2] ——, "An overview of mutation strategies in bat algorithm," *international journal of advanced computer science and applications*, vol. 9, no. 8, pp. 523–534, 2018.

[3] N. D. Jana, J. Sil, and S. Das, "Particle swarm optimization with population adaptation," in *Evolutionary Computation (CEC), 2014 IEEE Congress on*. IEEE, 2014, pp. 573–578.

[4] Y. Shi *et al.*, "Particle swarm optimization: developments, applications and resources," in *evolutionary computation, 2001. Proceedings of the 2001 Congress on*, vol. 1. IEEE, 2001, pp. 81–86.

[5] M. Clerc and J. Kennedy, "The particle swarm-explosion, stability, and convergence in a multidimensional complex space," *IEEE transactions on Evolutionary Computation*, vol. 6, no. 1, pp. 58–73, 2002.

[6] J. Kennedy and R. Mendes, "Neighborhood topologies in fully informed and best-of-neighborhood particle swarms," *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, vol. 36, no. 4, pp. 515–519, 2006.

[7] R. Mendes, "Population topologies and their influence in particle swarm performance," *PhD Final Dissertation, Departamento de Informática Escola de Engenharia Universidade do Minho*, 2004.

[8] Y.-T. Kao and E. Zahara, "A hybrid genetic algorithm and particle swarm optimization for multimodal functions," *Applied Soft Computing*, vol. 8, no. 2, pp. 849–857, 2008.

[9] A. Alfi and H. Modares, "System identification and control using adaptive particle swarm optimization," *Applied Mathematical Modelling*, vol. 35, no. 3, pp. 1210–1221, 2011.

[10] A. Ratnaweera, S. K. Halgamuge, and H. C. Watson, "Self-organizing hierarchical particle swarm optimizer with time-varying acceleration coefficients," *IEEE Transactions on evolutionary computation*, vol. 8, no. 3, pp. 240–255, 2004.

[11] E. S. Peer, F. van den Bergh, and A. P. Engelbrecht, "Using neighbourhoods with the guaranteed convergence pso," in *Swarm Intelligence Symposium, 2003. SIS'03. Proceedings of the 2003 IEEE*. IEEE, 2003, pp. 235–242.

[12] N. Higashi and H. Iba, "Particle swarm optimization with gaussian mutation," in *Swarm Intelligence Symposium, 2003. SIS'03. Proceedings of the 2003 IEEE*. IEEE, 2003, pp. 72–79.

[13] A. Stacey, M. Jancic, and I. Grundy, "Particle swarm optimization with mutation," in *Evolutionary Computation, 2003. CEC'03. The 2003 Congress on*, vol. 2. IEEE, 2003, pp. 1425–1430.

[14] M. Pant, R. Thangaraj, V. P. Singh, and A. Abraham, "Particle swarm optimization using sobol mutation," in *Emerging Trends in Engineering and Technology, 2008. ICETET'08. First International Conference on*. IEEE, 2008, pp. 367–372.

[15] A. E. Muñoz Zavala, A. Hernández Aguirre, E. R. Villa Diharce, and S. Botello Rionda, "Constrained optimization with an improved particle swarm optimization algorithm," *International Journal of Intelligent Computing and Cybernetics*, vol. 1, no. 3, pp. 425–453, 2008.

[16] D. Jia, L. Li, Y. Zhang, and X. Chen, "Particle swarm optimization combined with chaotic and gaussian mutation," in *Intelligent Control and Automation, 2006. WCICA 2006. The Sixth World Congress on*, vol. 1. IEEE, 2006, pp. 3281–3285.

[17] X. Wu, B. Cheng, J. Cao, and B. Cao, "Particle swarm optimization with normal cloud mutation," in *Intelligent Control and Automation, 2008. WCICA 2008. 7th World Congress on*. IEEE, 2008, pp. 2828–2832.

[18] J. Chen, Z. Ren, and X. Fan, "Particle swarm optimization with adaptive mutation and its application research in tuning of pid parameters," in *Systems and Control in Aerospace and Astronautics, 2006. ISSCAA 2006. 1st International Symposium on*. IEEE, 2006, pp. 5–pp.

[19] B. Liu, L. Wang, Y.-H. Jin, F. Tang, and D.-X. Huang, "Improved particle swarm optimization combined with chaos," *Chaos, Solitons & Fractals*, vol. 25, no. 5, pp. 1261–1271, 2005.

[20] M. Yang, H. Huang, and G. Xiao, "A novel dynamic particle swarm optimization algorithm based on chaotic mutation," in *Knowledge Discovery and Data Mining, 2009. WKDD 2009. Second International Workshop on*. IEEE, 2009, pp. 656–659.

[21] J. Liu, X. Fan, and Z. Qu, "An improved particle swarm optimization with mutation based on similarity," in *Natural Computation, 2007. ICNC 2007. Third International Conference on*, vol. 4. IEEE, 2007, pp. 824–828.

[22] B. Cai, Z. Li, D. Fu, J. Hu, P. Ou, and Q. Li, "Mutated fast convergent particle swarm optimization and convergence analysis," in *First International Conference on Intelligent Networks and Intelligent Systems*. IEEE, 2008, pp. 5–8.

[23] S. Sheikhpour and A. Mahani, "Particle swarm optimization with intelligent mutation for nonlinear mixed-integer reliability-redundancy allocation," *International Journal of Computational Intelligence and Applications*, vol. 16, no. 01, p. 1750003, 2017.

[24] K. Wang and F. Li, "A dynamic chaotic mutation based particle swarm optimization for dynamic optimization of biochemical process," in *Information Science and Control Engineering (ICISCE), 2017 4th International Conference on*. IEEE, 2017, pp. 788–791.

[25] R. Eberhart and J. Kennedy, "A new optimizer using particle swarm theory," in *Micro Machine and Human Science, 1995. MHS'95., Proceedings of the Sixth International Symposium on*. IEEE, 1995, pp. 39–43.

[26] Y. Shi and R. C. Eberhart, "Empirical study of particle swarm optimization," in *Evolutionary computation, 1999. CEC 99. Proceedings of the 1999 congress on*, vol. 3. IEEE, 1999, pp. 1945–1950.

[27] Y. Shi and R. Eberhart, "A modified particle swarm optimizer," in *Evolutionary Computation Proceedings, 1998. IEEE World Congress on Computational Intelligence., The 1998 IEEE International Conference on*. IEEE, 1998, pp. 69–73.

[28] M. Clerc, "The swarm and the queen: towards a deterministic and adaptive particle swarm optimization," in *Evolutionary Computation, 1999. CEC 99. Proceedings*

of the 1999 Congress on, vol. 3. IEEE, 1999, pp. 1951–1957.

[29] H. Fan, "Study on vmax of particle swarm optimization," in *Proc. of the Workshop on Particle Swarm Optimization, 2001*, 2001.

[30] R. C. Eberhart and Y. Shi, "Comparing inertia weights and constriction factors in particle swarm optimization," in *Evolutionary Computation, 2000. Proceedings of the 2000 Congress on*, vol. 1. IEEE, 2000, pp. 84–88.

[31] ——, "Comparison between genetic algorithms and particle swarm optimization," in *International conference on evolutionary programming*. Springer, 1998, pp. 611–616.

[32] M. R. Bonyadi and Z. Michalewicz, "Analysis of stability, local convergence, and transformation sensitivity of a variant of the particle swarm optimization algorithm," *IEEE Transactions on Evolutionary Computation*, vol. 20, no. 3, pp. 370–385, 2016.

[33] J. Kennedy and R. C. Eberhart, "A discrete binary version of the particle swarm algorithm," in *Systems, Man, and Cybernetics, 1997. Computational Cybernetics and Simulation., 1997 IEEE International Conference on*, vol. 5. IEEE, 1997, pp. 4104–4108.