Sparse based Particle Swarm Optimization

Lalit Kumar[a]*, Manish Pandey[b], Mitul Kumar Ahirwal[c]

[a] *PhD Scholar, Dept. of CSE, MANIT Bhopal, India*
[b]*Assistant Professor, Dept. of CSE, MANIT Bhopal, India*

Abstract

Particle Swarm Optimization (PSO) is the most famous metaheuristic algorithm for optimization, inspired from swarm of species. PSO can be used in various problems related to engineering and sciences. In this study, a sparse representation based PSO (Sparse-PSO) algorithm has been presented. Comparison of proposed Sparse-PSO with Standard-PSO has been done though evaluation over several standard benchmark objective functions. Our proposed Sparse-PSO method takes less computation time and provides better solution for almost all benchmark objective functions as compared to Standard-PSO method. Execution time reduction is the advantage gained through proposed Sparse-PSO.

Keywords: Particle Swarm Optimization, Sparse Representation, Swarm Intelligence

Nomenclature

Vel	velocity direction
Pos	position of particle
r	current iteration
m	particular dimension at *i*th particle
W	inertia weight whose value is 1
h_1 and h_2	random numbers in the interval (0, 1)
Ψ_1 and Ψ_2	positive acceleration constants
N	population of size
M	number of dimensions
R	number of iterations
P_b	personal best
G_b	global best

1. Introduction

Nature Inspired Algorithms (NIA) are the most prominent strategies developed from natural surroundings [1] and mostly used in optimization tasks. Swarm based Intelligence algorithm is one of the category of NIA which is based on the cooperative behavior of swarm members. Particle Swarm Optimization algorithm (PSO) belongs to swarm based metaheuristic algorithm and was proposed in 1995 by J. Kennedy and R. Eberhart [2], [3]. PSO is encouraged by the vibrant movement of insects, birds, fishes, etc. In the field of science and engineering, PSO has been utilized to tackle non-differentiable, non-linear, and multimodal optimization problems [4]-[7]. Drawbacks such as inadequate speed-up for reaching optimum point and unsatisfactory efficiency of PSO in some studies, motivate for further development and enhancement in PSO. PSO is a simple to-actualize algorithm and furthermore has less customizable boundaries than practically equivalent to algorithms. In recent years many improvements have been done in PSO for many application areas of optimization such as chaos based PSO [8] to improve convergence speed, diminishing population based PSO to meet the swam on the most favorable point [9], FCPSO based on balancing the diversity of location to achieve convergence [10], and many more. Nature inspired optimization techniques are also used in field of signal processing to optimize adaptive noise canceller [11], to improve the range of search space [12], to filter the noisy signals [13].

Basically, the execution of PSO starts with a randomly distributed particles (population of solutions) inside the search space. As the iterations continue, the particles move as general group towards a most favorable point [2]. To appraise the optimality of each solution (particle) fitness function is evaluated in each cycle after that updation mechanism is applied to update the location of each particle so that they reach to optimal point and helps in convergence. This process of execution is repeated many times on particles to converge at global optima. All the above steps take long computation time if fitness objective function is more complex and it is difficult to achieve significant improvement.

* Lalit Kumar
E-mail address: lalit212212@gmail.com

The main aim of this paper is to gain speed-up by reducing computation time and accomplish better efficiency with the help of proposed Sparse-PSO. The rest of the paper is organized as follows: a basic description of the Standard-PSO algorithm has been provided in Section II. The proposed Sparse-PSO algorithm has been explained in Section III. Section IV displays the experiment results for the proposed approach and compared it with results for Standard-PSO. Section V contains a summary and concluding remarks.

2. Standard Particle Swarm Optimization

Particle swarm optimization is very popular optimization algorithm as only few parameters are there in this algorithm. PSO consist of a group of particle know as swarm. PSO is an iterative method since many iterations of the procedure is to be done to achieve the optimal value. In PSO, firstly an initial population of particles is initialized by arbitrarily initializing the position and velocity vectors in the defined search space. Each particle has fitness value to check solution quality at each iteration. This fitness value is obtained through objective functions which may be of minimization or maximization type, depending upon problem. Two significant things in this algorithm needs to be noticed, personal best (Pb) solution accomplished up until now and the global best (Gb) particle which is the best solution among all individual best solutions achieved until now. The dynamic journey of the particle is controlled by its personal flying knowledge as well as the flying involvement of different particles (Pb and Gb) in the swarm. Each particle's velocity and position are updated by using equations (1) and (2) respectively.

$$Vel_i^m(r+1) = W \times Vel_i^m(r) + \Psi_1 h_1 * (P_{bi}^m(r) - Pos_i^m(r)) + \Psi_2 h_2 \times (G_b^m(r) - Pos_i^m(r)) \quad (1)$$

$$Pos_i^m(r+1) = Pos_i^m(r) + Vel_i^m(r+1) \quad (2)$$

First term in velocity updation equation dominate the influence of earlier used velocities on the recent velocity or makes the particle move is same direction with the same velocity [14]. Second term control the position of particle by returning it to previous position if the previous position is better than the current position of particle according to objective function. While third term controls the particle to follow the best particle in the swarm.

3. Proposed Sparse Particle Swarm Optimization

3.1. Proposed Initialization Process

In proposed Sparse Particle Swarm Optimization (Sparse-PSO), a particle is represented as a position vector of magnitude M, where M indicates the number of dimensions or variables. Our aim is to achieve the speed-up the computation time of algorithm. In this effort, the population of particles called swarm is arbitrarily generated within defined boundaries of size N but in the form of sparse matrix representation to achieve our aim and reaching to the optimum point rapidly. After initialization of sparse matrix based population, each particle is evaluated according to the fitness function. However, this procedure is performed in the initial phase of algorithm.

3.2. Proposed Algorithm

- In proposed method, first of all swarm of size N is initialized similar to sparse matrix as deliberated in Segment *III-A* aswell as in the flow of proposed approach described below.
- Next, test functions are used to evaluate the optimal solution for every particle of swarm set. At this point, quality of solution depends on the nature of test function that can be minimization or maximization problem.
- Afterwards, personal best and global best are selected and revised repeatedly in accordance with the quality of particles.

Inputs: Initialize parameters (N, M, R, Ψ1, Ψ2, Density)
Output: global best solution and respective position vector;

Proposed Algorithm (Sparse-PSO):

```
1) Initialize sparse based swarm of size $N$ arbitrarily in search space by using a method 'sprand($N$, $M$, Density)' whichgenerates
     an arbitrary $N$ by $M$ sparse matrix in which number of non-zero entries are around $N*M*$Density;
2) For every particle
3) Compute fitness of particle using objective function;
4) Set personal best ($P_b$);
5) End of for
6) Set global best ($G_b$);
7) While number of iterations are not met
8) For each particle
```

```
9) Revise the velocity and location of particle by equations (1) and (2);
10) Determine fitness of new revised particle;
11) Update personal best;
12) End of for
13) Hence, update global best;
14) End of while loop
```

4. Experiment and Results

4.1. Benchmark Functions

The efficacy of the proposed Sparse-PSO algorithm is verified using several experiments conducted on fourteen benchmark functions taken from [15] with different characteristics. The functions used to test our proposed method have been listed in Appendix *A.* Table 1 represents the bound ranges and global minimum value of benchmark functions where dimension, problem domain size, and optimal solution are denoted by M, $Lb \leq x_i \leq Ub$, and $f(X^*)$ respectively. Out of these functions, the Exponential function *(f_1)*, the Sphere function *(f_2)*, and the Step function *(f_3)* are unimodal in nature whereas the Ackley function *(f_4)*, the Periodic function *(f_5)*, the Quartic function *(f_6)*, and the Qing function *(f_7)* are multimodal in nature.

Table 1: -Detail of Benchmark Functions

Name of Functions	Range	Global Minima
Exponential	$-1 \leq x_i \leq 1$	$f_1(X^*) = -1$
Sphere	$-100 \leq x_i \leq 100$	$f_2(X^*) = 0$
Step	$-100 \leq x_i \leq 100$	$f_3(X^*) = 0$
Ackley	$-32 \leq x_i \leq 32$	$f_4(X^*) = 0$
Periodic	$-10 \leq x_i \leq 10$	$f_5(X^*) = 0.9$
Quartic	$-1.28 \leq x_i \leq 1.28$	$f_6(X^*) = 0 + noise$
Qing	$-500 \leq x_i \leq 500$	$f_7(X^*) = 0$

4.2. Parameter Setup

The basic parameter settings are defined in Table 2. All experimental results of the algorithms namely Standard-PSO and Sparse-PSO are collected from 25 independent runs, each involving 2000 iterations.

Table 2: - Parameter Setting for Algorithms

Parameter	PSO	Sparse-PSO
Population Size (*N*)	150	150
Dimension (*M*)	30	30
Acceleration factor ($\Psi 1$ and $\Psi 2$)	1.5	1.5
Density	-	0.51
R (total number of iterations)	2000	2000
Max (total number of runs)	25	25

4.3. Experiment Results

In this work, swarm size is taken as 150 and dimension as 30 for all the functions. Both the mentioned algorithms are implemented independently 25 times for 2000 iterations on each problem of benchmark function. The data achieved from 25 independent runs are given in Tables 3 and 4. Table 3 which shows a correlation of the computation time as mean time, best time, and worst time taken by both the algorithms during each run. By analyzing the outcomes of Table 3, it very well may be inferred that our proposed Sparse-PSO algorithm has a far superior execution time compared with the Standard-PSO algorithm. Also, Table 4 shows the performance achieved by Sparse-PSO and the Standard-PSO in the terms of mean (average of global values), best (minimum in global values), worst (maximum in optimum values), and std. (standard deviation between global values).

Fig. 1 represents the mean of best computation time for all the functions used in this work and also concludes that proposed Sparse-PSO takes less execution time overall with respect to Standard-PSO. Convergence graphs for each function are designed in Fig. 2 where the horizontal axis shows the number of iterations and the vertical axis represents the costs of each benchmark function for all iterations. It can be seen from Fig. 2(a) that for Exponential function start point of proposed method is better than Standard-PSO. Next, both the methods converge to optimal point in almost same manner. Fig. 2(b) represents the convergence graph for Sphere function which shows that start point as well as convergence curve of proposed method is better than Standard-PSO. Convergence graph for Step function are almost equal for both the algorithms is reported in Fig. 2(c). Fig. 2(d) demonstrates that in Sparse-PSO method the graph for Ackley function reaches global optimum point while in Standard-PSO approach the graph converges earlier before reaching the global optimum point. Sparse-PSO performs well for Periodic function presented in Fig. 2(e) as it converges firstly than Standard-PSO. Figures 2(f) and 2(g) determine that when the number of iterations is low the Sparse-PSO method enhances the results as compared to the Standard-PSO and later on both methods are showing similar results when number of iteration is high. It can be concluded from Fig. 2 that the proposed Sparse-PSO algorithm starts with better function value compared to Standard-PSO algorithm for all the functions and reaches to the global optimum.

Table 3: - Comparative Analysis of Standard-PSO and Sparse-PSO in terms of Computation Time (in Sec.)

Functions		**Standard-PSO Time (Sec.)**	**Sparse-PSO Time (Sec.)**
Exponential	Mean	16.2440	**12.4108**
	Best	14.2552	**12.0264**
	Worst	18.2122	**15.6286**
Sphere	Mean	17.9444	**12.0551**
	Best	15.3946	**11.6846**
	Worst	22.3657	**15.0164**
Step	Mean	13.1506	**11.8242**
	Best	12.7770	**11.2875**
	Worst	13.7343	**13.4691**
Ackley	Mean	20.1344	**13.2763**
	Best	18.2952	**11.9703**
	Worst	22.0618	23.0440
Periodic	Mean	15.5580	**11.9585**
	Best	12.9640	**11.4463**
	Worst	19.0293	**15.5228**
Quartic	Mean	15.6129	**13.4802**
	Best	14.6902	**13.3035**
	Worst	16.6461	**14.1285**
Qing	Mean	17.4904	**11.8417**
	Best	12.8546	**11.5059**
	Worst	20.3011	**12.6486**

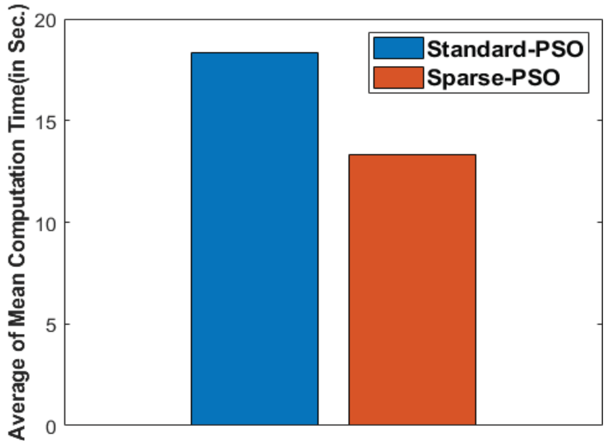
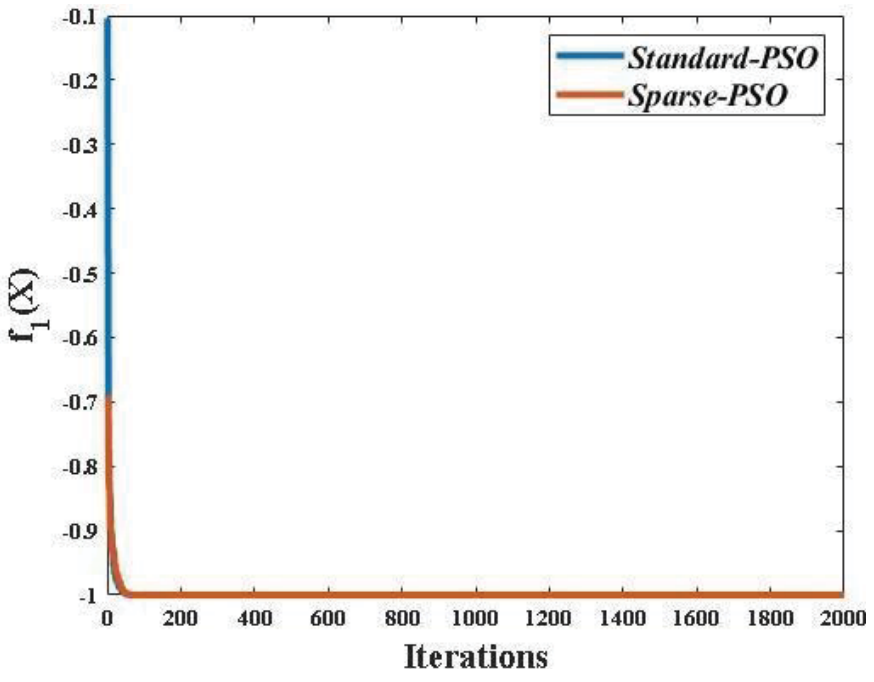


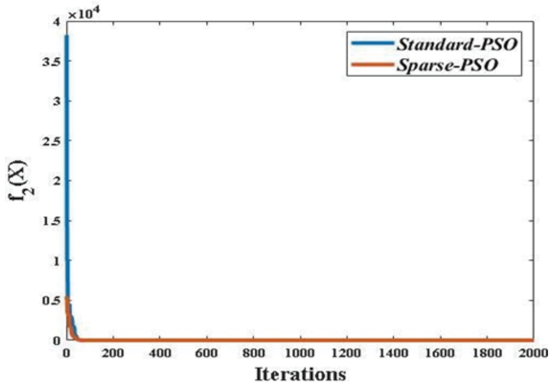
Fig.1. Overall average computation time for all functions

Table 4: - Comparative Analysis of Standard-PSO and Sparse-PSO in terms of performance

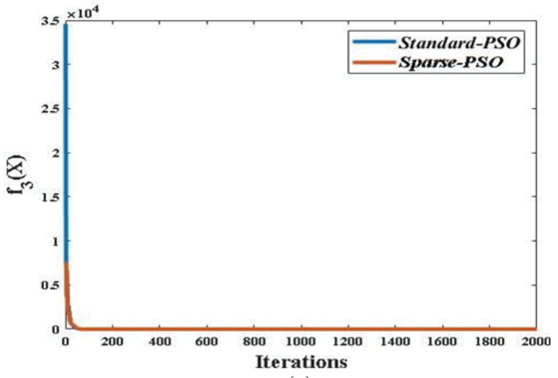
Functions		**Standard-PSO Final *f(X)* value**	**Sparse-PSO Final *f(X)* value**
Exponential	Mean	-1.0000	-1.0000
	Best	-1.0000	-1.0000
	Worst	-1.0000	-1.0000
	Std.	$8.2927e^{-16}$	$5.8922\ e^{-16}$
Sphere	Mean	$1.0302\ e^{-20}$	$5.3397\ e^{-22}$
	Best	$9.0677\ e^{-30}$	$6.9299\ e^{-33}$
	Worst	$2.3039\ e^{-19}$	$1.1973\ e^{-20}$
	Std.	$4.5988\ e^{-20}$	$2.3922\ e^{-21}$
Step	Mean	1.1600	1.2000
	Best	0	0
	Worst	5	5
	Std.	1.2138	1.1180
Ackley	Mean	0.8266	0.6943
	Best	$4.1478\ e^{-13}$	$6.2172\ e^{-13}$
	Worst	2.4083	2.3168
	Std.	0.7686	0.7471
Periodic	Mean	1.0000	1.0000
	Best	1.0000	1.0000
	Worst	1.0000	1.0000
	Std.	$9.0876\ e^{-16}$	$7.8382\ e^{-15}$
Quartic	Mean	0.0069	0.0054
	Best	0.0020	0.0018
	Worst	0.0187	0.0200
	Std.	0.0038	0.0040
Qing	Mean	$2.6747\ e^{-17}$	$6.4860\ e^{-17}$
	Best	$2.7593\ e^{-26}$	$3.1645\ e^{-26}$
	Worst	$4.9303\ e^{-16}$	$1.1821\ e^{-15}$
	Std.	$1.0006\ e^{-16}$	$2.4831\ e^{-16}$



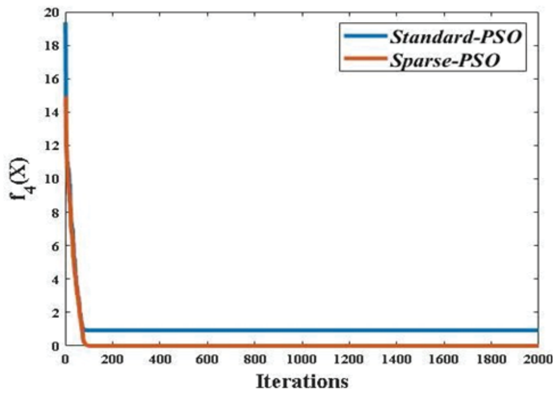
(a)


Standard-PSO
Sparse-PSO
$f_2(X)$
$\times 10^4$
Iterations

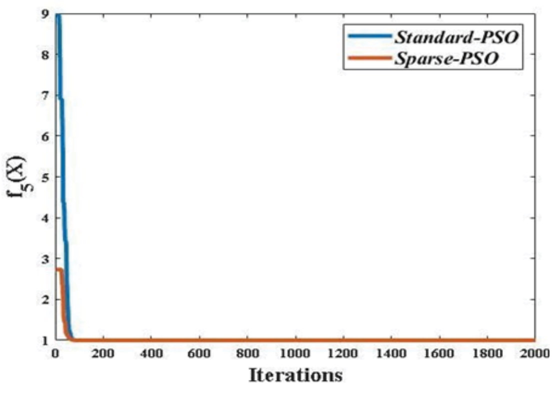
(b)


Standard-PSO
Sparse-PSO
$f_3(X)$
$\times 10^4$
Iterations

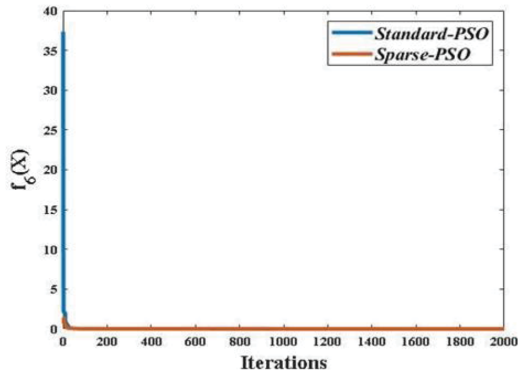
(c)


Standard-PSO
Sparse-PSO
$f_4(X)$
Iterations

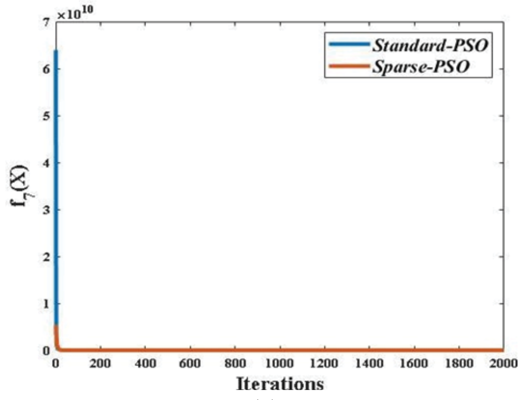
(d)


Standard-PSO
Sparse-PSO
$f_5(X)$
Iterations

(e)



(f)



(g)

Fig.2. Convergence graphs of Sparse-PSO and Standard-PSO for seven test functions (a) Exponential, (b) Sphere, (c) Step, (d) Ackley, (e) Periodic, (f) Quartic, (g) Qing

5. Conclusion and Future Directions

A new initialization technique on the basis of sparse representation has been proposed for PSO in this paper. The main objective of the proposed Sparse-PSO algorithm is to reduce the computation time. Seven benchmark functions as listed in Segment IV-A have been used to test the proposed method, and the outcomes of the same are compared with the standard approach. It is clear from results that the Sparse-PSO takes less computation time as compared to the Standard-PSO. Sparse- PSO over performs Standard-PSO in terms of execution time with an average speed-up of 24%, 33%, 10%, 34%, 23%, 13%, and 32% for the exponential function, Sphere function, Step function, Ackley function, Periodic function, Quartic function, and Qing function respectively. The proposed method achieves the global best value with low computation time compared to Standard-PSO. The future direction of this work would be to achieve more improvement by correcting and updating the used parameters and performing this method on more complex functions.

Appendix A: Benchmark Functions

$$f_1(X)=-\exp\left(-0.5\sum_{i=1}^{M}x_i^2\right)$$

$$f_2(X)=\sum_{i=1}^{M}x_i^2$$

$$f_3(X)=\sum_{i=1}^{M}\left(\left\lfloor\left(x_i+0.5\right)^2\right\rfloor\right)$$

$$f_4(X)=\exp(1)-20*\exp\left(-0.2\sqrt{\frac{1}{M}\sum_{i=1}^{M}x_i^2}\right)-\exp\left(\frac{1}{M}\sum_{i=1}^{M}\cos\left(2\Pi x_i\right)\right)+20 \quad f_5(X)=1+\sum_{i=1}^{M}\sin^2(x_i)-0.1e^{\left(\sum_{i=1}^{M}x_i^2\right)}$$

$$f_6(X)=\sum_{i=1}^{M}i x_i^4+random[0,1)$$

$$f_7(X)=\sum_{i=1}^{M}\left(x_i^2-i\right)^2$$

References

1. Kennedy, “Particle swarm optimization,” Encyclopedia of machine learning 2011, pp. 760-766, Springer US, 2011.
2. F. Marini, and B. Walczak, “Particle swarm optimization (PSO). A tutorial,” Chemometrics and Intelligent Laboratory Systems, 2015 Dec 15, 149:153- 65.
3. Y. Shi, “Particle swarm optimization: developments, applications and resources,” Proceedings of the 2001 congress on evolutionary computation Vol. 1, pp. 81-86, IEEE, 2001.
4. R. Poli, “Analysis of the publications on the applications of particle swarm optimisation,” Journal of Artificial Evolution and Applications, 2008.
5. K. R Harrison, A. P. Engelbrecht, and B. M. Ombuki-Berman, “Self-adaptive particle swarm optimization: a review and analysis of convergence,” Swarm Intelligence, 12(3), 187-226, 2018.
6. Z. Liu, J. Lu, and P. Zhu, “Lightweight design of automotive composite bumper system using modified particle swarm optimizer,” Composite Structures, 140, 630-643, 2016.
7. D. Gao, X. Li, and H. Chen, “Application of improved particle swarm optimization in vehicle crashworthiness,” Mathematical problems in Engineering, 2019.
8. B. Soudan, and M. Saad, “An evolutionary dynamic population size PSO implementation,” 3rd International Conference on Information and Communication Technologies: From Theory to Applications, pp. 1-5, IEEE, 2008.
9. A. Sahu, S. K. Panigrahi, and S. Pattnaik, “Fast convergence particle swarm optimization for functions optimization,” Procedia Technology, 4, 319-324, 2012.
10. M. K. Ahirwal, A. Kumar, and G. K. Singh, “Study of ABC and PSO algorithms as optimised adaptive noise canceller for EEG/ERP,” International Journal of Bio-Inspired Computation, 8(3), 170-183, 2016.
11. M. K. Ahirwal, A. Kumar, and G. K. Singh, “Improved range selection method for evolutionary algorithm based adaptive filtering of EEG/ERP signals,” Neurocomputing, 144, 282-294, 2014.
12. M. K. Ahirwal, A. Kumar, and G. K. Singh, “Sub-band adaptive filtering method for electroencephalography/event related potential signal using nature inspired optimization techniques,” IET Science, Measurement & Technology, 9(8), 987-997, 2015.
13. Y. Shi, and R. Eberhart, “A modified particle swarm optimizer,” International conference on evolutionary computation proceedings. IEEE world congress on computational intelligence, Cat. No. 98TH8360, pp. 69-73, IEEE, 1998.
14. J. Momin, and X. S. Yang, “A literature survey of benchmark functions for global optimization problems,” Journal of Mathematical Modelling and Numerical Optimization, 4(2), 150-194, 2013.