International Journal of
Applied Mathematics,
Electronics and Computers
ISSN:2147-8228 www.atscience.org/IJAMEC Original Research Paper

Particle Swarm Optimization for Continuous Function Optimization Problems

Muhlis OZDEMIR*[1]

Accepted : 08/09/2017 Published: 30/09/2017 **DOI: 10.18100/ijamec.2017331879**

Abstract In this paper, particle swarm optimization is proposed for finding the global minimum of continuous functions and tested on benchmark problems. Particle swarm optimization applied on 21 benchmark test functions, and its solutions are compared to those former proposed approaches: ant colony optimization, a heuristic random optimization, the discrete filled function algorithm, an adaptive random search, dynamic random search technique and random selection walk technique. The implementation of the PSO on several test problems is reported with satisfactory numerical results when compared to previously proposed heuristic techniques. PSO is proved to be successful approach to solve continuous optimization problems.

Keywords: *continuous function optimization, global minimum, heuristic techniques, particle swarm optimization*

1. Introduction

The main goal of function optimization is to find a global solution of an objective function throughout the iterations. The swarm intelligent optimization algorithms are simulate the nature, mainly animals. Particle swarm optimization(PSO) proposed in 1995 by Eberhart and Kennedy and based on defining the social behaviours of the living beings.[1]. In 2002, Clerc and Kennedy added constriction coefficients and ameliorate the PSO much better[2]. Due to advancement of computer technologies in the computer science area, continuous function optimizations have become popular among the researchers. There are abundant test functions used in the literature to measure the proposed algorithms' performance by the scientists. Many heuristic methods and techniques are proposed and applied for solving these continuous functions like discrete filled function[3], dynamic random search technique[4], ant colony optimization[5], a heuristic random optimization[6], an adaptive random search technique[7], random selection walk[8], fruit fly optimization[9], biogeography-based optimization algorithm[10], flower pollination algorithm[11], continuous action-set reinforcement learning automata model[12], artificial bee colony algorithm[13], genetic algorithm[14] respectively. PSO is proved to be successful approach to solve continuous optimization problems. All the selected benchmark problems are minimization problems in this work. In this paper PSO applied on 21 benchmark test functions and were used to compare with the results of ant colony optimization(ACO), a heuristic random optimization(HRO), the discrete filled function algorithm, an adaptive random search technique(ARSET), dynamic random search technique(DRASET) and random selection walk(RSW) technique. PSO scrutinizes reasonable quality solutions much rapidly than other swarm based evolutionary algorithms. In this paper PSO's performance and robustness are shown for the aforementioned techniques. The author's findings show that for 21 benchmark problems PSO's results are quite competitive. The rest of the paper organized as follows; section two covers the information of proposed PSO algorithm, function optimization problems are given in section three and finally conclusion stated in section four.

[1]*Istanbul University, School of Business, Department of Quantitative Methods, Avcılar Campus 34320 Istanbul TURKEY*
** Corresponding Author: Email: muhlisozdemir@istanbul.edu.tr*
Orcid ID: http://orcid.org/0000-0002-4921-8209

2. The Proposed PSO Algorithm

The PSO became a very popular evolutionary algorithm and successfully applied for a wide range of continuous optimization functions. The algorithm exploits the solution space by improving the trajectories as moving particles in multidimensional solution space. Population of PSO consists of personal positions called particles, denoted $\vec{X_i}$. Each particle has velocities $\vec{V_i}$ and a cost function evaluated by using the particle's position. Positions and velocities are adjusted and cost function evaluated as the particle moves at each time step($\boldsymbol{Min\ f(x)}$, $\boldsymbol{x_n = [x_1, x_2, x_3, \dots, x_n]}$). $\boldsymbol{n}$ represents the number of decision variables. $\boldsymbol{x_n \in [LB, UB]}$, $\boldsymbol{n = 1, 2, \dots, n}$. $\boldsymbol{LB}$ and $\boldsymbol{UB}$ are the lower and upper bounds for the variable $\boldsymbol{x_n}$ respectively. When the particle moves better position than any found formerly, personal best positions explored so far are stored in $\vec{P_i}$. The difference between personal best position and particle current position is added to velocity. Thus velocities are used to move particles better coordinates.

$$\boldsymbol{V_{ij}(t+1) = wV_{ij}(t) + c_1 r_{1j}(t)\big(P_{ij}(t) - X_{ij}(t)\big) + c_2 r_{2j}(t)\big(g_j(t) - X_{ij}(t)\big)} \quad (1)$$

Velocity consists of inertia term, cognitive component and social component as stated equation 2. Where $\boldsymbol{w}$ represents the inertia weight, $\boldsymbol{r_{1j}}$ and $\boldsymbol{r_{2j}}$ are uniformly distributed random numbers between the range of 0 and 1. $\boldsymbol{c_1}$ and $\boldsymbol{c_2}$ are constants, called cognitive and social scaling parameters; $\boldsymbol{V_{ij}}$ the velocity of the particle, $\boldsymbol{P_{ij}}$ the personal best fitness value of the particle and $\boldsymbol{g_j}$ represents the best position and the global best in the whole population.

$$X_{ij}(t+1) = X_{ij}(t) + V_{ij}(t+1) \quad (2)$$

Each particle represents a position and a potential solution of the search space. As stated eq. 3 X_{ij} represents the position of the particle and is updated by the $V_{ij}(t+1)$ in eq. (2) throughout the algorithm.

As stated before Clerc and Kennedy added constriction coefficients in 2002[2], and therefore eq. 2 reorganized and used in this work as follows;

$$V_{ij}(t+1) = \chi\left[V_{ij}(t) + \phi_1\left(P_{ij}(t) - X_{ij}(t)\right) + \phi_2\left(g_j(t) - X_{ij}(t)\right)\right] \quad (3)$$

$$\chi = \frac{2\kappa}{\left|2-\phi-\sqrt{\phi^2-4\phi}\right|} \quad (4)$$

Where

$$\phi = \phi_1 + \phi_2,$$
$$\phi_1 = c_1 r_1,$$
$$\phi_2 = c_2 r_2,$$
$$\phi \geq 4 \ \ and \ \ \kappa \in [0, 1]$$

Thus

$$w = \chi, \qquad c_1 = \chi\phi_1, \qquad c_2 = \chi\phi_2$$

Eq. 2 can be used with these parameters stated above. PSO's Pseudocode as follows;

Step 1
Initialization
Do
Set n, LB, UB, $Iter_{max}$, ϕ_1 , ϕ_2, κ and χ
Generate population(PS)
Assign each particle random positions
***for i** = 1 to population size*
  ***if** fitness($\vec{X_i}$)<fitness($\vec{P_i}$)*
    $\vec{P_i}=\vec{X_i}$
    g=min($\vec{P_i}$)
  end
Next i
Until termination criterion is met

Step 2
Do
***for i**=1 to $Iter_{max}$*
  ***for j** = 1 to PS*
    $V_{ij}(t+1) = wV_{ij}(t) + c_1 r_{1j}(t)\left(P_{ij}(t) - X_{ij}(t)\right) + c_2 r_{2j}(t)\left(g_j(t) - X_{ij}(t)\right)$
    $X_{ij}(t+1) = X_{ij}(t) + V_{ij}(t+1)$
      ***if** fitness($\vec{X_{ij}}$)<fitness($\vec{P_{ij}}$)*
        $\vec{P_i}=\vec{X_i}$
        g_j=min($\vec{P_{ij}}$)
      end
  Next j
Next i
Until termination criterion is met

3. Benchmark Function Optimization Problems

In this section, the author compare the presented PSO with benchmark problems which have already been studied by numerous researchers. The algorithm coded in Matlab and is run Intel Core i7, 2.7 GHz with 16.00 GB Ram in macOS Sierra operating system. All figures in this paper were generated by using Matlab. The results are compared to those obtained from previous studies, which include ARSET[7], ACO[5], HRO[6], RSW[8], discrete filled function[3] and DRASET[4] algorithms. For the detail of the compared techniques, the readers can refer [3-8]. Except PSO, all comparative results for the benchmark problems are derived from CURA's previous work[8]. Throughout the iterations, performance of PSO for first 3 problems are shown in figures. Also third and sixth problem objective functions are shown in figures within the range [*-10*, *10*].

3.1. Benchmark Problem 1

This function has one variable and two minima. The local minimum is at *x=0*, furthermore function's global minimum is at *x=3*.

$$f(x) = \begin{cases} x^2, & \text{if} \quad x \leq 1, \\ (x-3)^2 - 3, & \text{if} \quad x > 1. \end{cases} \quad \textbf{(5)}$$

Differ from the RSW and PSO, the initial point were set to 0.5 for other three techniques. RSW were set the initial points *23.9342* and *14.0356*. For this problem PSO's $\boldsymbol{LB}$ and $\boldsymbol{UB}$ were set to *-50* and *50*. All results compared for the benchmark problem 1 are shown in table 1.

Table 1: Results for the Problem 1

Algorithms	***x***	***f(x)***	***Epoch Number***
HRO	3.000324	-2.9999998	1,000
ARSET	3	-3	1,000
ACO	3	-3	500
RSW($x^{initial} = 23.9342$)	2.999581	-2.9999998	500
RSW($x^{initial} = 14.0356$)	3	-3	1,000
PSO	3	-3	50

For instance PSO's performance for Problem 1 throughout the iterations is shown in figure 1.

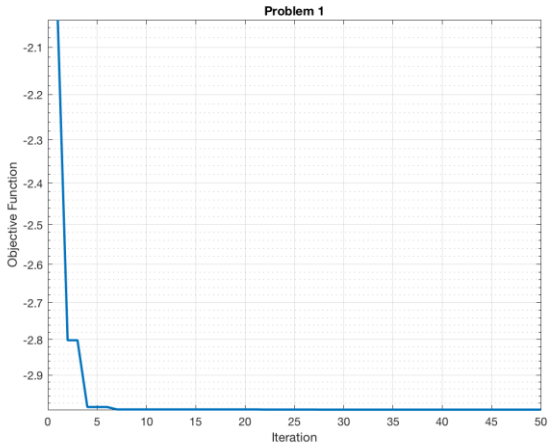

Figure 1: Performance of PSO for Problem 1

3.2. Benchmark Problem 2

Second function has one variable and its minimum at *x=0*.

$$f(x) = \begin{cases} \left[x * sin\left(\frac{1}{x}\right)\right]^4 + \left[x * cos\left(\frac{1}{x}\right)\right]^4, & \text{if} \quad x \neq 0, \\ 0, & \text{otherwise}. \end{cases} \quad \textbf{(6)}$$

ARSET and ACO were set the initial point to 1. All four techniques have three different epoch number results. RSW were set the initial points *40.1959*, *45.287* and 29.9729. Also PSO was run for *1000*, *3000*, *5000* epoch numbers. The results are given in table 2.

Table 2: Results for the Problem 2

Algorithms	***x***	***f(x)***	***Epoch Number***
ARSET	1.90E-06	6.58E-024	10,000
ACO	3.36E-10	8.16E-039	10,000
RSW($x^{initial} = 40.1959$)	-2.55E-74	1.47E-295	10,000
PSO	1.10E-53	1.11E-212	1,000
ARSET	4.39E-08	1.85E-30	30,000
ACO	-1.57E-11	5.47E-44	30,000
RSW($x^{initial} = 45.287$)	8.17E-82	0	30,000
PSO	-5.34E-82	0	3,000
ARSET	-2.53E-11	2.21E-43	50,000
ACO	7.79E-12	1.40E-45	50,000
RSW($x^{initial} = 29.9729$)	-1.34E-81	0	50,000
PSO	3.91E-102	0	5,000

Also PSO's performance for problem 2 shown in figure 2. It is obvious that although selected epoch number 5,000, PSO achieved its best solution throughout the iteration number approximately 1700.

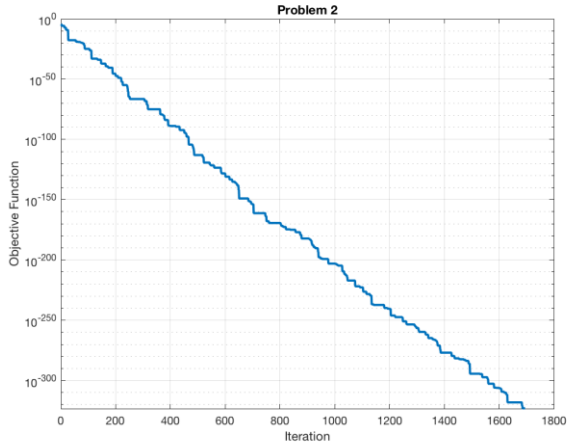

Figure 2: PSO's performance for 5,000 epoch number

3.3. Benchmark Problem 3

Problem 3 is 8th degree polynomial and has two variables, shown in eq. 7. The minimum point of this function *x=3*, *y=3* and *f(x, y)=0*.

$$f(x,y) = \frac{(x-3)^8}{1+(x-3)^8} + \frac{(y-3)^4}{1+(y-3)^4} \qquad (7)$$

It is stated that ARSET, ACO and RSW were run for *E=10,000, 30,000* and *50,000*. ARSET and ACO both took the initial point *{0,0}* and RSW was run for three different initial point for x and y i.e. $\begin{pmatrix} x^{initial} = 11.8192 \\ y^{initial} = -27.2218 \end{pmatrix}$, $\begin{pmatrix} x^{initial} = 43.0038 \\ y^{initial} = -41.6007 \end{pmatrix}$ and $\begin{pmatrix} x^{initial} = 25.5261 \\ y^{initial} = -25.8986 \end{pmatrix}$.
A visual for the objective function of benchmark problem 3 can be seen in figure 3.

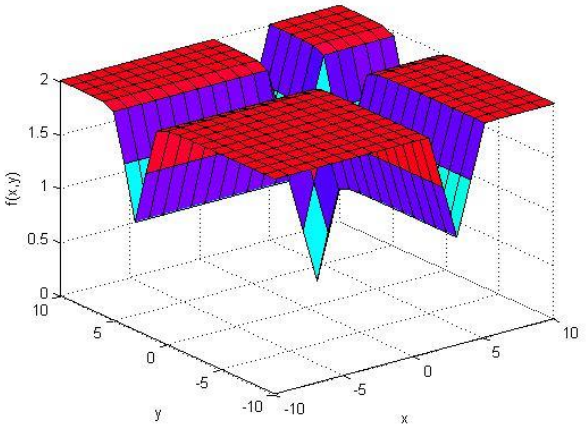

Figure 3: Visual for the Objective Function of Benchmark Problem 3

PSO was run for 1,000, 3,000 and 5,000 epoch numbers. The results are given in table 3. Also PSO's performance for problem 3 is shown in figure 4. It is obvious that although epoch number were selected for 5,000, PSO achieved its best solution throughout the iteration number approximately 450.

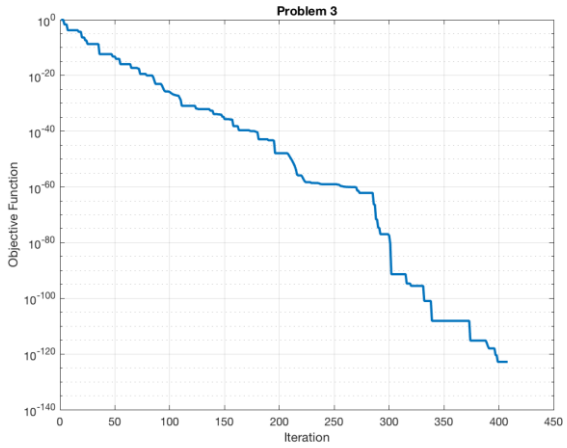

Figure 4: performance of PSO for problem 3

Table 3: Results for the Problem 3

Algorithms	***x***	***y***	***f(x, y)***	***Epoch Number***
ARSET	3.0157	2.9999	3.71E-15	10,000
ACO	3x2066E-09	3x2384E-09	2.62E-21	10,000
RSW$\begin{pmatrix} x^{initial} = 11.8192 \\ y^{initial} = -27.2218 \end{pmatrix}$	2.9991	2.9999	4.17E-25	10,000
PSO	2.9993	2.9999	3.64E-26	1,000
ARSET	3.0072	3	7.32E-18	30,000
ACO	3	3	0	30,000
RSW$\begin{pmatrix} x^{initial} = 43.0038 \\ y^{initial} = -41.6007 \end{pmatrix}$	3.0005	3	6.73E-27	30,000
PSO	3,0000	3	1.08E-84	3,000
ARSET	3.0015	3	5.04E-23	50,000
ACO	3	3	0	50,000
RSW$\begin{pmatrix} x^{initial} = 25.5261 \\ y^{initial} = -25.8986 \end{pmatrix}$	2.9996	3	3.43E-28	50,000
PSO	3	3	0	5,000

3.4. Benchmark Problem 4

This benchmark problem is known as Rosenbrock's Banana Function in the literature. ARSET, ACO and PSO searched for the minimum function value within the solution range [*0, 6*]. The minimum of the function is at *x=1, y=1* and *f(x, y)=0.*

$$f(x,y) = 100(x - y^2)^2 + (1 - x)^2 \quad (8)$$

It is stated that ARSET was run for the initial point for [*-1.9, 2*]. Also RSW $\binom{x^{initial} = 24.7355}{y^{initial} = 42.3291}$, $\binom{x^{initial} = 43.3763}{y^{initial} = 23.2853}$ and $\binom{x^{initial} = 43.8923}{y^{initial} = 46.455}$. PSO initialized totally random points for this problem.

Table 4: Results for the Problem 4

Algorithms	*x*	*y*	*f(x, y)*	*Epoch Number*
ARSET	0.99401	0.997	3.58E-05	10,000
ACO	1.00021	1.00004	1.73E-06	10,000
RSW$\binom{x^{initial} = 24.7355}{y^{initial} = 42.3291}$	0.99999	0.99999	2.27E-27	10,000
PSO	1	1	0	1,000
ARSET	1.0001	1.0001	2.03E-08	30,000
ACO	1	1	5.68E-12	30,000
RSW$\binom{x^{initial} = 43.3763}{y^{initial} = 23.2853}$	1	1	5.21E-28	30,000
PSO	1	1	0	3,000
ARSET	1	1	4.02E-16	50,000
ACO	1	1	0	50,000
RSW$\binom{x^{initial} = 43.8923}{y^{initial} = 46.455}$	1	1	1.97E-31	50,000
PSO	1	1	0	5,000

3.5. Benchmark Problem 5

Problem 5 is a different form of problem 4 as seen in eq. (9). Youngjian and Yumei tested their algorithm on this function by using discrete filled function algorithm[3].

$$f(x) = \sum_{i=1}^{N-1} 100\left(x_{i+1} - x_i^2\right)^2 + (1 - x_i)^2 \quad (9)$$

Youngjian and Yumei stated that they set the initial point for this problem [*5, 5*], [*-5, -5*], $x \in [-5\ -5]$ *and* $y \in [5\ 5]$, $x \in [5\ 5]$ *and* $y \in [-5\ -5]$. This problem solved with PSO in the range of [*-5, 5*]. Three different dimensional(N=10, 25 and 50) results for RSW and PSO can be seen in Table 5.

Table 5: Results for the Problem 5

N	*RSW*	*PSO*
10	8,20E-04	1.14E-08
25	4,84E-03	5.17E-06
50	1,58E-02	1.78E-05

3.6. Benchmark Problem 6

Benchmark problem 6 is given in eq. (10) where the goal is to minimize for the range of [*-10, 10*]. ACO, ARSET took initial point *x=9, y=9.* RSW took three different initial point for *x* and *y* as RSW$\binom{x^{initial} = 42.2649}{y^{initial} = -31.5613}$, RSW$\binom{x^{initial} = 49.8498}{y^{initial} = 45.5905}$ and RSW$\binom{x^{initial} = -0.0093}{y^{initial} = 48.8558}$. PSO started in totally random points between the range of [*-50, 50*].

$$f(x,y) = \frac{x}{1+|y|} \quad (10)$$

A visual can be seen in figure 5 for the objective function of benchmark problem 6. Also comperative results are given in the table 6.

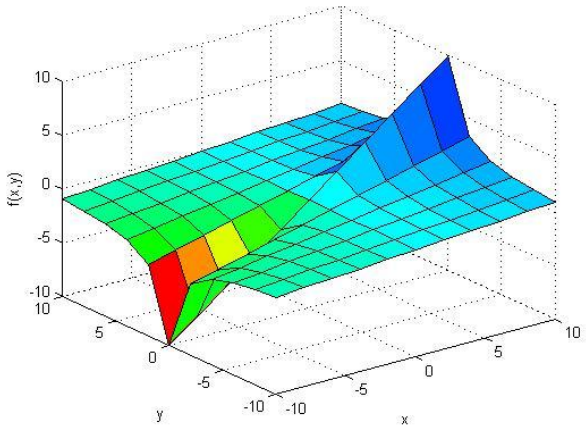

Figure 5: Visual for the Objective Function of Benchmark Problem 6

Table 6: Results for the Problem 6

Algorithms	*x*	*y*	*f(x, y)*	*Epoch Number*
ARSET	-9.9968	3.46E-009	-9.9968	10,000
ACO	-9.9989	2.01E-004	-9.9989	10,000
RSW$\binom{x^{initial} = 42.2649}{y^{initial} = -31.5613}$	-9.773	3.42E-17	-9.773	10,000
PSO	-9.999	-3.85E-17	-9.9999	1,000
ARSET	-9.9996	-2.08E-018	-9.9996	30,000
ACO	-9.9999	-6.05E-008	-9.9999	30,000
RSW$\binom{x^{initial} = 49.8498}{y^{initial} = 45.5905}$	-9.9016	6.52E-17	-9.9016	30,000
PSO	-10	-6.82E-17	-9.9999	3,0000
ARSET	-10	6.67E-008	-10	50,000
ACO	-10	8.07E-011	-10	50,000
RSW$\binom{x^{initial} = -0.0093}{y^{initial} = 48.8558}$	-9.9996	-6.57E-17	-9.9996	50,000
PSO	-10	0	-10	5,000

3.7. Benchmark Problems 7-21

Table 8 covers the information for functions, dimensions, variable ranges and their theoretical bests of the benchmark problems. Although problem 18, same as the problem 4 its parameters are different. The proposed PSO algorithm's results compared with the DRASET and RSW. Results can be seen in table 7.

Table 7: Results for the Problems 7 - 21

No	Epoch #	RSW	DRASET	PSO	Theoretical Best
7	2,501,000	0.00	0.00	0.00	0.00
8	2,501,000	-16.09172	-16.09172	-16.09172	-16.09172
9	2,502,000	0.998	0.998	0.998	0.998
10	2,501,000	0.39788735	0.39788737	0.3978873	0.3978873
11	2,502,000	-1.0316284	-1.0316284	-1.031628	-1.031628
12	2,503,000	3.00	3.00	3.00	3.00
13	2,502,500	-186.73091	-186.73091	-186.7309	-186.7309
14	25,015,000	8.01275646286	8.01275646263	8.01276	8.01276
15	25,050,000	1.28E-28	3.72E-12	3.37E-54	0.00
16	25,050,000	0.00	2.45E-16	1.03E-43	0.00
17	25,080,000	2.3558E-32	5.93E-12	1.86E-36	0.00
18	2,501,500	2.8399E-29	3.9053E-15	4.32E-47	0.00
19	2,506,000	1.0091	1.00	1.00	1.00
20	250,200	1.74415200558	1.74415200796	1.74	1.74
21	25,080,000	1.02E-11	8.17E-9	2.52E-23	0.00

Table 8: Benchmark Problems 7-21

Problem No	Function	Dimension	Variable Range	Theoretical Best
7	$f(x,y) = x^2 + 2y^2 - 0.3\cos(3\pi x) - 0.4\cos(4\pi y) + 0.7$	2	[-1,28 1,28]	0
8	$f(x,y) = [\cos(2\pi x) + \cos(2.5\pi x) - 2.1] * [2.1 - \cos(3\pi y) - \cos(3.5\pi y)]$	2	[-1 1]	-16.09172
9	$f(x_1,x_2) = \left[0,002 + \sum_{j=1}^{25}\left(j + \sum_{i=1}^{2}(x_i - a_{ij})^6\right)^{-1}\right]^{-1}$ $a = \begin{vmatrix} -32 & -16 & 0 & 16 & 32 & -32 & -16 & 0 & 16 & 32 & -32 & -16 & 0 & 16 & 32 & -32 & -16 & 0 & 16 & 32 & -32 & -16 & 0 & 16 & 32 \\ -32 & -32 & -32 & -32 & -32 & -16 & -16 & -16 & -16 & -16 & 0 & 0 & 0 & 0 & 0 & 16 & 16 & 16 & 16 & 16 & 32 & 32 & 32 & 32 & 32 \end{vmatrix}$	2	[-65,536 65,536]	0.998
10	$f(x,y) = \left(y - \frac{5.1}{4\pi^2}x^2 + \frac{5}{\pi}x - 6\right)^2 + 10\left(1 - \frac{1}{8\pi}\right)\cos(x) + 10$	2	$x \in [-5\ 10]$ $y \in [0\ 15]$	0.3978873
11	$f(x,y) = \left(4 - 2.1x^2 + \frac{x^4}{3}\right)x^2 + xy + (4y^2 - 4)y^2$	2	$x \in [-3\ 3]$ $y \in [-2\ 2]$	-1.0316285
12	$f(x,y) = [1 + (x + y + 1)^2(19 - 14x + 3x^2 - 14y + 6xy + 3y^2)] *$ $[30 + (2x - 3y)^2(18 - 32x + 12x^2 + 48y - 36xy + 27y^2)]$	2	[-5 5]	3
13	$f(x,y) = \left[\sum_{i=1}^{5} i\cos((i+1)x + i)\right] * \left[\sum_{i=1}^{5} i\cos((i+1)y + i)\right]$	2	[-10 10]	-186.73091
14	$f(x,y,z) = \sum_{i=1}^{5}(x(a_i)^y(b_i)^z - c_i)^2$ $a = \|5\ 3\ 0.6\ 0.1\ 3\|$ $b = \|10\ 1\ 0.6\ 2\ 1.8\|$ $c = \|2.122\ 9.429\ 23.57\ 74.25\ 6.286\|$	3	$[-\infty\ +\infty]$	8.01276
15	$f(x) = 100(x_2 - x_1^2)^2 + (1 - x_1)^2 + 90(x_4 - x_3^2)^2 + (1 - x_3)^2 +$ $10.1[(x_2 - 1)^2 + (x_4 - 1)^2] + 19.8(x_2 - 1)(x_4 - 1)$	4	[-10 10]	0
16	$f(x) = \sum_{i=1}^{19}\left[\left(x_i^2\right)^{(x_i^2+1)} + \left(x_{i+1}^2\right)^{(x_i^2+1)}\right]$	20	[-1 4]	0
17	$f(x) = (\pi/20)\left[10\sin^2(\pi x_1) + \sum_{i=1}^{19}\left((x_i - 1)^2(1 + 10\sin^2(\pi x_{i+1}))\right) + (x_{20} - 1)^2\right]$	20	[-10 10]	0
18	$f(x,y) = 100(y - x^2)^2 + (1 - x)^2$	2	[-10 10]	0
19	$f(x,y) = exp\left\{\frac{1}{2}(x^2 + y^2 - 25)^2\right\} + \sin^4(4x - 3y) + \frac{1}{2}(2x + y - 10)^2$	2	[-5 5]	1
20	$f(x,y) = \left[12 + x^2 + \frac{1+y^2}{x^2} + \frac{x^2y^2+100}{(xy)^4}\right] * 0.1$	2	[0 10]	1.74
21	$f(x) = (x_1 + 10x_2)^2 + 5(x_3 - x_4)^2 + (x_2 - 2x_3)^4 + 10(x_1 - x_4)^4$	4	[-5 5]	0

4. Conclusion

In this paper PSO's performance and robustness are shown. For 21 benchmark problems PSO's results are quite competitive. PSO converges the theoretical best results with lesser epoch number for first four problems and problem 6. In this work when table 8 examined, 2 problems were one dimensional, 13 problems were two dimensional, 1 problem was three dimensional, two problems were 4 dimensional and finally 2 problems were twenty dimensional. Problem 5 were run for 10, 25 and 50 dimensional. Due to problem 5, 16 and 17's dimension toughness(N=10, 20, 25 and 50), PSO was run for with 100 and 200 particles and its solutions are yet close to the global minimum for these dimensions. 14th problem's solution space is within the range $[-\infty +\infty]$. The proposed PSO started within the range of [*-inf inf*]. For first four problems and problem 6, PSO achieved best known results. For problems 7-14, 19 and 20, PSO achieved theoretical bests. For those 15-18 and 21 problems PSO obtained worse solution yet its results are relatively close to the global minimums. Algorithms for function optimization are widely studied by the researchers. Each algorithm has their ups and downs when subject to their results and performance.

References

[1] Eberhart, R., & Kennedy, J. A new optimizer using particle swarm theory. In Micro Machine and Human Science, 1995. MHS'95., Proceedings of the Sixth International Symposium on (pp. 39-43). (1995), IEEE.

[2] Clerc, M., & Kennedy, J. The particle swarm-explosion, stability, and convergence in a multidimensional complex space. IEEE transactions on Evolutionary Computation, 6(1), (2002). 58-73.

[3] Y.Yongjian, L.Yumei, A new discrete filled function algorithm for discrete global optimization, Journal of Computational and Applied Mathematics 202 (2007) 280 – 291.

[4] C. Hamzacebi, F. Kutay, Continuous functions minimization by dynamic random search technique, Applied Mathematical Modelling 31 (2007) 2189-2198.

[5] M. D. Toksari, Ant colony optimization for finding the global minimum, Applied Mathematics and Computation 176 (2006) 308–316.

[6] J. Li, R. R. Rhinehart, Heuristic random optimization, Computers chem. Engng (1998) 22 427-444.

[7] C. Hamzacebi, F. Kutay, A heuristic approach for finding the global minimum: Adaptive random search technique, Applied Mathematics and Computation 173 (2006) 1323–1333.

[8] Cura, Tunchan. "A random search approach to finding the global minimum." Int. J. Contemp. Math. Science 5.4 (2010): 179-190.

[9] Pan, Quan-Ke, et al. "An improved fruit fly optimization algorithm for continuous function optimization problems." Knowledge-Based Systems 62 (2014): 69-83.

[10] Wang, Jie-Sheng, and Jiang-Di Song. "Application and Performance Comparison of Biogeography-based Optimization Algorithm on Unconstrained Function Optimization Problem." International Journal of Applied Mathematics 47.1 (2017).

[11] Nabil, Emad. "A modified flower pollination algorithm for global optimization." Expert Systems with Applications 57 (2016): 192-203.

[12] Guo, Ying, et al. "Function Optimization via a Continuous Action-Set Reinforcement Learning Automata Model." Proceedings of the 2015 International Conference on Communications, Signal Processing, and Systems. Springer Berlin Heidelberg, (2016).

[13] Wang, Chun-Feng, and Yong-Hong Zhang. "An improved artificial bee colony algorithm for solving optimization problems." IAENG International Journal of Computer Science 43.3 (2016): 336-343.

[14] Y. Liang, and K. S. Leung, "Genetic Algorithm with adaptive elitist-population strategies for multimodal function optimization," Applied Soft Computing, vol. 11, no. 2, (2011), pp. 2017–2034.