Modern Applied Science; Vol. 12, No. 1; 2018
ISSN 1913-1844 E-ISSN 1913-1852
Published by Canadian Center of Science and Education

Movement Particle Swarm Optimization Algorithm

Amjad A. Hudaib[1] & Ahmad K. Al Hwaitat[2]

[1] Department of Computer Information Systems, King Abdullah II School for Information Technology, The University of Jordan, Amman, Jordan

[2] Department of Computer Science, King Abdullah II School for Information Technology, The University of Jordan, Amman, Jordan
Correspondence: Amjad A. Hudaib, School King Abdullah II School for Information Technology, The University of Jordan, Amman, Jordan. E-mail: ahudaib@ju.edu.jo

Received: December 10, 2017 Accepted: December 18, 2017 Online Published: December 31, 2017

doi:10.5539/mas.v12n1p148 URL: https://doi.org/10.5539/mas.v12n1p148

Abstract

Particle Swarm Optimization (PSO) is a well known meta-heuristic that has been used in many applications for solving optimization problems. But it has some problems such as local minima. In this paper proposed an optimization algorithm called Movement Particle Swarm Optimization (MPSO) that enhances the behavior of PSO by using a random movement function to search for more points in the search space. The meta-heuristic has been experimented over 23 benchmark faction compared with state of the art algorithms: Multi-Verse Optimizer (MFO), Sine Cosine Algorithm (SCA), Grey Wolf Optimizer (GWO) and particle Swarm Optimization (PSO). The Results showed that the proposed algorithm has enhanced the PSO over the tested benchmarked functions.

Keywords: optimization, Particle Swarm Optimization, optimal solution, swarm intelligence

1. Introduction

Particles Swarm Optimization is an evolutionary computing technology that is derived from a simplified social model of the simulation. It is also known as Particle Swarm Algorithm developed by (Kennedy J. and Mendes R. in 2002), where "Swarm" comes from the particle swarm in line with five basic principles of group intelligence in the development of models applied to artificial life. "Particle" is a compromise because it is necessary to describe the members of the group as no quality, no volume, and to describe its speed and acceleration status (Vijayalakshmi in 2014).

Particle Swarm Optimization, (PSO) is a method of numerical optimization for the use of which you do not need to know the exact gradient of the optimized function. This method belongs to the group algorithms of swarm intelligence, which describes collective behavior of a decentralized self-organizing system (Shi, Y., & Eberhart, R. in1998). The systems of swarm intelligence, as a rule, consist of many agents locally interacting with each other and with the environment. The agents themselves are usually rather simple, but all together, locally interacting, create the so-called swarm intelligence. The method was first developed to simulate the social behavior of a flock of birds and fish. As a result of the development of the method, it has been successfully applied to problems finding the extreme points of a function (Kessentini, S., & Barchiesi, D. in 2015).

The PSO algorithm was originally designed for graphical simulation of a beautiful and an unpredictable movement of birds. Through the observation of animal social behavior, found in the group of social sharing of information to provide an evolutionary advantage, and as a basis for the development of the algorithm. The initial version of the PSO is formed by adding the speed matching of the neighbors, taking into account the multidimensional search and the acceleration of the distance. And then introduced the inertia weight to better control the exploitation (exploitation), exploration (exploration), the formation of a standard version (Eberhart R., Y. Shi, and J. Kennedy in 2001).

The PSO algorithm uses the following psychological assumptions: In seeking a consistent cognitive process, individuals tend to remember their beliefs while taking into account the beliefs of their colleagues. When it is better aware of the beliefs of colleagues, will be adaptively adjusted.

This paper proposes a novel algorithm for solving optimization problems that is called Movement Particle Swarm Optimization Algorithm (MPSO). The algorithm has been tested over 20 will known benchmarks functions from CEC2005 suit, the results have been verified by a comparative study with the state of art

optimization algorithms GWO, SCA, MFO, and also with PSO.

The rest of this paper is organized as follows. Section 2, presents the related work. Section 3 describes the proposed MPSO algorithm. Section 4 provides experimental results and comparison of MPSO with other meta-heuristic. Finally, the conclusions and future work are presented in Section 6.

1.1 Related Work

(Bergh and Engelbrecht, 2004) proposed a cooperative PSO (Cooperative PSO) algorithm, This idea is the use of cooperative behavior, the use of multiple groups in the target search space in different dimensions of the search, that is, an optimization solution by a number of independent groups are collaboratively completed, and each group is only responsible for optimizing the components of this solution vector. (Baskar S. and Suganthan P. N., 2004) proposed a similar collaboration PSO, known as concurrent PSO (concurrent PSO, CONPSO), which uses two groups to optimize a solution vector. Recently, (El Abd et al., 2006) have combined a hierarchical cooperative PSO (hierarchical cooperative PSO) that was proposed by Bergh and (Engelbrecht, 2004 and Baskar and Suganthan, 2004). Whether the particle swarm is in the D-dimensional search or multiple particle groups on different dimensions of the collaborative search, its purpose is to be able to find each particle in favor of fast convergence to the global optimal solution learning object. (Liang et al., 2004) proposed a method that can be both D-dimensional search and cannot be selected on different dimensions a new learning strategy for learning objects is called the Comprehensive Learning Particle Swarm Optimizer (CLPSO).

Stretching PSO (SPSO): SPSO the so-called stretching technique (Parsopoulos et al., 2001) and the deflection and repulsion technology applied to the PSO, the objective function of the transformation, limiting the particles have been found to the local minimum solution movement, which will help the particles have more opportunity to find the global optimal solution (Parsopoulos and Vrahatis, 2004).

Chaotic particle swarm optimization: chaos is a seemingly cluttered nature, in fact, implied inherent regularity of the common nonlinear phenomenon, with random and regularity. The chaos of the chaotic motion is used to generate the chaotic sequence based on the historical best position of the particle swarm, and the optimal position chaos PSO (chaos particle swarm optimization, CPSO) is proposed to replace the position of a particle in the particle swarm. In addition, (Wang et al., 2007) proposed using the adaptive PSO with the inertia weight to adapt to the objective function value, the global search is carried out, and the optimal location is searched by chaos local search.

A chaotic PSO combining PSO with chaos search is used to determine PSO parameters (inertia weights and acceleration constants) using chaotic sequences. A particle swarm model based on deterministic chaotic Hopfield neural network is proposed. (Kalman swarm: Monson and Seppi, 2004) used Kalman filter to update the particle position. Principal component PSO: (Mark et al., 2005) combined with the principal component analysis technique, the particles weren't only in the n-dimensional x space flight according to the traditional algorithm, but also in the m-dimensional z Space synchronized flight (m <n).

The PSO algorithm is based on the group, moving the individual in the group to a good area according to the fitness of the environment. However, it does not use the evolutionary operator for the individual, but rather treats each individual as a volume-free particle (point) in the D-dimensional search space that travels at a certain speed in the search space, depending on its own flight experience and companion flight experience to dynamically adjust (Suganthan, P.N in 1999). The i-th particles are expressed as Xi = (xi1, xi2,...,xiD), and the best position (with the best fitness) that is experienced is Pi = (pi1, pi2,..., piD), also called pbest. The index number of the best position where all the particles of the group has been traced is denoted by the symbol g, that is, Pg, also known as gbest. The velocity of particle i is expressed by Vi = (vi1, vi2,...,viD). For each generation, its d-dimensional ($1 \le d \le D$) varies according to the following equation:

$$v[id] = w * v[id] + c1 * rand() * (pbest[id] - present[xid]) + c2 * rand() * (gbest[gd] - present[xid]) \quad (1)$$

Figure 1. Equation PSO Algorithm

v () is the particle velocity, present () is the current particle (solution). pbest () and gbest () are defined as stated before. rand () is a random number between (0,1). c1, c2 are learning (acceleration) factors. usually c1 = c2 = 2. w is inertia weight.

In addition, the velocity Vi of the particles is limited by a maximum velocity Vmax. If the current acceleration of the particles causes its velocity vid in a dimension to exceed the maximum velocity vmax, d of the dimension, the velocity of the dimension is limited to the dimension maximum velocity vmax, d. for the equation (1), the first part is the inertia of the particle's previous behavior, the second part is the "cognition" part, the reflection of the particle itself; the third part is the "social" information sharing and mutual cooperation (Junying C.et al. in 2005).

The "cognitive" part can be explained by Thorndike's law of effect, that is, an enhanced random behavior is more likely to occur in the future. The behavior here is "cognition" and assumes that the correct knowledge is reinforced, and that such a model assumes that the particles are motivated to reduce the error (Kata S. et al., 2004).

The "social" part can be explained by Bandura's vicarious reinforcement. According to the theory, when the observer observes that a model strengthens a behavior, it increases the chances of its behavior. That the micro particle itself will be imitated by other particles (Shuquan L. and Kongguo Z,2008).

```
For each particle
  Initialize particle
END
Do
  For each particle
    Calculate fitness value
    If the fitness value is better than the best fitness value (pBest) in history
      set current value as the new pBest
  End

  Choose the particle with the best fitness value of all the particles as the gBest
  For each particle
    Calculate particle velocity according equation          (2)
    Update particle position according equation             (1)
  End
While maximum iterations or minimum error criteria is not attained
```

Figure 2. PSO algorithm pseudo code

2. Movement Particle Swarm Optimization Algorithm (MPSO)

2.1 Movement Particle Swarm Optimization Algorithm (MPSO)

In this section, the inspiration of the proposed algorithm is first discussed. Then the mathematical model is provided.

2.2 Inspiration of MPSO

Viruses have both similarities and differences with other living organisms as shown in figure 3. One of the features of viruses that indicate their belonging to living matter is their need for replication and creation of offspring, but, unlike living organisms, a virus cannot survive on its own. It is activated only when it replicates in the host cell using host resources and nutrients. When a virus enters the cell, its sole purpose is to create multiple copies of itself to infect other cells. Everything that it does is aimed at increasing the fitness and the number of offsprings.

The virus, in order to proliferate and, thus, cause infection, it is necessary to penetrate into the cells of the host organism and begin using cell material. To penetrate the cell, proteins of the surface of the virus bind to specific surface proteins of the cell. Attachment, or adsorption, occurs between the virus particle and the cell membrane. A hole is formed in the membrane and the virus particle or only the genetic material gets inside the cell where the virus will multiply. Then the virus must take control of the cellular replication mechanism. At this stage, the distinction between susceptibility and tolerance takes place in the host cell. Tolerance leads to a decoupling of the infection. Once control of the cell is established and its environment is suitable for the virus to start creating its own copies, replication occurs quickly, giving rise to millions of new viruses. After the virus has created many copies of itself, the cell is exhausted due to the use of its resources. More viruses are not needed, so the cell often dies and newborn viruses have to look for a new host. This represents the final stage of the life cycle of

the virus.

Thus, the virus entirely depends on the host cell. Most viruses are species-specific and affect only a narrow range of hosts - plants, animals, fungi or bacteria.

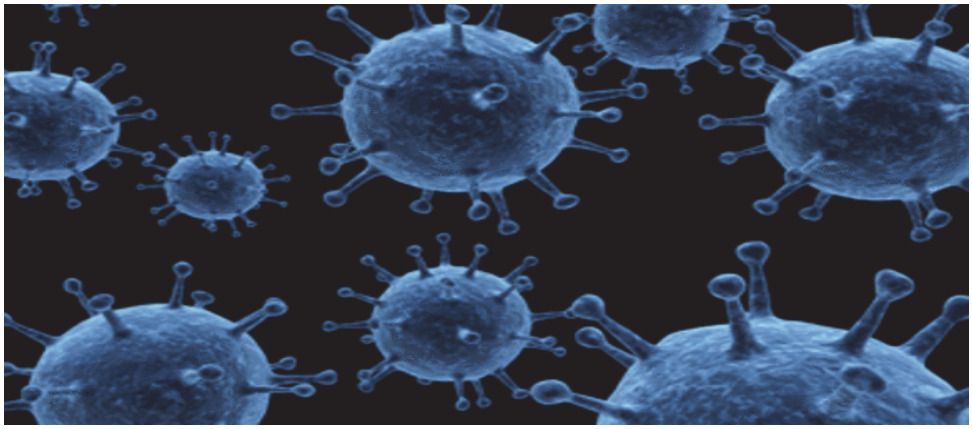


Figure 3. Virus particles

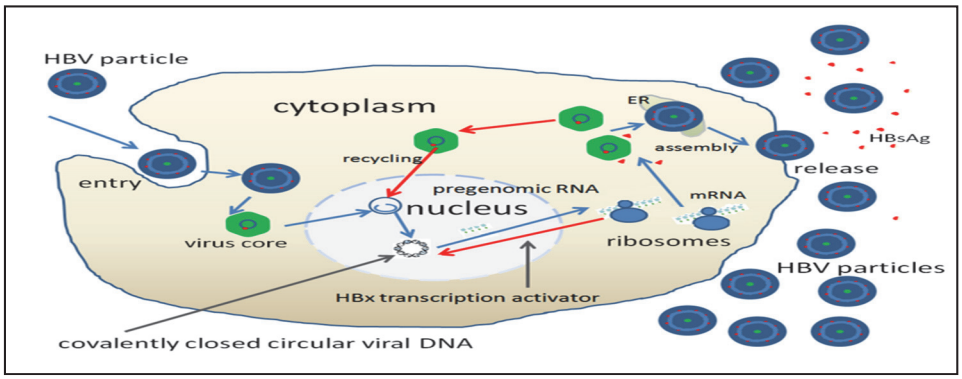


Figure 4. Viruses Movements

Some viruses can "hide" inside the cell. This can result from evading the host's defense reactions and immune system, or simply because the continuation of replication is not in the interests of the virus. This hiding is called latency. During this time, the virus does not give rise to the offspring and remains inactive until the external stimulus - for example, light or stress, activates it.

2.3 Mathematical Model of MPSO

In this subsection the mathematical models of the algorithm is described, I chose main operations of the viral particles, also known as virions, consist of two or three parts: the genetic material made from either DNA, a protein coat and and the protein coat when they are outside a cell.

Initialize each virus position and velocity if virus position below the lower bound or above the upper bound then we initialize virus position and initialize virus velocity and way of movement as described in equation 1and 2

$$VP = lb + (ub - lb).*rand(N,dim) \quad (1)$$

$$VV = lb + (ub - lb).*rand(N,dim) \quad (2)$$

Where VV is the virus velocity and VP is the virus position, rand (): is a random function that generates numbers between 0 and 1, ub: the upper bound of the space dimension, lb: the lower bound of the space dimension, N is the number of generated viruses and Dim is the dimentaion of the search space perform local search on overall global best position and communicate genetic material RNA between viruses are mathematically represented by equation 3 that if the obtained value (OBV) is less than the global value of the virus particles (GP) then each virus will have a local value after applying in the fitness function as shown in equation 4.

$$OBV = fitness\ (VP) \quad (3)$$

$$VIRUS\_VALUE = OBV \quad (4)$$

Find minimum in the neighborhood as shown in equation 5 by fist sorting the particles fitness value where the neighborhood of a particle is its near particles in the sorted index where the minimum value is in the upper of the index sorted from largest to smallest value.

$$VMIN = max(OBV) \quad (5)$$

Each virus will perform random movement (RW)and local search (LS) for new cells to invade which is represented by new fitness value and it moves to a period of space defines as Cwhich is larger than 1.

$$VNP = rand(N,1) - C \quad (6)$$

Where VNP is the virus new position with the random movement is a movement space that the virus will move larger than 1, N is the virus particle population number

The pseudocode of the algorithm that describe the steps that is applied to the virus particle is shown in figure5

```
initialize position
Initialize velocity
initialize best position
initialize global best
find minimum in the neighbourhood
update individual best positions
update local best positions
find minimum in the neighborhood
update global best
local search
perform local search on overall global best position
perform local search on pbest
Random movement of particles as shown in equation 6
```

Figure 5. MPSO Pseudo code

3. Results and Discussion

In this section the MPSO algorithm is tested on 23 benchmark functions usedc by many researchers (16, 48-51) from CEC 2005 special session benchmark functions (52). We have chosen these test functions to be able to compare our results to those of the current meta-heuristics. These benchmark functions are listed in Table 1. Where Dim indicates dimension of the function, range is the boundary of the function's search space, and f min is the optimum. Some of these functions represent the shifted, rotated, expanded, and combined variants of the classical functions which offer the greatest complexity among the current benchmark functions, Generally speaking, the benchmark functions used are minimization functions and can be divided into four groups: unimodal, multimodal, fixed-dimension multimodal, and composite functions. Note that detailed descriptions of the composite benchmark functions are available in the CEC 2005 technical report.

Table 1. The benchmark functions mathematical formulation

Unimodal benchmark functions

Function		Dim	Range	Fmin
F1	$f1(x) = \sum_{i=1}^{n} x_i^2$	30	-100,100	0
F2	$f2(x) = \sum_{i=1}^{n} \lvert x_i \rvert + \prod_{i=1}^{n} \lvert x_i \rvert$	30	-10,10	0

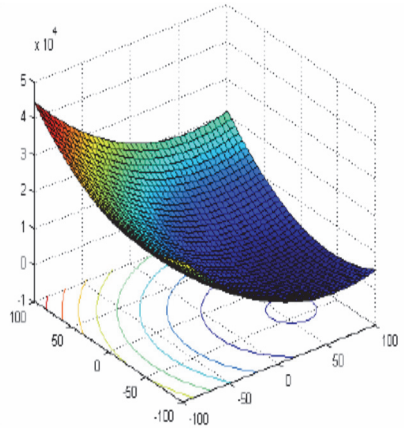
F3	$f3(x) = \sum_{i=1}^{n} \left(\sum_{j=1}^{n} x_j \right)^2$	30	-100,100	0
F4	$f4(x) = \max i\{\lvert x_i \rvert, 1 \le i \le n\}$	30	-100,100	0
F5	$f5(x) = \sum_{i=1}^{n=1} [100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2]$	30	-30,30	0
F6	$f6(x) = \sum_{i=1}^{n} (x_i + 0.5)^2$	30	-100,100	0
F7	$f7(x) = \sum_{i=1}^{n} i x_i^4 + random(0,1)$	30	-1.28,1.28	0

Multimodal benchmark functions		**Dim**	**Range**	**Fmin**
Function				
F8	$f8(x) = \sum_{i=1}^{n} -x_i \sin\sqrt{\lvert x_i \rvert}$ * $\sum_{i=1} i x_i^4 *$ random(0,1) *	30	-500,500	-418.9829 5
F9	$f9(x) = \sum_{i=1}^{n} [x_i^2 - 10\cos(2\pi x_i) + 10]$	30	-5.12,5.12	0
F10	$f10(x) = -20\exp\left(-0.2\sqrt{\frac{1}{n}\sum_{x=i}^{n} x_i^2}\right) - \exp\left(\frac{1}{n}\sum_{i=1}^{n}\cos(2\pi x_i)\right) + 20 + e$	30	-32,32	0
F11	$f11(x) = \frac{1}{4000}\sum_{i=1}^{n} x_i^2 - \prod_{i=1}^{n}\cos\left(\frac{x_i}{\sqrt{i}}\right) + 1$	30	-600,600	0
F12	$f12(x) = \frac{\pi}{n}\left\{10\sin(\pi y_1) + \sum_{i=1}^{n=1}(y_i - 1)^2[1 + 10\sin^2(\pi y_{i+1})] + (y_n - 1)^2\right\} + \sum_{i=1}^{n} u(x_i, 10,100,4)$ $y_i = 1 + \frac{x_i + 1}{4}$ $u(x_i, a, k, m) = \begin{cases} k(x_i - a)^m x_i > a \\ 0 \quad -a < x_i < a \\ k(-x_i - a)^m x_i < a \end{cases}$	30	-50,50	0
F13	$f13(x) = 0.1\left\{\sin^2(3\pi x_1) + \sum_{i=1}^{n}(x_i - 1)^2[1 + \sin^2(3\pi x_i + 1)] + (x_n - 1)^2[1 + \sin^2(2\pi x_n)]\right\} + \sum_{i=1}^{n} u(x_i, 5,100,4)$	30	-50,50	0
F14	$f14(x) = -\sum_{i=1}^{n}\sin(x_i).\left(\sin\left(\frac{ix_i^2}{\pi}\right)\right)^{2m}$, m=10	30	0, π	-4.687
F15	$f15(x) = \sum_{i=1}^{11}\left[a_i - \frac{x_1(b_i^2 + b_i x_2)}{b_i^2 + b_i x_3 x_4}\right]$	30	-20,20	-1
F16	$f16(x) = 4x_1^2 - 2.1x_1^4 + \frac{1}{3}x_1^6 + x_1x_2 - 4x_2^2 + 4x_2^4$	30	-10,10	-1

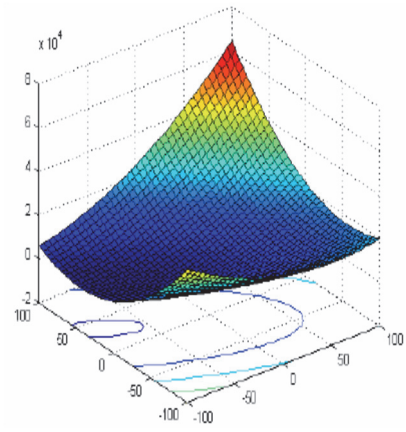
Fixed-dimensions multimodal benchmark functions				
Function		**Dim**	**Range**	**Fmin**

F17	$F17(X) = \left(x_2 - \frac{5.1}{4\pi^2}x_1^2 + \frac{5}{\pi}x_1 - 6\right)^2 + 10\left(1 - \frac{1}{8\pi}\right)\cos X_1 + 10$	2	-5,5	0.398
F18	$f18(x) = [1 + (x_1 + x_2 + 1)^2(19 - 14x_1 + 3x_1^2 - 14x_2 + 6x_1x_2 + 3x_2^2)]x[30 + (2x_1 - 3x_2)^2x\,(18 - 32x_1 + 12x_1^2 + 48x_2 - 36x_1x_2 + 27x_2^2]$	2	-2,2	3
F19	$f19(x) = -\sum_{i=1}^{4} C_i \exp\left(-\sum_{j=1}^{3} a_{ij}(x_j - p_{ij})^2\right)$	3	1,3	-3.86
F20	$f20(x) = -\sum_{i=1}^{4} C_i \exp\left(-\sum_{j=1}^{6} a_{ij}(x_j - p_{ij})^2\right)$	6	0,1	-3.32
F21	$f21(x) = -\sum_{i=1}^{5}[(X - a_i)(X - a_i)^T + C_i]^{-1}$	4	0,10	-10.1532
F22	$f22(x) = -\sum_{i=1}^{7}[(X - a_i)(X - a_i)^T + C_i]^{-1}$	4	0,10	-10.4028
F23	$f23(x) = -\sum_{i=1}^{10}[(X - a_i)(X - a_i)^T + C_i]^{-1}$	4	0,10	-10.5363

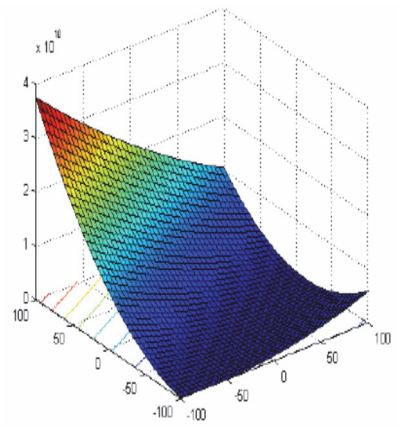
3.1 The Functions Charts



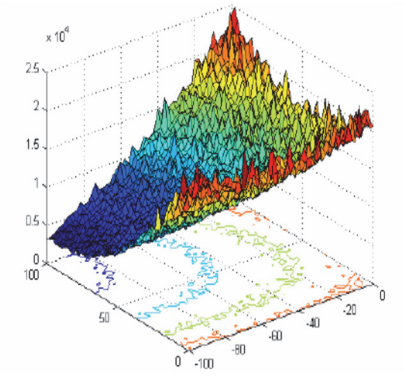
F 1



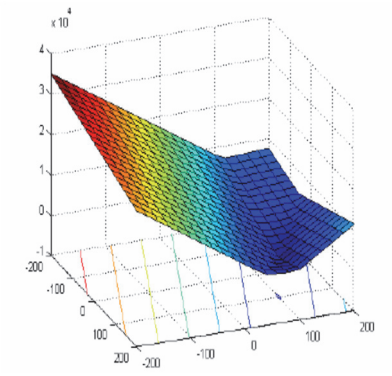
F2



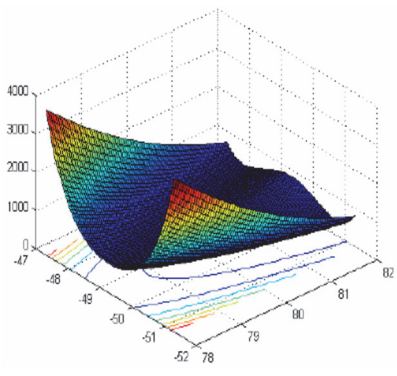
F3



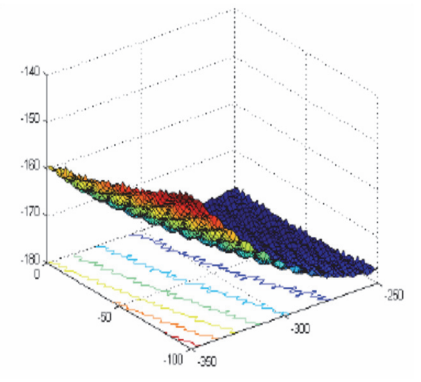
F4



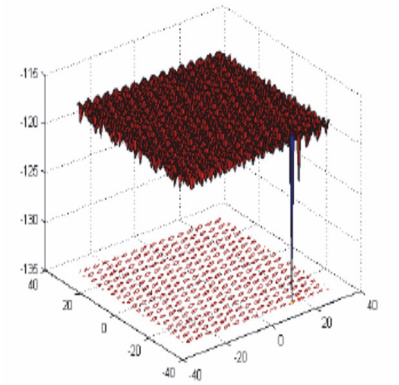
F5



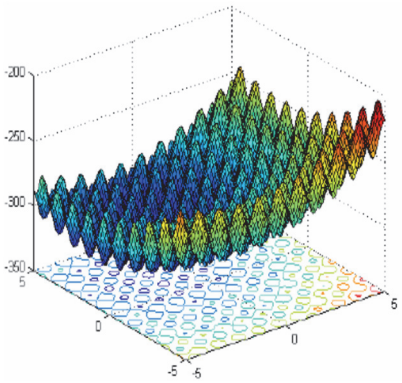
F6



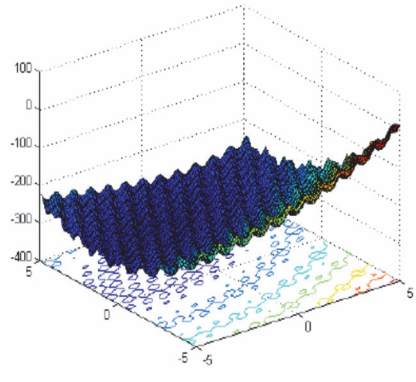
F7



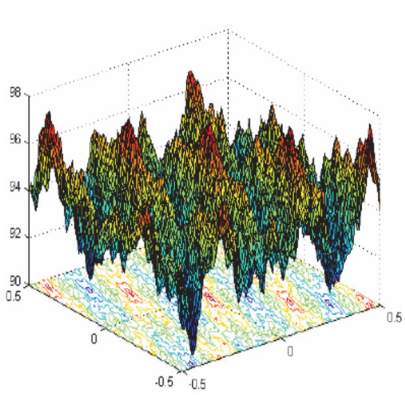
F8



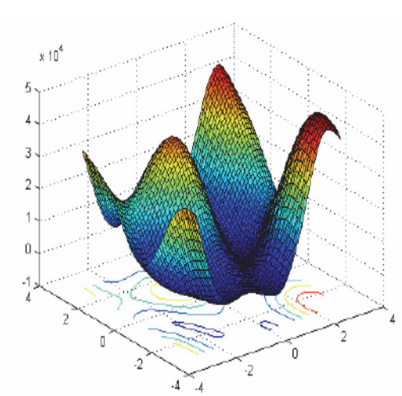
F9



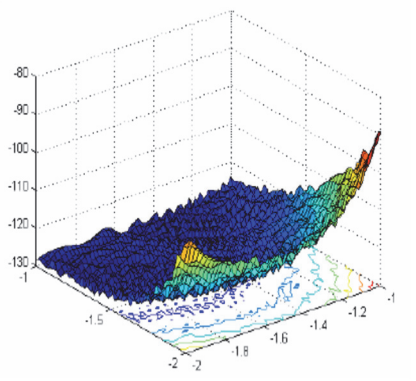
F10



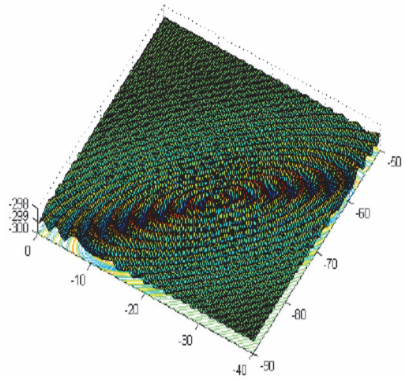
F11



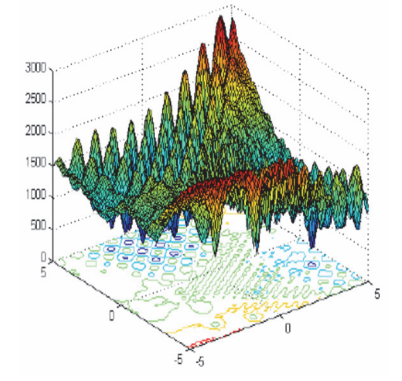
F12



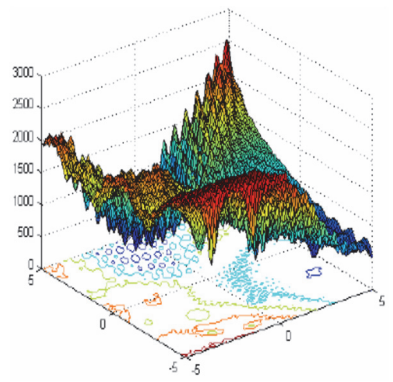
F13

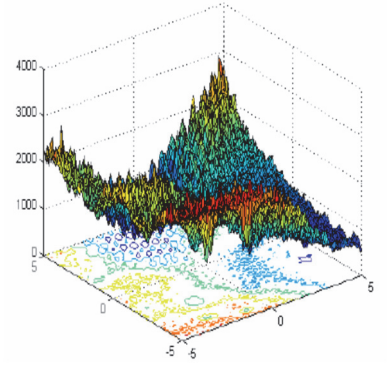


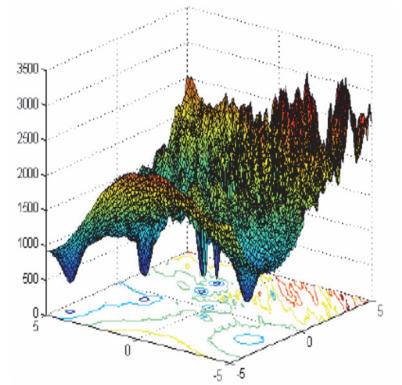
F14



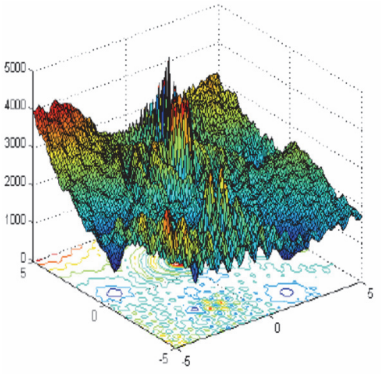
F15



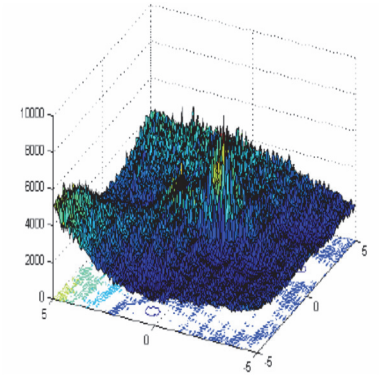




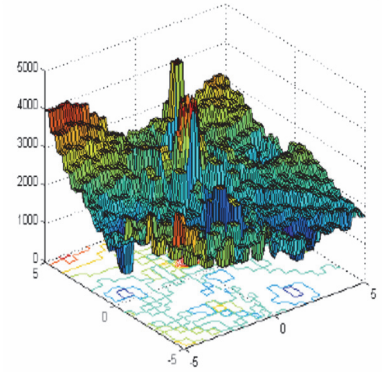
F16



F17

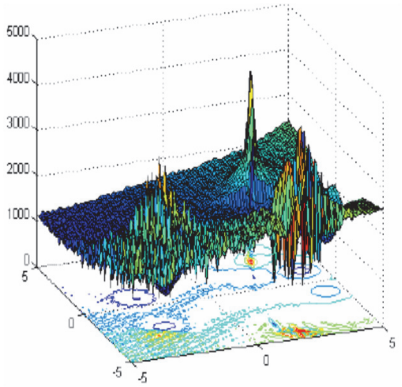


F18



F19

F20



F21

F22

3.2 Comparison with Other Algorithms

The MPSO algorithm was run 50 times on each benchmark function. The (average and standard deviation are reported in Table 3 to Table 4. For verifying the results, we compared MPSO with GWO, SCA, PSO and MFO on CEC2005 benchmark set problems.

Table 3. MEAN, STD, MIN, MAX values over 30 experiments for MPSO, SCA

	MPSO			**SCA**		
	MEAN	MIN	MAX	MEAN	MIN	MAX
F1	1.8E-113	3.5E-126	2.5E-112	4.67E-06	4.36E-11	0.000101
F2	4.58E-65	1.95E-70	3.95E-64	7.98E-08	9.81E-10	6.49E-07
F3	2.29E-60	4.19E-75	3.53E-59	302.7088	2.99102	1340.977
F4	2.15E-47	5.85E-51	4.57E-46	2.957373	0.089178	9.766226
F5	28.0683	27.85857	28.12677	27.68035	26.71609	28.84735
F6	4.881366	4.190022	5.200387	3.52211	3.056689	4.099798
F7	7.04E-05	4.56E-06	0.000247	0.006412	0.001282	0.018567
F8	-3369.02	-4136.36	-2678.46	-4389.09	-4859.08	-3980.85
F9	0	0	0	1.720257	3.2E-10	21.43796
F10	8.88E-16	8.88E-16	8.88E-16	7.593153	1.66E-06	20.18279
F11	0	0	0	0.038823	1.95E-10	0.691525
F12	0.483425	0.385479	0.549483	0.350343	0.285064	0.534462
F13	2.401158	2.217149	2.492816	1.958798	1.595011	2.507098
F14	2.347667	0.998259	3.027778	0.998004	0.998004	0.998008
F15	0.000373	0.000319	0.00044	0.000713	0.000312	0.001282

F16	-1.03162	-1.03163	-1.03161	-1.03163	-1.03163	-1.03162
F17	0.398026	0.397887	0.39857	0.398002	0.397892	0.398556
F18	3.000005	3	3.000023	3	3	3.000002
F19	-3.85964	-3.8621	-3.85485	-3.85673	-3.86265	-3.85448
F20	-3.06186	-3.15441	-2.95016	-3.07933	-3.27185	-2.62989
F21	-5.61528	-9.14535	-4.57493	-5.04308	-8.03242	-0.88067
F22	-6.08913	-9.54099	-4.71179	-6.48544	-9.21934	-4.94606
F23	-5.52455	-8.13257	-4.6243	-6.04945	-9.48245	-3.73482

Table 4. MEAN, STD, MIN, MAX values over 30 experiments for GWO, PSO

	PSO			**GWO**		
	MEAN	MIN	MAX	MEAN	MIN	MAX
F1	4.34E-20	2.28E-23	4.9E-19	1.7E-104	1.4E-106	2.3E-103
F2	2.88E-10	5.11E-12	2.07E-09	1.25E-59	4.78E-61	3.71E-59
F3	0.881832	0.256993	3.544075	1.02E-35	1.42E-40	2.13E-34
F4	0.07112	0.02485	0.172054	4.65E-27	2.75E-29	4E-26
F5	46.00925	6.282705	149.2509	25.56	24.20627	27.05306
F6	2.14E-20	4.57E-23	1.04E-19	0.06478	3.2E-06	0.500989
F7	0.0182	0.007609	0.029951	0.000104	3.51E-05	0.000328
F8	-6884.27	-8009.21	-5284.21	-6741.27	-7815.45	-5946.13
F9	23.62278	12.93446	38.80348	0.17346	0	5.377251
F10	8.23E-11	7.39E-12	2.87E-10	8.57E-15	7.99E-15	1.51E-14
F11	0.008423	0	0.029547	0.000322	0	0.009971
F12	5.41E-22	2.73E-24	6.47E-21	0.003681	1.5E-07	0.019781
F13	9.27E-16	3.42E-23	2.86E-14	0.038523	2.22E-06	0.202303
F14	0.998004	0.998004	0.998004	1.382023	0.998004	2.982105
F15	0.000544	0.000307	0.001062	0.00166	0.000307	0.020363
F16	-1.03163	-1.03163	-1.03163	-1.03163	-1.03163	-1.03163
F17	0.397887	0.397887	0.397887	0.397887	0.397887	0.397888
F18	3	3	3	3	3	3.000001
F19	-3.86278	-3.86278	-3.86278	-3.86232	-3.86278	-3.85491
F20	-3.24145	-3.322	-3.2031	-3.26146	-3.32199	-3.13279
F21	-8.5175	-10.1532	-5.0552	-9.66121	-10.1532	-5.05519
F22	-9.72108	-10.4029	-5.08767	-9.71961	-10.4029	-5.08767
F23	-9.84166	-10.5364	-5.12848	-10.3618	-10.5364	-5.12848

The first set of the benchmark test functions (F1 to F5) has no local optima and there is only one global optima. This makes them very suitable for testing the convergence speed and exploitation of algorithms.

The second set of the benchmark test functions (F6 to F23), has multiple local values in addition to the global optimum. This set is used to test local optima avoidance and explorative ability of an algorithm.

The third set of the benchmark test functions is the composite test functions are the rotated, shifted, biased, and combined version of several unimodal and multi-modal test functions.

3.3 Exploitation and Exploration Analysis

The results in Table 6 show that the MPSO showed a competitive results with the compared optimizer's. Firstly, the MPSO algorithm shows better results on unimodal test functions. Due to the characteristics of the unimodal test functions, these results strongly show that the MPSO algorithm has a high exploitation and convergence.

According to the results, MPSO is able to provide very competitive results. This algorithm outperforms GWO and SCA in (F1to F4) and (F8 to F11), (F15 to F16), (F19 to F23). Also, it outperforms PSO in (F1 to F5) and (F7), (F9 to F11), (F14 to F17), (F19), (F21) it also outperforms MFO in (F1 to F4) and (F7), (F8 to F11) and (F15 to F16).

Table 6. MPSO results compared to GWO, SCA, PSO, MFO

	MPSO VS GWO	MPSO VS SCA	MPSO VS PSO	MPSO VS MFO
F1	yes	yes	yes	yes
F2	yes	yes	yes	yes
F3	yes	yes	yes	yes
F4	yes	yes	yes	yes
F5	yes	yes	yes	yes
F6	no	no	no	no
F7	no	no	yes	yes
F8	yes	yes	yes	no
F9	no	yes	yes	yes
F10	yes	yes	yes	yes
F11	yes	yes	yes	yes
F12	yes	yes	yes	yes
F13	yes	yes	yes	yes
F14	no	no	no	no
F15	no	yes	yes	no
F16	yes	yes	yes	yes
F17	no	no	no	no
F18	no	no	no	no
F19	no	yes	no	no
F20	no	yes	no	yes
F21	yes	yes	yes	yes
F22	yes	yes	yes	yes
F23	no	yes	yes	yes

3.4 Local Minima Avoidance

MPSO shows a good balance between exploration and exploitation that result in high local optima avoidance.

3.5 Stability of the MPSO

We calculated the standard deviation for all functions from F1 to F23 to test the stability of the MPSO as shown in Table 7

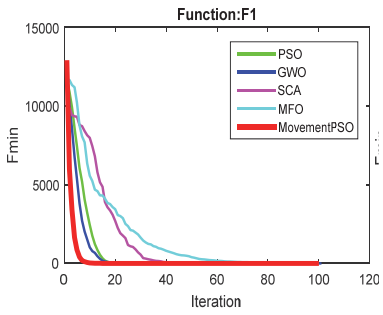
Table 7. Standard deviation of MPSO, SCA, PSO, GWO and MFO

	STD of MPSO	STD of SCA	STD of PSO	STD OF GWO	STD OF MFO
F1	5.2E-113	STD	9.62E-20	4.2E-104	0.01721
F2	1.03E-64	1.87E-05	5.2E-10	9.25E-60	0.031259
F3	8.14E-60	1.42E-07	0.643122	3.98E-35	1.437699
F4	8.19E-47	379.5462	0.037261	7.64E-27	0.122061
F5	0.060842	2.619147	35.00681	0.675395	342.5613
F6	0.204624	0.528366	2.41E-20	0.144584	0.020428
F7	4.99E-05	0.270907	0.006012	6.25E-05	0.002286
F8	338.2874	0.004667	715.4585	548.5262	840.3777
F9	0	210.3055	5.246117	0.965783	23.66112
F10	0	4.794991	7.33E-11	1.61E-15	0.511412
F11	0	9.33643	0.007596	0.001791	0.065966
F12	0.042397	0.134853	1.38E-21	0.005147	0.819071
F13	0.065208	0.065	5.13E-15	0.066354	0.009557
F14	0.701407	0.198483	0	0.796834	1.52E-12

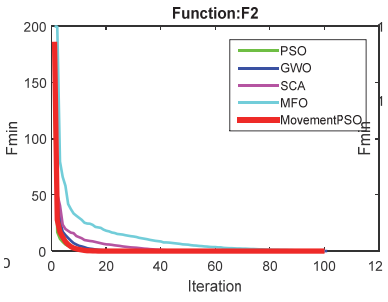
F15	3.06E-05	8.28E-07	0.000324	0.004998	0.003642
F16	4.12E-06	0.000439	6.77E-16	6.74E-10	1.55E-08
F17	0.00015	2.37E-06	0	6.38E-08	2.3E-08
F18	5.38E-06	0.000147	2.35E-15	1.49E-07	1.84E-07
F19	0.001865	4.19E-07	2.71E-15	0.001787	3.39E-08
F20	0.04548	0.003278	0.056497	0.064709	0.060492
F21	1.175414	0.123783	2.409565	1.52758	2.808354
F22	1.407452	1.155592	1.800816	1.804327	1.830623
F23	0.826681	1.495206	1.834882	0.97127	2.487936

3.6 Convergence Curve

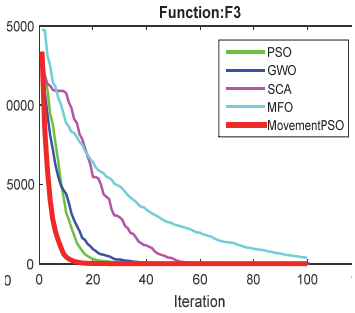
The Convergence curve has been done to test the speed of MPSO in comparison with other algorithms



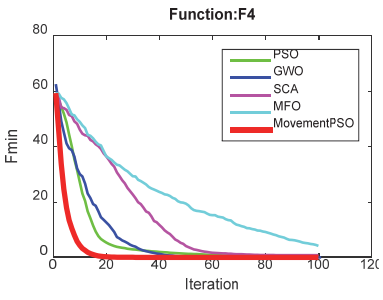
F1



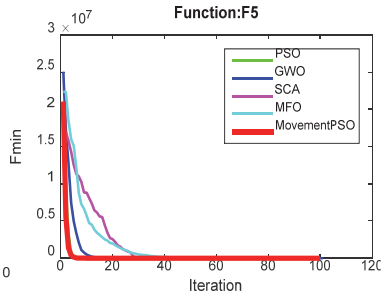
F2



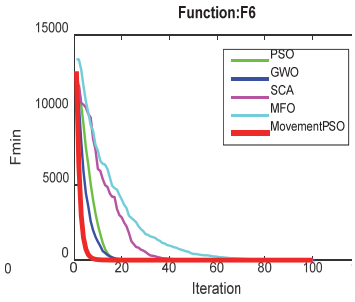
F3



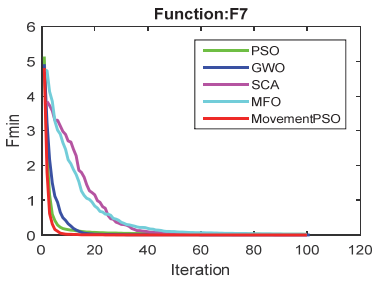
F4



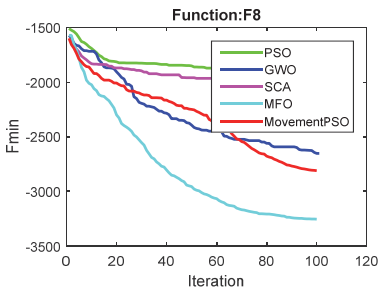
F5



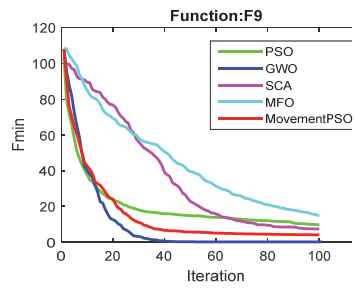
F6



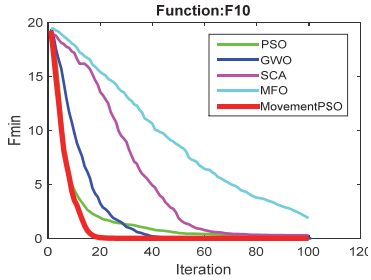
F7



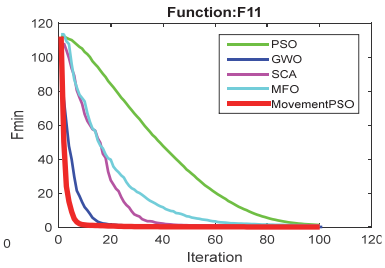
F8



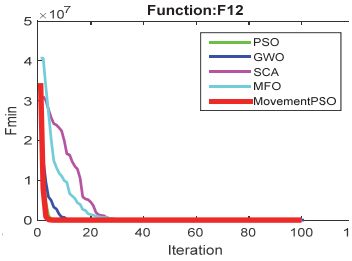
F9



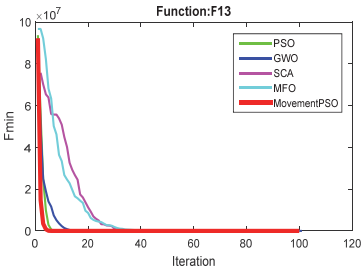
F10



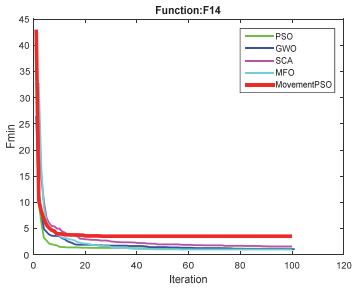
F11



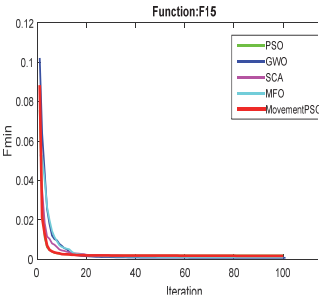
F12



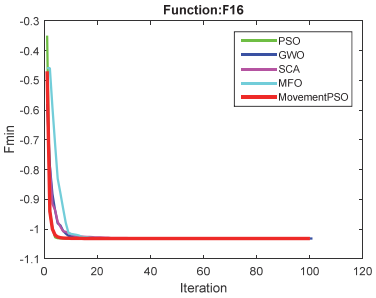
F13



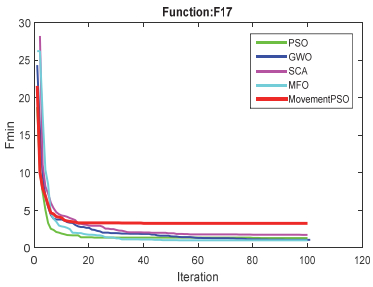
F14



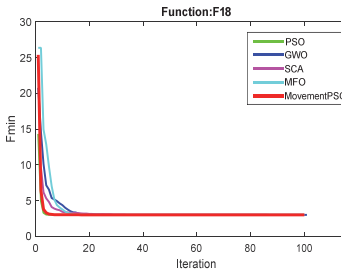
F15



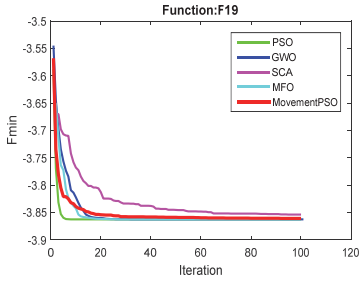
F16



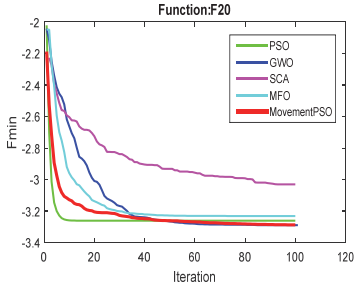
F17



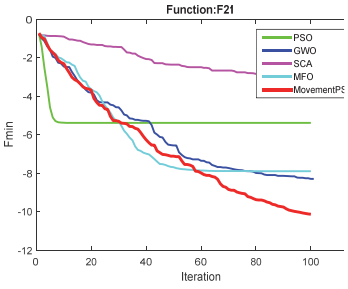
F18



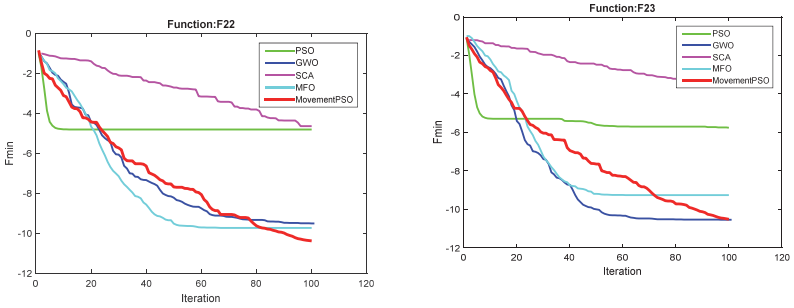
F19



F20



F21



F22 F23

Figure 7. Convergence curve of MPSO algorithm in comparison with GWO, SCA,, MFO and PSO.

It can be noticed from figure 7 that MPSO optimizer outperformed all of the four optimization algorithms (GWO, SCA, MFO, and PSO) and it was the fastest to find the best solution as shown in the Convergence curve chart for each iteration for the bench mark functions from F1 to F7 which represent Unimodal Functions (Shifted Sphere Function, Shifted Schwefel's Problem 1.2, Shifted Rotated High Conditioned Elliptic Function, Shifted Schwefel's Problem 1.2 with Noise in Fitness, Schwefel's Problem 2.6 with Global Optimum on Bounds) and F7 which is a Multimodal Basic Functions (Shifted Rotated Griewank's Function without Bounds),: Shifted Rosenbrock's Function, F7: Shifted Rotated Griewank's Function without Bounds this testifies that the proposed algorithm has a high exploitation ability.

But, in F8 we observed MFO algorithm better than Convergence curve MPSO, and F9 and F23 GWO algorithm better than Convergence curve MPSO algorithm also F14 and F17 PSO algorithm better than Convergence curve MPSO algorithm and we observed all fitness functions in F17 (PSO, GWO, SCA, MFO) are better than MPSO and in F16 all fitness Functions are similar. Where F14 is the Shifted Rotated Expanded Sca ffer's and F15 is the Hybrid Composition Function and F16 is Rotated Hybrid Composition space Function, F17 is the Rotated Hybrid Composition Function with Noise in Fitness

It can be also observed from figure 7 that MPSO optimizer succeed to be the fastest and most effective optimizer for the compared algorithms (GWO, SCA, MFO, MFO, and PSO when it is tested for the bench mark functions F10- F13, F15, F18, F20- F21, F23 which is from the Multimodal Functions and Hybrid Compositions where F10 is the Shifted Rotated Rastrigin's Function, F11 is Shifted Rotated Weierstrass Function, F12 is Schwefel's Problem, F13 is the Expanded Extended Griewank's plus Rosenbrock's Function, F15 is Hybrid Composition Function. F18 is Rotated Hybrid Composition Function, F20 is Rotated Hybrid Composition Function with the Global Optimum on the Bounds, F21 is Rotated Hybrid Composition Function

4. Case Study for MPSO to Design Problems

4.1 Welded Beam Design

This problem has four variables such as thickness of weld (h), length of attached part of bar (l), the height of the bar (t), and thickness of the bar (b). The mathematical formulation is as follows:

PSO algorithm, Coello GA Carlos A. and COELLO C. in 2000, Deb Deb K.in 2000 employed GA, GWO and used GSA to solve this problem. The comparison results are provided in Table 8. The results show that MPSO finds a design with the minimum cost compared to others.

Consider Minimize Subject to :

$$\vec{X} = [X_1, X_2, X_3, X_4] = [h, l, t, b]$$

$$F(\vec{X}) = 1.10471 X_1^2 X_2 + 0.04811 X_3 X_4 (14.0 + X_2)$$

$$g_1(\vec{X}) = \tau(\vec{X}) - \tau_{max} \leq 0$$

$$g_2(\vec{X}) = \sigma(\vec{X}) - \sigma_{max} \leq 0$$

$$g_3(\vec{X}) = \delta(\vec{X}) - \delta_{max} \leq 0$$

$$g_4(\vec{X}) = X_1 - X_4 \leq 0$$

$$g_5(\vec{X}) = P - Pc(\vec{X}) \leq 0$$

$$g_6(\vec{X}) = 0.125 - X_1 \leq 0$$

$g7(\vec{X})= F(\vec{X}) =1.10471X_1^2 X_2+0.04811X_3X_4(14.0+X_2)-05.0 \leq 0$

Variable range

$0.1 \leq X_1 \leq 0;$

$0.1 \leq X_2 \leq 10;$

$0.1 \leq X_3 \leq 10;$

$0.1 \leq X_4 \leq 2;$

Where

$$\tau(\vec{X})=\sqrt{(\tau')^2 + 2\tau'\tau''\frac{X_2}{2R} + (\tau'')^2}$$

$$\tau' = \frac{P}{\sqrt{2X_1}X_2}, \quad \tau'' =\frac{MR}{J}, \; M=P(L+ \frac{X_2}{2}) ;$$

$$R=\sqrt{\frac{X_2^2}{4} + (\frac{X_1+X_3}{2})^2} \; , J =2\left\{\sqrt{2X_1}X_2 \left[\frac{X_2^2}{4} + (\frac{X_1+X_3}{2})^2\right]\right\};$$

$$\sigma(\vec{X}) =\frac{6PL}{X_4 X_3^2} , \quad \delta(\vec{X}) = \frac{6PL^3}{E X_4 X_3^2},$$

$$P_c(\vec{X})=\frac{4.013E\sqrt{\frac{X_3^2+X_4^6}{36}}}{L^2}\left(1 - \frac{X_3}{2L}\frac{E}{4G}\right);$$

$P = 6000$ lb, $L = 14$ in., $\delta_{max} = 0.25$ in., $E = 30 \times 1^6$ psi, $G = 12 \times 10^6$ psi, $\tau_{max} = 13600$ psi, $\sigma_{max} = 30000$ psi

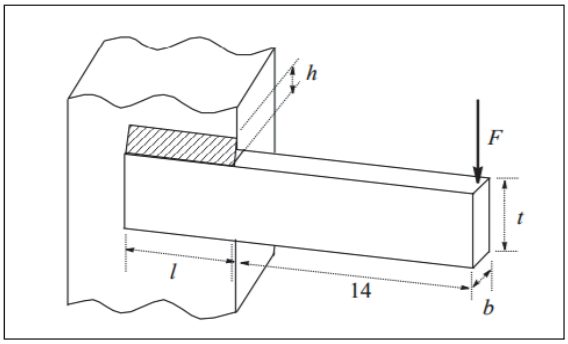


Figure8. Structure of welded beam design

Table 8. Comparison results

Algorithms	Optimum Variables				
	h	i	b	t	Optimum Cost
MPSO	0.201200	2.862001	10.03681	0.204423	1.6895
PSO	0.26780	4.72829	9.89287	0.204902	1.7926
GWO	0.205676	3.478377	9.03681	0.205778	1.72624
GA Coello)	N/A	N/A	N/A	N/A	1.8245
GA (Deb)	0.2489	6.1730	8.1789	0.2533	2.4331
GSA	0.182129	3.856979	10.00000	0.202376	1.879952

It can be noticed from Table 8 that MPSO optimizer gave good results in compasion with other algorithms (PSO,GWO, GA Coello, GA (Deb) and GSA) in solving the problem of Welded beam design and was able to find the Optimum cost in solving the equations discussed above related to this problem and the Optimum values obtained is 1.6895.

5. Conclusion

This work proposed an enhanced PSO optimization algorithm. 23 functions used to benchmark the performance of the proposed meta-heuristic for the features of exploration, exploitation, local optima avoidance. The findings described that MPSO has the ability to provide good results in comparing to well-known heuristics including GWO, MFO, and SCA. For the first set of functions (unimodal functions), MPSO algorithm showed good exploitation the exploration ability of MPSO was competitive too over the multimodal functions. Finally the composite functions showed local optima avoidance

Reference

Amjad, A. H., & Hussam, N. F. (2017). Supernova Optimizer: A Novel Natural Inspired Meta-Heuristic. *Modern Applied Science, 12*(1). 2018 ISSN 1913-1844 E-ISSN 1913-1852 Published by Canadian Center of Science and Education.

Baskar, S., & Suganthan, P. N. (2004). *A novel concurrent particle swarm optimization.* in Proc. IEEE Congr. Evol. Comput. [C], 1, 792-796.

Bergh, F., & Engelbrecht, A. (2004). A cooperative approach to particle swarm optimization. *IEEE Trans. Evol. Comput., 8*(3), 225-239.

Bonabeau, E., Dorigo, M., & Theraulaz, G. (1999). Swarm Intelligence: From Natural to Artificial System". Oxford University Press, New York.

Carlos, A., & Coello, C. (2000). Constraint-handling using an evolutionary multiobjective optimization technique. *Civil Engineering Systems, 17*, 319-346.

Chen, J., Qin, Z., Yu, L. et al. (2005). Particle Swarm Optimization with Local Search. in: Proc. IEEE.

Deb, K. (1991). Optimal design of a welded beam via genetic algorithms. *AIAA Journal, 29*, 2013-2015.

Deb, K. (2000). An efficient constraint handling method for genetic algorithms. *Computer Methods in Applied Mechanics and Engineering, 186*, 311-338.

Deneubourg, J. L., Aron, S., Goss, S., & Pasteels, J. M. (1990). The self-organizing exploratory pattern of the Argentine ant. *Journal of Insect Behavior, 3*, 159-16.

Di Caro, G., & Dorigo, M. (1998). AntNet: Distributed stigmergetic control for communications networks. *Journal of Artificial Intelligence Research, 9*, 317-365.

Di Caro, G., Ducatelle, F., Gambardella, L. M. (2005). AntHocNet: An adaptive nature-inspired algorithm for routing in mobile ad hoc networks. *European Transactions on Telecommunications, 16*(5), 443-455.

Dorigo, M., Maniezzo, V., & Colorni, A. (1996). Positive feedback as a search strategy. Technical Report 91-016, Dipartimento di Elettronica, Politecnico di Milano, Milan, Italy. Revised version published as: M. Dorigo, V. Maniezzo, and A. Colorni. Ant System: Optimization by a colony of cooperating agents. *IEEE Transactions on Systems, Man, and Cybernetics - Part B, 26*(1), 29-41.

Eberhart, R., Shi, Y., & Kennedy, J. (2001). Swarm Intelligence. San Mateo, CA: Morgan Kaufmann.

El-Abd, M., & Kamel, M. S. (2006). *A hierarchal cooperative particle swarm optimizer*. In Proc. IEEE Swarm Intell. Symp. [C], pp. 43-47.

Fan, S. K., Liang, Y. C., & Zahara, E. (2004). Hybrid Simplex Search and Particle Swarm Optimization for the Global Optimization of Multimodal Functions. *Engineering Optimization, 36*(4), 401-418.

Herbert, G. M. (2000). Space-time adaptive processing (stap) for wide-band airborne radar," in Record of the IEEE 2000 International radar conference, (Alexandria, Virginia, USA), pp. 620–625.

Høydal, T. O. (2001). Advanced Digital Radio Frequency Memory (DRFM) Technology – New Capabilities for Intelligent Radar Electronic Countermeasures. ATEDS/SA Symposium and Exibition, San Diego, USA, March.

Kata, S., & Kalos, A. (2004). *West D. A Hybrid Swam Optimizer for Efficient Parameter Estimation.* Proceedings of the IEEE Congress on Evolutionary Computation [C], 309-315.

Kennedy, J., & Mendes, R. (May 2002). *Population structure and particle swarm performance.* in Proc. IEEE Congr. Evol. Comput, 2, 1671-1676.

Kessentini, S., & Barchiesi, D. (2015). Particle Swarm Optimization with Adaptive Inertia Weight. *International Journal of Machine Learning and Computing, 5*(5), 368-373. https://doi.org/10.7763/ijmlc.2015.v5.535

Klemm, R. (2002.). Comparison between monostatic and bistatic antenna configurations for stap. *IEEE Transactions on Aerospace and Electronic Systems, 36*, 596–608.

Lothes, R. N., Szymanski, M. B., & Wiley, R. G. (1990). Radar Vulnerability to Jamming". Artech House, Inc.

Mahdavi, M., Fesanghary, M., & Damangir, E. (2007). An improved harmony search algorithm for solving optimization problems. *Applied Mathematics and Computation, 188*, 1567-1579.

Melvin, W., Callahan, M., & Wicks, M. (2002). Bistatic stap: application to airborne radar. In Proceedings of the IEEE Radar Conference, 2002, RadarCon-02, (Long Beach, California, USA), pp. 1-7.

Monson, C. K., & Sepp, I. K. D. (2004). The Kalman Swarm: A New Approach to Particle Motion in Swarm Optimization. Proceedings of the Genetic and Evolutionary Computation Conference. Springer, 140-150.

Parsopoulos, K. E., & Vrahatis, M. (2004). On the computation of all global minimizers through particle swarm optimization. *IEEE Trans. Evol. Comput, 8*, 211-224.

Pavlidis, N. G., Parsopoulos, K. E., & Vrahatis, M. N. (2005). Computing Nash Equilibria Through Computational Intelligence Methods. *Journal of Computational and Applied Mathematics, 175*(1), 113-136.

Richards, M., & Ventura, D. (2003). *Dynamic sociometry in particle swarm optimization.* International Conference on Computational Intelligence and Natural Computing [C], pp. 1557-1560.

Rizik, M. H., Al-Sayyed, H., Fakhouri, N., & Ali, R. (2017). Colin Pattinson, Polar Particle Swarm Algorithm for Solving Cloud Data Migration Optimization Problem. *Modern Applied Science, 11*(8). ISSN 1913-1844 E-ISSN 1913-1852, Published by Canadian Center of Science and Education.

Shi, Y., & Eberhart, R. (1998). A modified particle swarm optimizer. IEEE International Conference on Evolutionary Computation Proceedings. IEEE World Congress on Computational Intelligence (Cat. No.98TH8360). https://doi.org/10.1109/icec.1998.699146

Shuquan, L., & Kongguo, Z. (2008). *Research on multi-objective Optimization of lean construction project.* Proceedings of 2008 International Conference on Multimedia and InformationTechnology, MMIT2008, 480- 483.

Suganthan, P. N. (1999). *Particle swarm optimiser with neighbourhood operator.* In Proceedings of the IEEE Congress on Evolutionary Computation (CEC) [C], Piscataway, NJ, 1958-1962.

Surendra, L., et al. (2014). Analysis of Self Screening Jammer Parameters with RADAR Equation. *Int. Journal of Engineering Research and Applications, 4*(3)(Version 1), 205-207.

Vijayalakshmi, S., Sudha, D., Mercy, S., & Sigamani, K. K. D. (2014). Particle Swarm Optimization with Aging Leader and Challenges for Multiswarm Optimization. *International Journal of Advanced Research in Computer Engineering & Technology (IJARCET), 3*(3).

Wang, L., Kang, Q., & Wu, Q. (2007). *A Novel Self-organizing Particle Swarm Optimization based on Gravitation Field Model*. Proceedings of the American Control Conference [C], New York City, USA, July 11-13, 528-533.