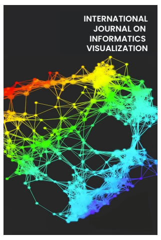
INTERNATIONAL JOURNAL
ON INFORMATICS VISUALIZATION

journal homepage : www.joiv.org/index.php/joiv



TPPSO: A Novel Two-Phase Particle Swarm Optimization

Tareq M. Shami [a], Mhd Amen Summakieh [b], Mohammed Alswaitti [c], Majan Abdullah Al Jahdhami [d], Abdul Manan Sheikh [d], Ayman A. El-Saleh [d,*]

[a] *Department of Electronic Engineering, University of York, York, U.K*
[b] *Faculty of Engineering, Multimedia University, Selangor, Malaysia*
[c] *Interdisciplinary Centre for Security, Reliability and Trust, University of Luxembourg, Luxembourg*
[d] *Department of Electronics and Communication Engineering, College of Engineering, A'Sharqiyah University, Ibra, Oman*
*Corresponding author: *ayman.elsaleh@asu.edu.om*

***Abstract*— Particle swarm optimization (PSO) is a stout and rapid searching algorithm that has been used in various applications. Nevertheless, its major drawback is the stagnation problem that arises in the later phases of the search process. To solve this problem, a proper balance between investigation and manipulation throughout the search process should be maintained. This article proposes a new PSO variant named two-phases PSO (TPPSO). The concept of TPPSO is to split the search process into two phases. The first phase performs the original PSO operations with linearly decreasing inertia weight, and its objective is to focus on exploration. The second phase focuses on exploitation by generating two random positions in each iteration that are close to the global best position. The two generated positions are compared with the global best position sequentially. If a generated position performs better than the global best position, then it replaces the global best position. To prove the effectiveness of the proposed algorithm, sixteen popular unimodal, multimodal, shifted, and rotated benchmarking functions have been used to compare its performance with other existing well-known PSO variants and non-PSO algorithms. Simulation results show that TPPSO outperforms the other modified and hybrid PSO variants regarding solution quality, convergence speed, and robustness. The convergence speed of TPPSO is extremely fast, making it a suitable optimizer for real-world optimization problems.**

***Keywords*— Particle swarm optimization; global optimization; swarm intelligence; exploration; evolutionary algorithms (EAs).**

Manuscript received 11 Dec. 2022; revised 19 Jun. 2023; accepted 8 Sep. 2023. Date of publication 30 Nov. 2023.

I. INTRODUCTION

In 1995, a promising evolutionary algorithm, namely Particle Swarm Optimization (PSO) to solve optimization problems was proposed [1], [2]. The PSO algorithm draws inspiration from the social relationship between birds flocking and fish schooling. Naturally, a swarm of birds' veers to follow a leader closer to the food source, guiding their flight in the open space. Birds' social behavior can be translated into an algorithm for resolving optimization problems. In this algorithm, particles form a swarm, with each particle representing a potential solution, and the swarm searches the space to find the best solution.

PSO is arguably one of the best optimization algorithms due to its ease of deployment, limited control parameters, and optimal results at a shorter computational time. In certain cases, it can outperform various evolutionary algorithms such as genetic algorithm (GA) [3] and ant colony optimization (ACO) [4]. However, PSO suffers from premature convergence [5], [6]. The PSO algorithm has two significant limitations. Firstly, the particles tend to stagnate in the later stages of the search process, leading to a lack of further improvements. Secondly, this issue arises due to an inadequate balance between exploration and exploitation. Exploration involves searching for a wide area in the solution space, while exploitation focuses on intensively searching around a promising region. To address these limitations, modifications have been made to the standard PSO (SPSO).

In this paper, we propose a TPPSO that includes a searching process, divided into two phases. The first phase focuses on exploration while the second phase focuses on exploitation. In the first phase, the standard PSO with linearly decreasing inertia weight is used. In the second phase, a new idea is introduced where two random positions located near the global best position is generated in each iteration. The two generated random positions are benchmarked with the global best position. Any of the two generated positions replace the global best position if it achieves better results. TPPSO still has a simple algorithm structure because the only modification in TPPSO is the addition of the two random positions.

The rest of this paper is organized as follows. Section 2 presents the related work in brief. The proposed algorithm is discussed in detail in section 3. In section 4, we tested the proposed PSO variant on a set of 16 benchmark functions and compared it to 6 existing PSO variants. Finally, section 5 summarizes this work.

II. Materials and Method

In this section, the related works and proposed algorithms are presented.

A. Related Works

The PSO is an algorithm consisting of a particle swarm where each particle acts as a candidate solution and an objective function evaluates each solution. Each particle i can fly in the search space and find better solutions by using its velocity and position vectors, which are defined as:

$$V_i = (v_{i1}, v_{i2}, \dots, v_{iD}), i = 1,2,\dots,N \quad (1)$$

$$X_i = (x_{i1}, x_{i2}, \dots, x_{iD}), i = 1,2,\dots,N \quad (2)$$

where the velocity vector is (V_i), the position vector is (X_i), the number of dimensions is (D), and the population size is (N). During the PSO process's initial stages, each particle's velocity and position are randomly set within predefined ranges. During the evolutionary process, the particles are attracted by their own historical best position $Pbest_i$ as well as by the best position found in the whole swarm $gbest$. Moreover, the velocity and position of the particles are updated using the following formulas:

$$v_{id} = v_{id} + c_1 rand_1(Pbest_{id} - x_{id}) + c_2 rand_2(gbest_d - x_{id}) \quad (3)$$

$$x_{id} = x_{id} + v_{id} \quad (4)$$

where c_1 and c_2 are the cognitive and social acceleration coefficients. $rand_1$ and $rand_2$ are two uniform random values generated within $[0,1]$.

PSO is a repetitive process, which means that each particle in the swarm will fly in the search space to find a better solution iteratively. If a particle finds a position that is better than the position in the previous iteration, it records it as $Pbest$ and $gbest$ is recorded as the $Pbest$, which has the best solution in the whole swarm. This process continues until the stopping condition is satisfied. Since the introduction of the standard PSO, many attempts have been made to propose PSO variants that can enhance the PSO performance. Three approaches of PSO variants have been widely used to improve the PSO performance by avoiding premature convergence.

The first approach improves the PSO performance by adjusting the controlling parameters of PSO [7]–[13]. Shi and Eberhart [14] modified the original velocity update Equation in (3) by introducing a new parameter called inertia weight and, denoted as w. Based on this modification, the velocity update equation becomes now in the following form:

$$v_{id} = wv_{id} + c_1 rand_1(Pbest_{id} - x_{id}) \\ + c_2 rand_2(gbest_d - x_{id}) \quad (5)$$

Increasing the value of the inertia weight increases exploration, while decreasing it increases exploitation. According to Shi and Eberhart [14] it is recommended to start the PSO run with an inertia weight value of 0.9 and linearly decrease it until it reaches 0.4 by the end of the run. This approach allows for a global search in the initial stages and gradually transitions towards a more localized search to refine the results. Shi and Eberhart named this PSO variant as PSO with linearly varying inertia weight (PSO-LVIW) and provided a mathematical formula for the linearly varying inertia weight, which can be written as follows:

$$w = (w_{max} - w_{min})\left(\frac{T-t}{T}\right) + w_{min} \quad (5)$$

where t the number of the current iteration is, T is the maximum number of iterations, w_{max} and w_{min} are the initial and final values of the inertia weight.

The other controlling parameter that can be adjusted are the acceleration coefficient c_1 and c_2. In [5], a PSO variant called a hierarchical PSO with a time-varying acceleration coefficient (HPSO-TVAC) is proposed. In HPSO-TVAC, the process starts with a large value of c_1 and a small value of c_2 which allows the particles to explore the search space. As the number of the current iteration increases, the value of c_1 decreases while the value of c_2 increases causing the particles to focus more on exploitation at the last part of the process. By only adjusting the controlling parameters, the particles still follow $gbest$ which results in premature convergence in case that $gbest$ is trapped into local optima.

The second approach hybridizes PSO with other optimization techniques to enhance the performance of PSO [15]–[26]. PSO has been hybridized in the literature with genetic operators like mutation [27], [28], crossover[29], and selection [30] or with other searching techniques such as DE [31], GA[32], ACO [33], and gravitational search algorithm (GSA) [26] . In [26], PSO was combined with GSA, resulting in a hybrid PSO named GPS. The velocity update equation in (GPS) algorithm is influenced by both the velocity of the Particle Swarm Optimization (PSO) and the acceleration of the Gravitational Search Algorithm (GSA). In [17], a centripetal accelerated PSO (CAPSO), which is a combination of PSO and Newton's law of motion is proposed. Each particle in CAPSO has two more specifications (acceleration and centripetal acceleration) besides velocity and position. Although hybridizing PSO with other optimization techniques enhances the PSO performance, the resultant PSO variant due to hybridization is more sophisticated than the original PSO.

The third approach implements different neighborhood topologies to improve diversity [34]–[36]. The most famous neighborhood topologies that have been extensively studied are the star, ring, and Von Neumann topologies. Basically, the swarm topology can be global (star), local (ring), or a combination of both. As explained in [37]–[39], the global version achieves the fastest convergence but cannot avoid being trapped in local optima whereas the local version increases the diversity but convergences slowly. In [15], a fully informed particle swarm (FIPS) is presented where the velocity of a particle i depends on $Pbest$ of each neighbor of i. The star topology and the ring topology were combined together to form a single PSO named as unified PSO (UPSO).

Implementing neighborhood topologies helps to avoid premature convergence but it convergences slowly since particles are less attracted to the $gbest$. PSO's simplicity and robustness have led to its wide application in various fields, including wireless communications and image processing. It has proven effective in solving diverse optimization problems [40]–[55].

B. The Proposed Algorithm

The TPPSO improves the PSO performance and eliminates its stagnation problem by maintaining a proper balance between exploration and exploitation during the search operation; PSO algorithm achieves effective results. The basic concept of TPPSO is to divide the total number of iterations of PSO into two phases, where each phase consists of a predefined number of iterations focusing on either exploration or exploitation. The first phase consists of α number of iterations, while the number of iterations in the second phase is the total number of iterations−α. In the first phase, TPPSO uses the standard PSO with linearly decreasing inertia weight to focus on exploration since the SPSO has a very strong ability to explore at the early stage of the PSO process. The second phase introduces a new concept that balances exploration and exploitation and prevents premature convergence. The new concept is to generate two random positions in each iteration close to the global best position, as shown in Equations (7) and Equation (8). To utilize the good position of the global best position, it is promising to search around this position seeking better solutions. To perform this, we develop Equations (7) and Equation (8) to search around the global best position to find solutions that can lead to the optimal solution.

The two generated positions are compared with $gbest$ sequentially. If any of the two generated positions achieve better results as compared to $gbest$, then the global best position is replaced by that generated position.

$$Position_1 = gbest(rand_3) \tag{6}$$

$$Position_2 = gbest + (rand_4) \tag{7}$$

where rand_3 and rand_4 are two uniform random variables in the range $[0,1]$. The pseudo-code of the proposed algorithm to solve a minimization problem is provided in Algorithm 1.

Algorithm 1 Pseudo-code of the TPPSO algorithm

```
1: Initialize the swarm size $N$, randomly generate the velocity
   and positions of each particle in the searching space
2: Phase one:
3: Perform the original PSO with decreasing inertia weight as in
     [56]
4: Stop when the number of iterations is $> \alpha$
5: Phase two:
6: Compute the fitness value of each particle
7: Update the best position for each particle $Pbest$
8: Update the global best position $gbest$
9: Generate the two random positions as in Equation (7) and
     Equation (8)
10:     if  fitness ($position_1$) < fitness ($gbest$) then
11:             $gbest = position_1$
12:             else
13:             $gbest = gbest$
14:     endif
15:     if   fitness ($position_2$) < fitness ($gbest$), then
16:             $gbest = position_2$
17:             else
18:             $gbest = gbest$
19:     endif
20: Update the velocity and positions of each particle using
     Equations (3) and (4)
21: Repeat Phases one and two until a stopping condition is
     satisfied (e.g., the maximum number of iterations is reached)
22: Return the best solution
```

Fig. 1 shows an example that illustrates the fundamental concept of the proposed algorithm. As seen in Fig. 1, the two generated positions ($Posistion_1$ and $Posistion_2$) must be located close to the global best position. In other words, these two positions are restricted from having values higher than the value of the global best position. Implementing this approach allows the generation of new particles that can focus on promising areas around the global best position, leading to better fitness values. In addition, focusing the search around the global best position, as in the proposed method, improves the search algorithm's exploitation ability. This is particularly important in the later stages of the search, as it leads to higher accuracy and faster speed.

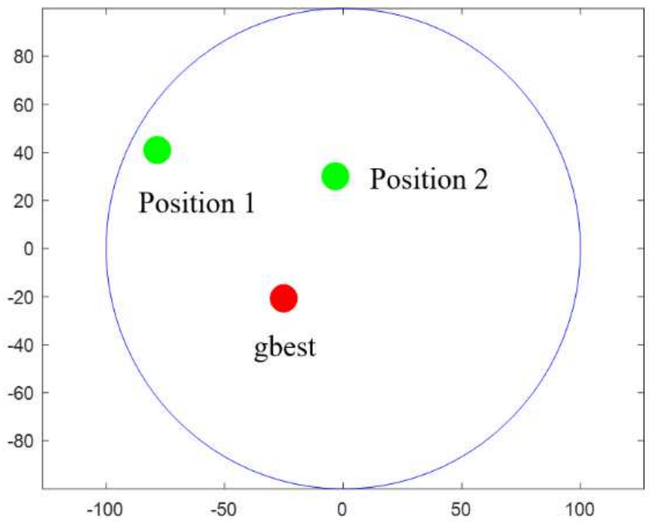


Fig. 1 2D-illustration of the generation boundaries of the two new positions.

III. Results and Discussion

A. Benchmark Functions

To test the performance of the proposed TPPSO, four groups of functions listed in Table 1 are used. These groups are unimodal functions (f_1-f_5), multimodal functions (f_6-f_11), shifted functions (f_12 and f_13) and rotated functions (f_14-f_16). Table 1 shows the search ranges (column 4), minimum values (column 5), and acceptable solutions (column 6) for each tested function. The acceptable solution is a value that shows whether a PSO variant can reach a satisfactory level or not. Table 1 presents the tested functions, including unimodal and multimodal functions that can evaluate the performance of TTPSO.

TABLE I
TEST FUNCTIONS

	Test function	**Name**	**Search space**	f_{min}	**Acceptance solution**
	$f_1(x)=\sum_{i=1}^{D}x_i^2$	Sphere	[-100,100]	0	1×10^{-6}
	$f_2(x)=\sum_{i=1}^{D-1}\left[100\left(x_{i+1}-x_i^2\right)^2+(x_i-1)^2\right]$	Rosenbrock	[-10,10]	0	100
Unimodal	$f_3(x)=\sum_{i=1}^{D}\lvert x_i\rvert+\prod_{i=1}^{D}\lvert x_i\rvert$	Schwefel P2.22	[-10,10]	0	1×10^{-6}
	$f_4(x)=\sum_{i=1}^{D}\left(\sum_{j=1}^{i}x_j\right)^2$	Quadric	[-100,100]	0	1×10^{-6}
	$f_5(x)=\sum_{i=1}^{D}\left(\lfloor x_i+0.5\rfloor\right)^2$	Step	[-100,100]	0	1×10^{-6}
	$f_6(x)=\sum_{i=1}^{D}\left[x_i^2-10cos(2\pi x_i)+10\right]$	Rastrigin	[-5.12,5.12]	0	100
	$f_7(x)=\sum_{i=1}^{D}\left[y_i^2-10cos(2\pi x_i)+10\right]$	Noncontinuous Rastrigin	[-5.12,5.12]	0	100
Multimodal	$f_8(x)=\frac{1}{4000}\sum_{i=1}^{D}x_i^2-\prod_{i=1}^{D}cos\left(\frac{x_i}{\sqrt{i}}\right)+1$	Griewank	[-600,600]	0	1×10^{-6}
	$f_9(x)=\frac{\pi}{D}\left\{10\sin(\pi y_1)+\sum_{i=1}^{D-1}(y_i-1)^2[1+\sin^2(\pi\,\mathrm{y}_{i+1})]+(\mathrm{y}_D-1)^2)\right\}$ $+\sum_{i=1}^{D}u(\mathrm{x}_i,10,100,4)$	Generalized Penalized	[-50,50]	0	1×10^{-6}
	$f_{10}(x)=-20\exp\left(-0.2\sqrt{\frac{1}{D}\sum_{i=1}^{D}x_i^2}-\exp\left(\frac{1}{D}\sum_{i=1}^{D}\cos 2\pi x_i\right)\right)+20+e$	Ackley	[-32,32]	0	1×10^{-6}
	$f_{11}=\sum_{i=1}^{D}(\sum_{k=0}^{k\,max}[a^k\,cos(2\pi b^k(x_i+0.5))])$ $-n\sum_{k=0}^{k\,max}[a^k\,cos(2\pi b^k.0.5)],a=0.5,b=3,k\,max=20$	Weierstrass	[-0.5,0.5]	0	1×10^{-6}
Shifted	$f_{12}(x)=\frac{1}{4000}\sum_{i=1}^{D}z_i^2-\prod_{i=1}^{D}cos\left(\frac{z_i}{\sqrt{i}}\right)+1+fbias_3,z=x-\mathrm{o}$	Shifted Griewank	[-600.600]	o	-100
	$f_{13}(x)=-20\exp\left(-0.2\sqrt{\frac{1}{D}\sum_{i=1}^{D}z_i^2}-\exp\left(\frac{1}{D}\sum_{i=1}^{D}\cos 2\pi z_i\right)\right)$ $+20+e+fbias_4,z=x-o$	Shifted Ackley	[-32,32]	o	-130
	$f_{14}(x)=\sum_{i=1}^{D}\left[y_i^2-10cos(2\pi y_i)+10\right],y=M\times x$	Rotated Rastrigin	[-5.12,5.12]	0	100
Rotated	$f_{15}(x)=\frac{1}{4000}\sum_{i=1}^{D}y_i^2-\prod_{i=1}^{D}cos\left(\frac{y_i}{\sqrt{i}}\right)+1,y=M\times x$	Rotated Griewank	[-600,600]	0	1×10^{-6}
	$f_{16}(x)=-20\exp\left(-0.2\sqrt{\frac{1}{D}\sum_{i=1}^{D}y_i^2}-\exp\left(\frac{1}{D}\sum_{i=1}^{D}\cos 2\pi y_i\right)\right)+20$ $+e,y=M\times x$	Rotated Ackley	[-32,32]	0	1×10^{-6}

* In f_7 , $y_i = \begin{cases} x_i, & |x_i| < 0.5 \\ \frac{round(2x_i)}{2}, & |x_i| \geq 0.5 \end{cases}$. In f_9 , $u(x_j,a,k,m) = \begin{cases} k(x_j - a)^m, & x_j > a \\ 0, & -a \leq x_j \leq a \\ k(-x_j - 1)^m, & x_j < -a \end{cases}$.

* In f_{12} and f_{13} , ***o*** is a shifted vector [57].

* In f_{14}–f_{16} , ***M*** is a transformation matrix [58].

B. Comparison Results

To prove the effectiveness of TPPSO, TPPSO is tested on 16 benchmarking functions and compared with 3 prominent PSO variants, namely SPSO, inertia weight PSO (IWPSO) [36], PSO-LVIW [14], HPSO-TVAC [5], and two well-known meta-heuristic algorithms: MTDE [59] and GWO [60]. The selected PSO variants have shown significant improvements in PSO performance. Besides, these PSO variants represent three different categories that have been discussed in section II. PSO-LVIW and HPSO-TVAC are two prominent PSO variants that control the PSO parameters to improve the PSO performance. Lastly, MTDE and GWO are two recent meta-heuristic algorithms with outstanding optimization performance.

C. Experimental Settings

The parameter settings for all the compared algorithms are listed in Table 2. The parameter settings for each algorithm are set as recommended by their original references. For TPPSO, α is chosen to be 50, which means that the first phase is performed in the first 50 iterations while the second phase is performed in the rest of the iterations. It is expected that TPPSO requires some iterations at the early stages to explore the search space. The amount of time to explore the search space can vary depending on the problem; however, performing 50 to 100 iterations can ensure efficient exploration. Thus, this work sets alpha to 50 for exploration purposes.

TABLE II
PARAMETER SETTINGS FOR THE PSO VARIANTS AND THE OTHER ALGORITHMS USED IN THIS EXPERIMENTAL STUDY

Approach	**Parameter**	**Setting**
All	Number of iterations	500
	Runs	30
	Swarm Size	40
SPSO	c_1,c_2	2, 2
PSO-LVIW	c_1,c_2,w	2, 2 , 0.9- 0.4
HPSO-TVAC	c_{1i},c_{1f},c_{2i},c_{2f}	2.5,0.5,0.5,2.5
MTDE	WinIter, *H*, initial, final, *Mu*, *μf*, *σ*	20,5,0.001,2, log(*D*),0.5,0.2
GWO	*a*	[2 0]
TPPSO	c_1, c_2, w, α	1.49,1.49,0.9-0.4,50

The population size and maximum number of iterations are kept consistent across all algorithms to ensure a fair comparison. For TPPSO, it is noteworthy that the swarm consists of 38 particles plus the two generated particles, resulting in a swarm size of 40.

D. Solution Accuracy

The solution accuracy and standard deviation (SD) on all the tested functions for D=30, D=100, and D=200 are reported in Table 3, Table 4, and Table 5, respectively. For each test problem, the algorithm that achieves the best performance (lowest value) is in the first rank, while the worst algorithm is in the sixth rank.

TABLE III
MINIMIZATION, RESULTS OF THE TESTED FUNCTIONS, IN TABLE. 1 (D=30)

Function		**TPPSO**	**SPSO**	**PSO-LVIW**	**HPSO-TVAC**	**MTDE**	**GWO**
	Mean	0	1.5571e+03	2.3224e-04	5.3543e-05	0.0282	1.4191e-30
f_1	SD	0	232.6220	3.1161e-04	5.5817e-05	0.0292	3.4283e-30
	Rank	1	6	4	3	5	2
	Mean	5.4972e-04	3.1748e+03	38.0283	56.2835	73.7438	26.8248
f_2	SD	9.8913e-04	691.3292	24.1431	38.5503	32.8107	0.8149
	Rank	1	6	3	4	5	2
	Mean	0	16.8016	0.0042	30.4333	0.0255	1.6316e-18
f_3	SD	0	1.2632	0.0043	14.5358	0.0384	1.6214e-18
	Rank	1	5	3	6	4	2
	Mean	5.0000e+07	4.4499e+03	372.5001	0.4216	363.2092	3.7773e-07
f_4	SD	1.3582e+08	722.5389	161.1148	0.2809	193.1975	6.8283e-07
	Rank	6	5	4	2	3	1
	Mean	0	1.4889e+03	2.5605e-04	0.0021	0.0465	0.7347
f_5	SD	0	163.2636	4.4336e-04	0.0017	0.0967	0.3695
	Rank	1	6	2	3	4	5
	Mean	0	202.1452	45.7019	26.0418	24.9509	1.8300
f_6	SD	0	15.9315	13.4910	9.5554	7.5813	2.9640
	Rank	1	6	5	4	3	2
	Mean	0	201.3163	46.3320	21.6541	23.6652	3.5404
f_7	SD	0	12.5967	13.2247	4.9287	6.6152	5.3267
	Rank	1	6	5	3	4	2
	Mean	0	14.5303	0.0184	0.0268	0.0649	0.0066
f_8	SD	0	1.7152	0.0170	0.0288	0.0645	0.0120
	Rank	1	6	3	4	5	2

Function		TPPSO	SPSO	PSO-LVIW	HPSO-TVAC	MTDE	GWO
f_9	Mean	1.5393e-04	14.3332	0.0104	0.1308	0.8957	0.0403
	SD	7.7397e-04	2.2265	0.0417	0.2274	0.8659	0.0260
	Rank	1	6	2	4	5	3
f_{10}	Mean	8.8818e-16	8.9471	0.0402	2.3013	1.6381	5.9745e-14
	SD	0	0.3360	0.1690	0.5667	0.5721	9.1263e-15
	Rank	1	6	3	5	4	2
f_{11}	Mean	0	18.4442	1.7916	4.0478	31.0632	5.9212e-15
	SD	0	1.5070	1.5749	1.5746	1.3599	5.3052e-15
	Rank	1	5	3	4	6	2
f_{12}	Mean	-179.9746	-145.6305	-158.2712	-179.9653	-179.7518	-162.0384
	SD	0.0399	21.8798	13.7675	0.0393	0.1873	16.6939
	Rank	1	6	5	2	3	4
f_{13}	Mean	-138.0809	-130.7126	-139.8459	-135.0970	-124.1309	-132.3962
	SD	0.8031	0.8917	0.4569	2.4616	5.0710	2.1531
	Rank	2	5	1	3	6	4
f_{14}	Mean	113.2833	233.0715	124.2203	288.2401	83.1584	167.2067
	SD	45.6016	18.3171	45.8939	42.3496	23.4971	70.3349
	Rank	2	5	3	6	1	4
f_{15}	Mean	-178.9469	-24.7025	-89.8141	-178.4281	-176.0862	-63.2488
	SD	0.2721	108.3838	75.7400	0.3813	2.4393	83.2179
	Rank	1	6	4	2	3	5
f_{16}	Mean	-119.0499	-118.9561	-118.9897	-119.0936	-118.9348	-118.9384
	SD	0.1004	0.0659	0.0697	0.1008	0.0601	0.0586
	Rank	2	4	3	1	6	5
Average rank		1.5	5.56	3.31	4.66	4.18	2.93
Final rank		**1**	6	3	5	4	2

TABLE IV
MINIMIZATION RESULTS OF THE TESTED FUNCTIONS IN TABLE 1 (D=100)

Function		TPPSO	SPSO	PSO-LVIW	HPSO-TVAC	MTDE	GWO
f_1	Mean	0	1.4322e+04	132.4667	61.3521	2.6160e+03	7.9548e-14
	SD	0	365.9316	29.6266	18.6388	693.2875	5.6941e-14
	Rank	1	6	4	3	5	2
f_2	Mean	0.6889	9.4012e+04	1.6874e+03	905.2238	9.0017e+03	97.4403
	SD	1.3342	1.4535e+04	3.0846e+03	217.0080	5.5723e+03	0.7104
	Rank	1	6	4	3	5	2
f_3	Mean	0	109.8841	32.2799	338.6000	33.8988	7.2193e-09
	SD	0	6.6518	79.7094	75.7676	6.3060	2.7829e-09
	Rank	1	5	3	6	4	2
f_4	Mean	6.7610e+09	8.2588e+04	4.7983e+04	1.1959e+05	2.1866e+04	199.9672
	SD	3.4801e+09	2.9740e+03	1.1870e+04	6.0372e+04	6.0384e+03	163.2223
	Rank	6	4	3	5	2	1
f_5	Mean	0	1.3991e+04	93.7115	21.0450	2.2731e+03	8.5891
	SD	0	907.8180	26.0172	17.2864	919.8658	1.0455
	Rank	1	6	4	3	5	2
f_6	Mean	0	951.5819	241.5384	127.6055	227.7619	7.9327
	SD	0	26.0928	31.0776	22.2214	37.7182	9.1829
	Rank	1	6	5	3	4	2
f_7	Mean	0	951.6198	219.9738	137.7836	230.3660	9.1897
	SD	0	14.0024	32.2415	27.7149	45.6752	6.6644
	Rank	1	6	4	3	5	2
f_8	Mean	0	142.6861	2.0673	1.5991	24.0153	4.7917e-14
	SD	0	7.6544	0.3373	0.1910	6.9443	3.3317e-14
	Rank	1	6	4	3	5	2
f_9	Mean	0.0554	8.8676e+05	7.4106	8.4648	2.3063e+03	0.2864
	SD	0.0149	9.7301e+05	2.7682	2.1066	3.3065e+03	0.0709
	Rank	1	6	3	4	5	2
f_{10}	Mean	8.8818e-16	12.7550	3.4417	10.1793	9.7578	2.4469e-08
	SD	0	0.0721	0.3792	0.7670	0.9278	9.5466e-09
	Rank	1	6	3	5	4	2
f_{11}	Mean	0	92.7434	33.5246	51.2011	136.0206	5.1661e-06
	SD	0	1.4782	6.4406	4.7726	21.4867	9.2979e-06
	Rank	1	5	3	4	6	2
f_{12}	Mean	-153.8430	145.7969	18.2130	-176.8263	76.0878	203.9014
	SD	12.7305	93.8520	84.5935	0.6018	65.0579	81.1562
	Rank	2	5	3	1	4	6
f_{13}	Mean	-127.4656	-127.0749	-134.5771	-122.9057	-125.4693	-125.7666
	SD	1.3826	0.6928	1.7883	0.4704	0.7489	0.5496
	Rank	2	3	1	6	5	4
f_{14}	Mean	2.9487e+03	1.3830e+03	1.7753e+03	1.6525e+03	1.7082e+03	1.5104e+03

Function		TPPSO	SPSO	PSO-LVIW	HPSO-TVAC	MTDE	GWO
	SD	273.5421	269.7822	190.9855	149.0092	126.7819	113.9867
	Rank	6	1	5	3	4	2
f_{15}	Mean	2.1413e+04	7.0081e+03	6.8055e+03	4.0869e+03	5.9378e+03	4.6191e+03
	SD	5.1119e+03	1.6157e+03	868.2218	1.3642e+03	593.9261	990.4402
	Rank	6	5	4	1	3	2
f_{16}	Mean	-118.2561	-118.5854	-118.6143	-118.3050	-118.5982	-118.5927
	SD	0.0594	0.0167	0.0276	0.0632	0.0195	0.0138
	Rank	6	4	1	5	2	3
Average rank		2.375	5	3.375	3.625	4.25	2.375
Final rank		1	5	2	3	4	1

TABLE V
MINIMIZATION RESULTS OF THE TESTED FUNCTIONS IN TABLE 1 (D=200)

`		TPPSO	SPSO	PSO-LVIW	HPSO-TVAC	MTDE	GWO
f_1	Mean	0	5.8899e+04	2.3614e+03	2.9993e+03	2.0182e+04	9.2093e-09
	SD	0	4.0496e+03	408.1431	496.8034	5.2114e+03	2.6866e-09
	Rank	1	6	3	4	5	2
f_2	Mean	3.4305	6.6707e+05	8.1355e+03	1.0625e+04	7.0923e+04	197.9074
	SD	6.7403	1.6182e+05	1.4991e+03	1.9523e+03	1.9856e+04	0.4328
	Rank	1	6	3	4	5	2
f_3	Mean	0	441.0067	163.3612	4.9937e+03	148.6501	9.6735e-06
	SD	0	36.7520	112.9599	854.6921	21.4207	1.9292e-06
	Rank	1	5	4	6	3	2
f_4	Mean	6.6705e+10	3.7899e+05	2.0448e+05	4.2709e+07	9.7116e+04	1.2677e+04
	SD	1.8059e+10	4.4378e+04	4.8161e+04	1.2218e+07	4.1539e+04	5.8830e+03
	Rank	6	4	3	5	2	1
f_5	Mean	0	5.5767e+04	2.5169e+03	246.8372	1.7616e+04	27.4161
	SD	0	1.9652e+03	444.6781	110.1380	3.5330e+03	1.9402
	Rank	1	6	4	3	5	2
f_6	Mean	0	2.0929e+03	725.6229	371.5672	915.0265	18.2434
	SD	0	6.3344	55.2308	27.6465	69.3331	7.6131
	Rank	1	6	4	3	5	2
f_7	Mean	0	2.0481e+03	749.2382	371.1106	909.3811	16.8214
	SD	0	112.1901	77.9294	43.2959	82.6924	8.9949
	Rank	1	6	4	3	5	2
f_8	Mean	0	578.1110	23.5791	27.8720	185.2158	5.0983e-09
	SD	0	16.2579	1.1681	5.1032	34.5019	2.9210e-09
	Rank	1	6	3	4	5	2
f_9	Mean	0.2357	3.0589e+07	1.4785e+03	43.1634	8.7800e+04	0.4620
	SD	0.0347	8.7002e+06	927.6033	16.9688	5.9280e+04	0.0536
	Rank	1	6	4	3	5	2
f_{10}	Mean	8.8818e-16	15.0721	6.2231	12.7309	12.3002	6.8640e-06
	SD	0	0.2812	0.3242	0.5037	0.5318	1.2885e-06
	Rank	1	6	3	5	4	2
f_{11}	Mean	0	213.5643	108.6252	139.4067	307.6020	0.0106
	SD	0	12.4614	8.2523	7.0823	24.6321	0.0023
	Rank	1	5	3	4	6	2
f_{12}	Mean	-179.4576	2.5810e+03	1.9450e+03	-176.3804	6.6474e+03	5.1459e+03
	SD	0.8174	749.3975	889.1476	2.0301	399.3929	300.6229
	Rank	1	4	3	2	6	5
f_{13}	Mean	-118.3003	-118.9938	-119.0615	-118.3657	-119.1880	-119.0543
	SD	0.0497	0.0685	0.0456	0.0573	0.0371	0.1161
	Rank	6	4	2	5	1	3
f_{14}	Mean	1.5294e+04	5.1201e+03	5.1038e+03	1.0151e+04	4.7748e+03	5.2578e+03
	SD	1.6277e+03	169.8610	87.2599	592.5911	79.9205	201.9053
	Rank	6	3	2	5	1	4
f_{15}	Mean	8.8804e+04	5.0024e+04	5.1477e+04	4.8672e+04	4.6875e+04	4.3713e+04
	SD	1.1213e+04	2.4274e+03	2.5277e+03	7.9278e+03	3.1219e+03	3.7427e+03
	Rank	6	4	5	3	2	1
f_{16}	Mean	-118.2745	-118.4886	-118.5002	-118.2885	-118.5066	-118.5094
	SD	0.0436	0.0088	0.0111	0.0382	0.0221	0.0148
	Rank	6	4	3	5	2	1
Average rank		2.56	5.37	3.31	3.5	3.87	2.18
Final rank		2	6	3	4	5	1

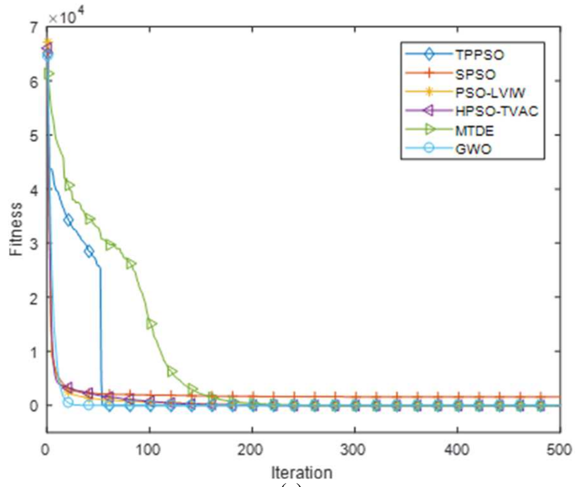
1) Solution Accuracy when D=30: The performance of TPPSO on all the unimodal functions was superior as compared to all other algorithms used in the test. Based on the results reported in Table 3, it is observable that TPPSO could achieve the exact optimal solution for (f_1, f_3, f_5). While the other algorithms had difficulties locating the global optimum for f_2, TPPSO managed to reach to a value that is very close to the global optimum. Unimodal functions are known to have a single minimum only, requiring more exploitation than explorations. Based on this fact, TPPSO, GWO, and MTDE proved they have strong exploitation abilities since they performed well on the unimodal functions.

Multimodal functions require strong exploration abilities since they have many local minimums. Besides the strong exploitation abilities of TPPSO on unimodal functions, TPPSO showed that its exploration abilities are strong as well. TPPSO outperformed the other algorithms for all the tested multimodal functions. It is noteworthy that TPPSO achieves the exact optimal solution for (f_6, f_7, f_8, f_{11}).

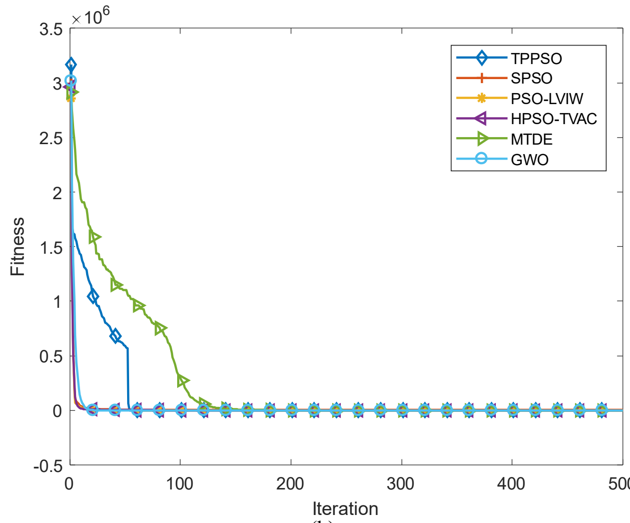
To further validate the effectiveness of TPPSO, it was tested on shifted and rotated functions. TPPSO successfully reached the acceptance solution on the two shifted functions f_{12} and f_{13} and its results on these two functions are comparable to the algorithm.

Overall, TPPSO achieves the best results for 12 functions $(f_1 - f_3, f_5 - f_{12}, f_{15})$ out of the 16 tested functions. Moreover, TPPSO had the ability to reach the exact optimal solution for 7 functions $(f_1, f_3, f_5, f_6, f_7, f_8, f_{11})$. As shown in Table 3, TPPSO has the best overall rank, followed by GWO, PSO-LVIW, MTDE, HPSO-TVAC, and SPSO.

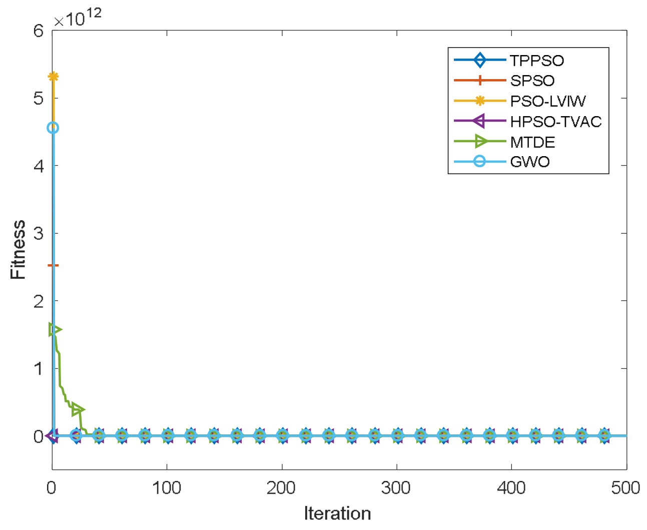
Figures 2, 3, and 4 show the convergence curves of the proposed TTPSO and the other algorithms for all tested functions when D=30. These figures show that TPPSO can converge toward the optimal solution without being stuck at the local optimal, unlike other algorithms. Moreover, the figures illustrate that PSO and MTDE algorithms are trapped in local optima very early, which restricts them from making any improvements.



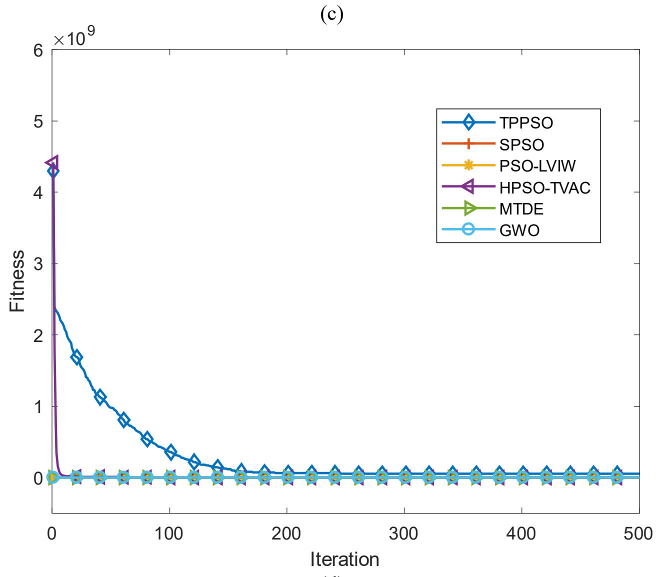
(a)



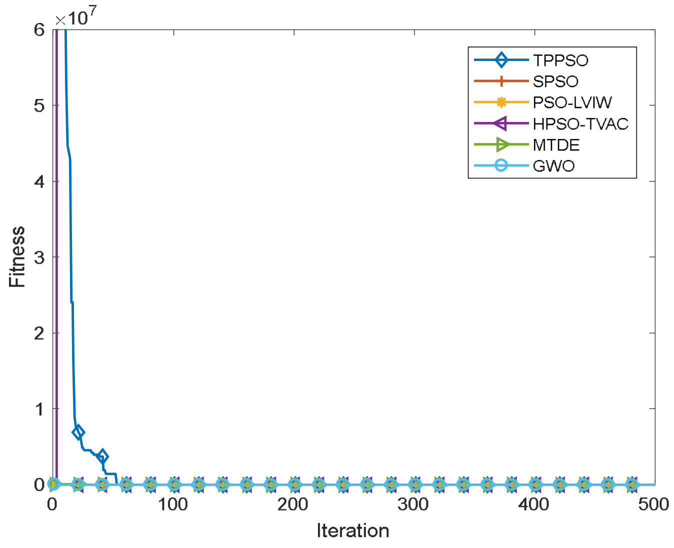
(b)



(c)



(d)



(e)

Fig. 2 Convergence graphs on the unimodal functions when D=30 (a) f_1 (b) f_2 (c) f_3 (d) f_4 (e) f_5

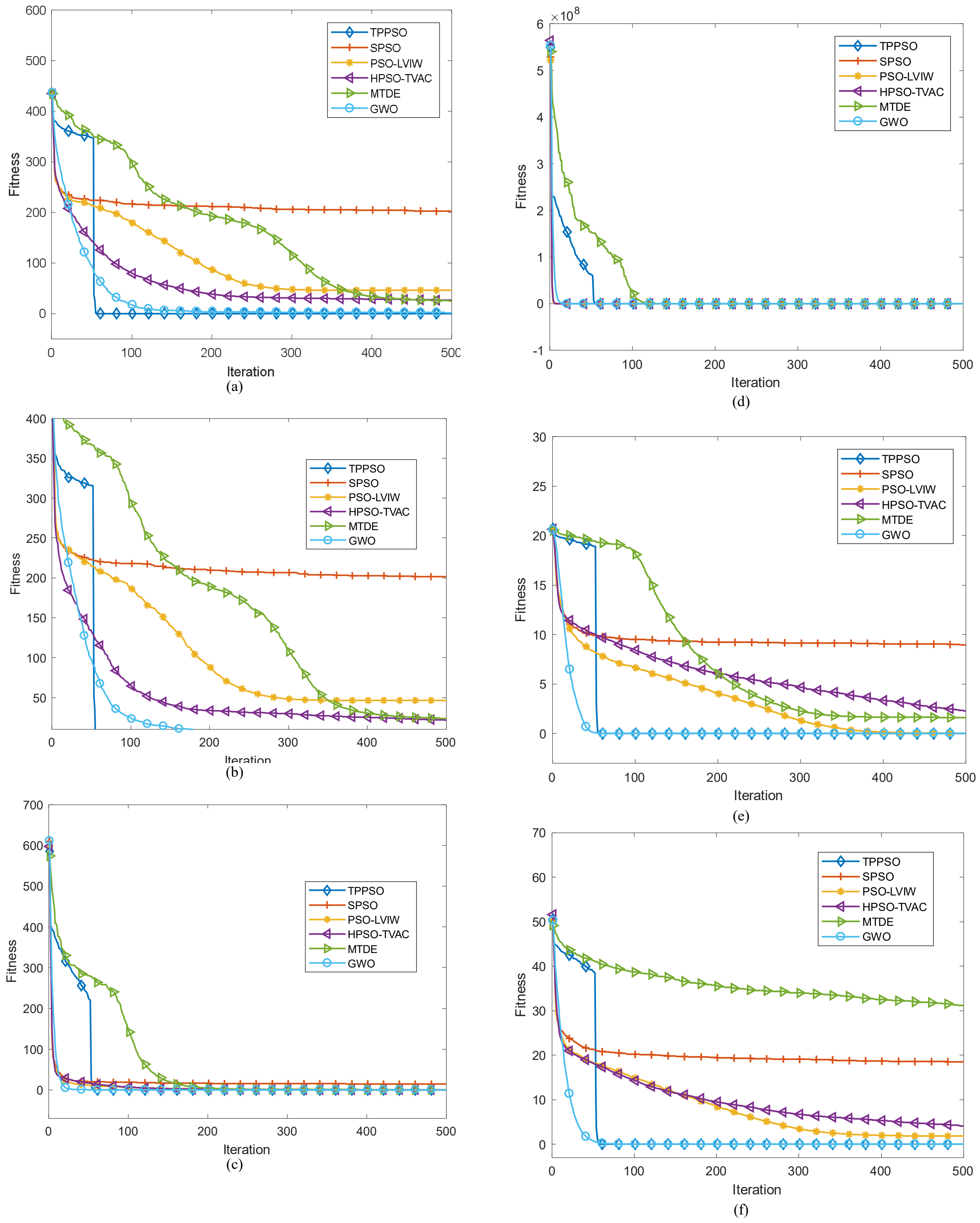
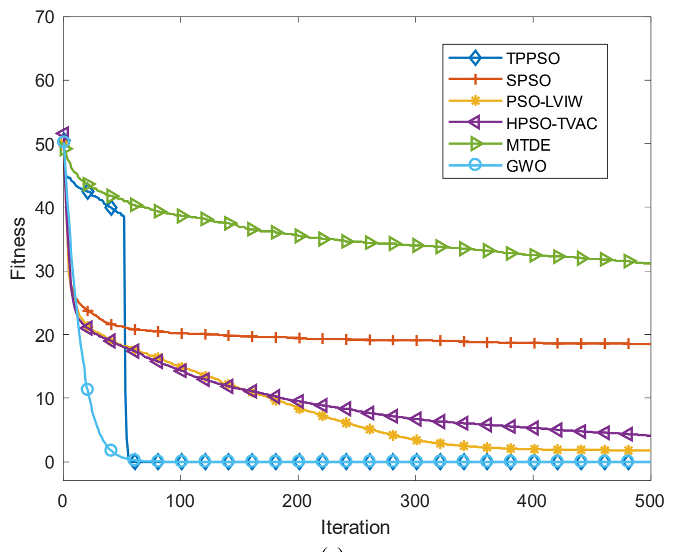
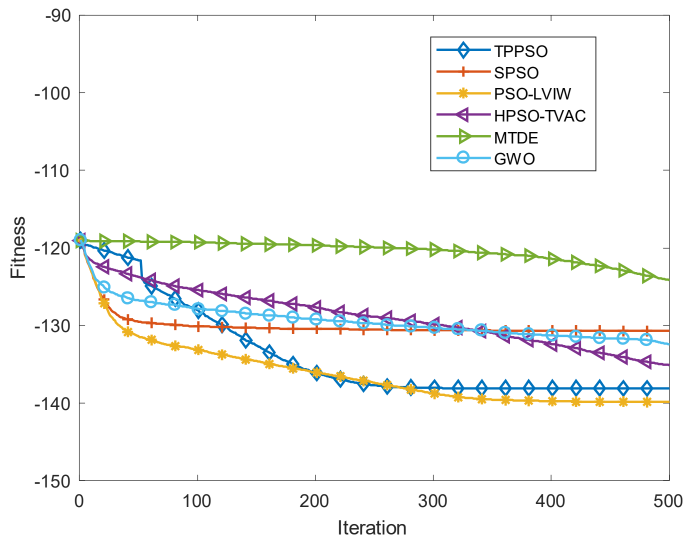


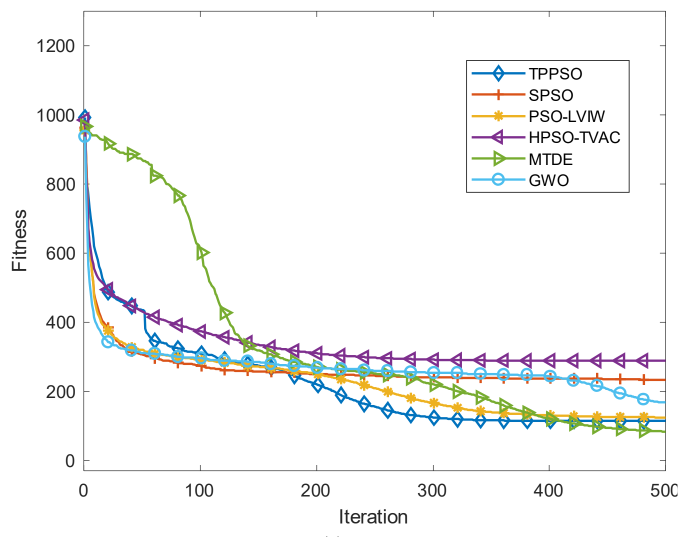
Fig. 3 Convergence graphs on the multimodal functions when D=30 (a) f_6 (b) f_7 (c) f_8 (d) f_9 (e) f_{10}(f) f_{11}.



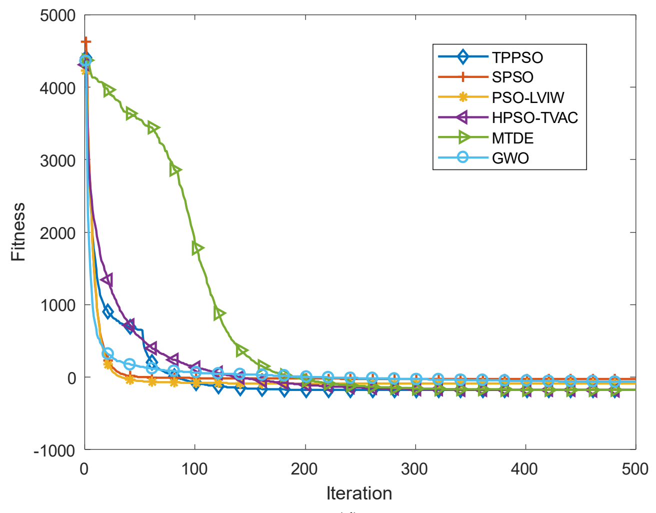
(a)



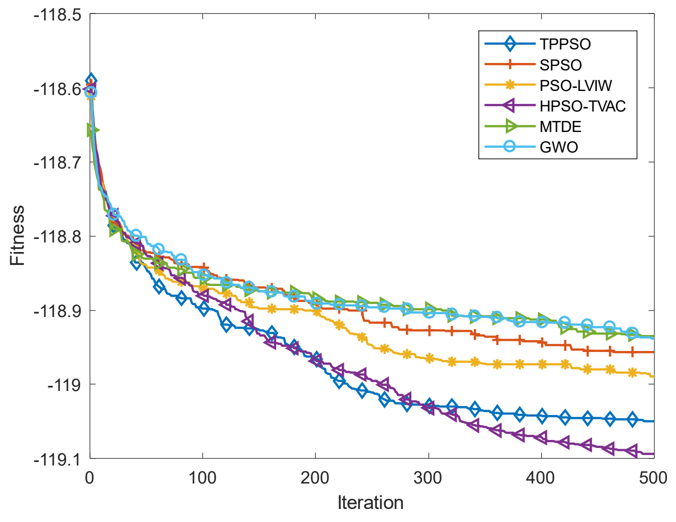
(b)



(c)



(d)



(e)

Fig. 4 Convergence graphs on the shifted and rotated functions when D=30 (a) f_{12} (b) f_{13},(c) f_{14} (d) f_{15} (e) f_{16}

2) *Solution Accuracy when D=100 and D=200:* As the number of dimensions increases and multiple local optima exist, the searching algorithm becomes more susceptible to be trapped in local optima. Therefore, testing the proposed PSO algorithm for high-dimensional cases is essential. TPPSO is tested on dimensions of 100 and 200, and the results are recorded in Table 4 and Table 5, respectively. The results in Table 4 and Table 5 validate that the TPPSO is not much affected if the number of dimensions increases. The results for all the tested functions $f_1, f_3, f_5 - f_8, f_{10}, f_{11}$ are the same results when D=30. For $f_2, f_9, f_{12} - f_{16}$ the solution accuracy of TPPSO is not much affected as compared to TPPSO with D=30. The performance of other compared algorithms decreases with increasing the number of dimensions.

E. Convergence Speed

TPPSO and the other algorithms are compared regarding convergence speed, as illustrated in Table 6. The convergence speed is determined by computing the mean number of iterations that are required to reach the acceptance solution shown in column 6 of Table 1. As seen in Table 6, TPPSO achieves the smallest mean number of iterations for f_1 , f_3, f_5

f_8, f_{10} and f_{11}. It is noteworthy that TPPSO required only a very small mean number of iterations (less than 123 mean number of iterations) to reach, the acceptable solution for all the functions expect f_4, f_9, f_{14} and f_{16}. This demonstrates that the convergence speed of TPPSO is extremely fast. When D=100 and D=200, TPPSO was not much affected except for f_9, f_{12}, f_{13}. Although TPPSO was not able to converge to the acceptance solution of f_9, it managed to obtain a value that is very close to it.

Table 6 shows the poor performance of SPSO, PSO-LVIW, HPSO-TVAC, and MTDE regarding convergence speed. GWO performed quite well for most of the tested functions. Overall, TPPSO outperforms all the other Algorithms in terms of convergence speed.

TABLE VI
CONVERGENCE SPEED OF THE TESTED FUNCTIONS IN TABLE 1 (D=30)

Function		TPPSO	SPSO	PSO-LVIW	HPSO-TVAC	MTDE	GWO
f_1	Mean iterations	59	-	-	-	-	90
	Rank	1	-	-	-	-	2
f_2	Mean iterations	54	-	274	368	385	36
	Rank	2	-	3	4	5	1
f_3	Mean iterations	57	-	-	-	-	115
	Rank	1	-	-	-	-	2
f_4	Mean iterations	-	-	-	-	-	426
	Rank	-	-	-	-	-	1
f_5	Mean iterations	63	-	-	-	-	-
	Rank	1	-	-	-	-	-
f_6	Mean iterations	53	-	185	79	313	48
	Rank	2	-	4	3	5	1
f_7	Mean iterations	53	-	186	71	307	49
	Rank	2	-	4	3	5	1
f_8	Mean iterations	59	-	-	-	-	-
	Rank	1	-	-	-	-	-
f_9	Mean iterations	-	-	-	-	-	-
	Rank	-	-	-	-	-	-
f_{10}	Mean iterations	62	-	-	-	-	124
	Rank	1	-	-	-	-	2
f_{11}	Mean iterations	64	-	-	-	-	149
	Rank	1	-	-	-	-	2
f_{12}	Mean iterations	56	23	22	78	160	35
	Rank	4	2	1	5	6	3
f_{13}	Mean iterations	123	93	34	310	-	272
	Rank	3	2	1	5	-	4
f_{14}	Mean iterations	-	-	-	-	435	-
	Rank	-	-	-	-	1	-
f_{15}	Mean iterations	83	53	33	141	190	217
	Rank	3	2	1	4	5	6
f_{16}	Mean iterations	-	-	-	-	-	-
	Rank	-	-	-	-	-	-

IV. CONCLUSION

This paper presented a new PSO variant named TPPSO that splits the search process into two stages. The first stage focuses on exploration, whereas the second focuses more on exploitation. The first phase performs the same operations as the original PSO with decreasing inertia weight. In the second phase, two random positions that are closely benchmarked with the global best position are generated and compared sequentially with the global best position in each iteration. Any of the two generated random positions replace the global best position in case it achieves better results. This concept helps to maintain a proper balance between exploration and exploitation.

To validate the effectiveness of the proposed algorithm, it was tested on 16 unimodal, multimodal, shifted, and rotated functions. The simulation results show that TPPSO is an effective and efficient PSO variant. The performance of TPPSO in terms of solution accuracy, convergence, speed, and reliability is superior as compared to the other existing PSO, variants and well-known meta-heuristic algorithms such as GWO, and MTDE. In addition, TPPSO does not add any complexity to the original PSO structure, as the only modification in TPPSO is the addition of the two random positions. In future work, TPPSO can be used to solve real-world power electronics and telecommunication optimization problems.

ACKNOWLEDGMENTS

This research work was supported in part by the Research Seed Fund of A'Sharqiyah University in the Sultanate of Oman.

REFERENCES

[1] J. Kennedy and R. Eberhart, "Particle swarm optimization," Proceedings of ICNN'95 - International Conference on Neural Networks, doi: 10.1109/icnn.1995.488968.

[2] R. Eberhart and J. Kennedy, "A new optimizer using particle swarm theory," MHS'95. Proceedings of the Sixth International Symposium on Micro Machine and Human Science, doi: 10.1109/mhs.1995.494215.

[3] E. Shahamatnia, I. Dorotovič, J. M. Fonseca, and R. A. Ribeiro, "An evolutionary computation based algorithm for calculating solar differential rotation by automatic tracking of coronal bright points," J. Sp. Weather Sp. Clim., vol. 6, 2016, doi: 10.1051/swsc/2016010.

[4] C. H. Jang, F. Hu, F. He, J. Li, and D. Zhu, "Low-Redundancy Large Linear Arrays Synthesis for Aperture Synthesis Radiometers Using Particle Swarm Optimization," IEEE Trans. Antennas Propag., vol. 64, no. 6, pp. 2179–2188, Jun. 2016, doi: 10.1109/TAP.2016.2543755.

[5] A. Ratnaweera, S. K. Halgamuge, and H. C. Watson, "Self-organizing hierarchical particle swarm optimizer with time-varying acceleration coefficients," IEEE Trans. Evol. Comput., vol. 8, no. 3, pp. 240–255, Jun. 2004, doi: 10.1109/TEVC.2004.826071.

[6] K. Luu, M. Noble, A. Gesret, N. Belayouni, and P. F. Roux, "A parallel competitive Particle Swarm Optimization for non-linear first arrival traveltime tomography and uncertainty quantification," Comput. Geosci., vol. 113, pp. 81–93, Apr. 2018, doi: 10.1016/j.cageo.2018.01.016.

[7] M. Abdulkadir, A. H. M. Yatim, and S. T. Yusuf, "An improved PSO-based MPPT control strategy for photovoltaic systems," Int. J. Photoenergy, vol. 2014, 2014, doi: 10.1155/2014/818232.

[8] P. Melin, F. Olivas, O. Castillo, F. Valdez, J. Soria, and M. Valdez, "Optimal design of fuzzy classification systems using PSO with dynamic parameter adaptation through fuzzy logic," Expert Syst. Appl., vol. 40, no. 8, pp. 3196–3206, Jun. 2013, doi: 10.1016/j.eswa.2012.12.033.

[9] L. Zhang, Y. Tang, C. Hua, and X. Guan, "A new particle swarm optimization algorithm with adaptive inertia weight based on Bayesian techniques," Appl. Soft Comput. J., vol. 28, pp. 138–149, 2015, doi: 10.1016/j.asoc.2014.11.018.

[10] Q. Liu, "Order-2 stability analysis of particle swarm optimization," Evol. Comput., vol. 23, no. 2, pp. 187–216, Jun. 2015, doi: 10.1162/EVCO_a_00129.

[11] K. R. Harrison, A. P. Engelbrecht, and B. M. Ombuki-Berman, "Optimal parameter regions and the time-dependence of control parameter values for the particle swarm optimization algorithm," Swarm Evol. Comput., vol. 41, pp. 20–35, Aug. 2018, doi: 10.1016/j.swevo.2018.01.006.

[12] M. B. Shafik, H. Chen, G. I. Rashed, R. A. El-Sehiemy, M. R. Elkadeem, and S. Wang, "Adequate topology for efficient energy resources utilization of active distribution networks equipped with soft open points," IEEE Access, vol. 7, pp. 99003–99016, 2019, doi: 10.1109/ACCESS.2019.2930631.

[13] H. Liu, X. W. Zhang, and L. P. Tu, "A modified particle swarm optimization using adaptive strategy," Expert Syst. Appl., vol. 152, Aug. 2020, doi: 10.1016/j.eswa.2020.113353.

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

[15] Institute of Electrical and Electronics Engineers and IEEE Computational Intelligence Society, 2020 IEEE Congress on Evolutionary Computation (CEC) : 2020 conference proceedings. 2020, 2020.

[16] P. J. Angeline, "Using selection to improve particle swarm optimization," 1998 IEEE International Conference on Evolutionary Computation Proceedings. IEEE World Congress on Computational Intelligence (Cat. No.98TH8360), doi: 10.1109/icec.1998.699327.

[17] Z. Beheshti and S. M. Siti, "CAPSO: Centripetal accelerated particle swarm optimization," Inf. Sci. (Ny)., vol. 258, pp. 54–79, Feb. 2014, doi: 10.1016/j.ins.2013.08.015.

[18] J. Dash, B. Dam, and R. Swain, "Optimal design of linear phase multi-band stop filters using improved cuckoo search particle swarm optimization," Appl. Soft Comput. J., vol. 52, pp. 435–445, Mar. 2017, doi: 10.1016/j.asoc.2016.10.024.

[19] Wen-Jun Zhang and Xiao-Feng Xie, "DEPSO: hybrid particle swarm with differential evolution operator," SMC'03 Conference Proceedings. 2003 IEEE International Conference on Systems, Man and Cybernetics. Conference Theme - System Security and Assurance (Cat. No.03CH37483), doi: 10.1109/icsmc.2003.1244483.

[20] N. Higashi and H. Iba, "Particle swarm optimization with Gaussian mutation," Proceedings of the 2003 IEEE Swarm Intelligence Symposium. SIS'03 (Cat. No.03EX706), doi: 10.1109/sis.2003.1202250.

[21] P. S. Andrews, "An Investigation into Mutation Operators for Particle Swarm Optimization," 2006 IEEE International Conference on Evolutionary Computation, doi: 10.1109/cec.2006.1688424.

[22] P. S. Shelokar, P. Siarry, V. K. Jayaraman, and B. D. Kulkarni, "Particle swarm and ant colony algorithms hybridized for improved continuous optimization," Appl. Math. Comput., vol. 188, no. 1, pp. 129–142, May 2007, doi: 10.1016/j.amc.2006.09.098.

[23] Y. P. Chen, W. C. Peng, and M. C. Jian, "Particle swarm optimization with recombination and dynamic linkage discovery," IEEE Trans. Syst. Man, Cybern. Part B Cybern., vol. 37, no. 6, pp. 1460–1470, Dec. 2007, doi: 10.1109/TSMCB.2007.904019.

[24] Y. T. Kao and E. Zahara, "A hybrid genetic algorithm and particle swarm optimization for multimodal functions," Appl. Soft Comput. J., vol. 8, no. 2, pp. 849–857, Mar. 2008, doi: 10.1016/j.asoc.2007.07.002.

[25] M. S. Kiran, M. Gündüz, and Ö. K. Baykan, "A novel hybrid algorithm based on particle swarm and ant colony optimization for finding the global minimum," Appl. Math. Comput., vol. 219, no. 4, pp. 1515–1521, Nov. 2012, doi: 10.1016/j.amc.2012.06.078.

[26] H. C. Tsai, Y. Y. Tyan, Y. W. Wu, and Y. H. Lin, "Gravitational particle swarm," Appl. Math. Comput., vol. 219, no. 17, pp. 9106–9117, 2013, doi: 10.1016/j.amc.2013.03.098.

[27] T. Jamrus, C. F. Chien, M. Gen, and K. Sethanan, "Hybrid Particle Swarm Optimization Combined With Genetic Operators for Flexible Job-Shop Scheduling Under Uncertain Processing Time for Semiconductor Manufacturing," IEEE Trans. Semicond. Manuf., vol. 31, no. 1, pp. 32–41, Feb. 2017, doi: 10.1109/TSM.2017.2758380.

[28] M. Sharma and J. K. Chhabra, "Sustainable automatic data clustering using hybrid PSO algorithm with mutation," Sustain. Comput. Informatics Syst., vol. 23, pp. 144–157, Sep. 2019, doi: 10.1016/j.suscom.2019.07.009.

[29] Y. Heryadi, "A Hybrid Particle Swarm Optimization With Crossover and Mutation of Genetic Algorithm for Solving the Wide Constraint Problem," 2019.

[30] Y. Ding, K. Zhou, and W. Bi, "Feature selection based on hybridization of genetic algorithm and competitive swarm optimizer," Soft Comput., vol. 24, no. 15, pp. 11663–11672, Aug. 2020, doi: 10.1007/s00500-019-04628-6.

[31] N. Kumar Yadav, "Hybridization of Particle Swarm Optimization with Differential Evolution for Solving Combined Economic Emission Dispatch Model for Smart Grid," 2019.

[32] G. F. Fan, L. L. Peng, X. Zhao, and W. C. Hong, "Applications of hybrid EMD with PSO and GA for an SVR-based load forecasting model," Energies, vol. 10, no. 11, Nov. 2017, doi: 10.3390/en10111713.

[33] S. A. Mogaji, B. K. Alese, A. O. Adetunmbi, M. S. Alaba, A. B. Kayode, and A. Adebayo, "Validation of Hybridized Particle Swarm Optimization (PSO) Algorithm with the Pheromone Mechanism of Ant Colony Optimization (ACO) using Standard Benchmark Function. Securing Networks and Cyber-physical Systems View project Validation of Hybridized Partic," 2018. [Online]. Available: www.caeaccess.org

[34] R. Mendes, J. Kennedy, and J. Neves, "The fully informed particle swarm: Simpler, maybe better," IEEE Trans. Evol. Comput., vol. 8, no. 3, pp. 204–210, Jun. 2004, doi: 10.1109/TEVC.2004.826074.

[35] X. Li, "Niching without niching parameters: Particle swarm optimization using a ring topology," IEEE Trans. Evol. Comput., vol. 14, no. 1, pp. 150–169, Feb. 2009, doi: 10.1109/TEVC.2009.2026270.

[36] J. Kennedy and K.-J. Gov, "Population Structure and Particle Swarm Performance," 2002.

[37] J. Kennedy and K.-J. Gov, "Small Worlds and Mega-Minds: Effects of Neighborhood Topology on Particle Swarm Performance," 1999.

[38] A. Lin, W. Sun, H. Yu, G. Wu, and H. Tang, "Global genetic learning particle swarm optimization with diversity enhancement by ring topology," Swarm Evol. Comput., vol. 44, pp. 571–583, Feb. 2019, doi: 10.1016/j.swevo.2018.07.002.

[39] N. Lynn, M. Z. Ali, and P. N. Suganthan, "Population topologies for particle swarm optimization and differential evolution," Swarm Evol. Comput., vol. 39, pp. 24–35, Apr. 2018, doi: 10.1016/j.swevo.2017.11.002.

[40] X. Hao, N. Yao, J. Wang, and L. Wang, "Distributed resource allocation optimisation algorithm based on particle swarm optimisation in wireless sensor network," IET Commun., vol. 14, no. 17, pp. 2990–2999, Oct. 2020, doi: 10.1049/iet-com.2020.0368.

[41] A. A. El-Saleh, T. M. Shami, R. Nordin, M. Y. Alias, and I. Shayea, "Multi-objective optimization of joint power and admission control in cognitive radio networks using enhanced swarm intelligence," Electron., vol. 10, no. 2, pp. 1–27, Jan. 2021, doi: 10.3390/electronics10020189.

[42] O. Evsutin, A. Shelupanov, R. Meshcheryakov, D. Bondarenko, and A. Rashchupkina, “The algorithm of continuous optimization based on the modified cellular automaton,” Symmetry (Basel)., vol. 8, no. 9, 2016, doi: 10.3390/sym8090084.

[43] O. Almomani, “A Hybrid Model Using Bio-Inspired Metaheuristic Algorithms for Network Intrusion Detection System,” Comput. Mater. Contin., vol. 68, no. 1, pp. 409–429, Mar. 2021, doi: 10.32604/cmc.2021.016113.

[44] M. H. Alkinani, E. A. Zanaty, and S. M. Ibrahim, “Medical image compression based on wavelets with particle swarm optimization,” Comput. Mater. Contin., vol. 67, no. 2, pp. 1577–1593, 2021, doi: 10.32604/cmc.2021.014803.

[45] J. Wang, Y. Gao, C. Zhou, R. Simon Sherratt, and L. Wang, “Optimal coverage multi-path scheduling scheme with multiple mobile sinks for WSNs,” Comput. Mater. Contin., vol. 62, no. 2, pp. 695–711, 2020, doi: 10.32604/cmc.2020.08674.

[46] E. N. Al-Khanak et al., “A heuristics-based cost model for scientific workflow scheduling in cloud,” Comput. Mater. Contin., vol. 67, no. 3, pp. 3265–3282, Mar. 2021, doi: 10.32604/cmc.2021.015409.

[47] M. El Mamoun, Z. Mahmoud, and S. Kaddour, “SVM model selection using PSO for learning handwritten Arabic characters,” Comput. Mater. Contin., vol. 61, no. 3, pp. 995–1008, 2019, doi: 10.32604/cmc.2019.08081.

[48] S. K. Gopalakrishnan, S. Kinattingal, S. P. Simon, and K. A. Kumar, “Enhanced energy harvesting from shaded PV systems using an improved particle swarm optimisation,” IET Renew. Power Gener., vol. 14, no. 9, pp. 1471–1480, Jul. 2020, doi: 10.1049/iet-rpg.2019.0936.

[49] H. Xiang, M. Peng, Y. Sun, and S. Yan, “Mode Selection and Resource Allocation in Sliced Fog Radio Access Networks: A Reinforcement Learning Approach,” IEEE Trans. Veh. Technol., vol. 69, no. 4, pp. 4271–4284, Apr. 2020, doi: 10.1109/TVT.2020.2972999.

[50] D. T. C. Lai, M. Miyakawa, and Y. Sato, “Semi-supervised data clustering using particle swarm optimisation,” Soft Comput., vol. 24, no. 5, pp. 3499–3510, Mar. 2020, doi: 10.1007/s00500-019-04114-z.

[51] T. R. Farshi, J. H. Drake, and E. Özcan, “A multimodal particle swarm optimization-based approach for image segmentation,” Expert Syst. Appl., vol. 149, Jul. 2020, doi: 10.1016/j.eswa.2020.113233.

[52] T. Gao, B. Cao, and M. Zhang, “Multiobjective Complex Network Clustering Based on Dynamical Decomposition Particle Swarm Optimization,” IEEE Access, vol. 8, pp. 32341–32352, 2020, doi: 10.1109/ACCESS.2020.2972123.

[53] B. Kizielewicz and W. Sałabun, “A new approach to identifying a multi-criteria decision model based on stochastic optimization techniques,” Symmetry (Basel)., vol. 12, no. 9, Sep. 2020, doi: 10.3390/SYM12091551.

[54] C. Qin and X. Gu, “Article improved PSO algorithm based on exponential center symmetric inertiaweight function and its application in infrared image enhancement,” Symmetry (Basel)., vol. 12, no. 2, Feb. 2020, doi: 10.3390/sym12020248.

[55] Z. Ma, X. Yuan, S. Han, D. Sun, and Y. Ma, “Improved chaotic particle swarm optimization algorithm with more symmetric distribution for numerical function optimization,” Symmetry (Basel)., vol. 11, no. 7, Jul. 2019, doi: 10.3390/sym11070876.

[56] B. Y. Qu, P. N. Suganthan, and S. Das, “A distance-based locally informed particle swarm model for multimodal optimization,” IEEE Trans. Evol. Comput., vol. 17, no. 3, pp. 387–402, 2013, doi: 10.1109/TEVC.2012.2203138.

[57] K. Tang et al., “Benchmark Functions for the CEC’2008 Special Session and Competition on Large Scale Global Optimization,” 2007. [Online]. Available: http://nical.ustc.edu.cn/cec08ss.php.

[58] J. J. Liang, A. K. Qin, P. N. Suganthan, and S. Baskar, “Comprehensive learning particle swarm optimizer for global optimization of multimodal functions,” IEEE Trans. Evol. Comput., vol. 10, no. 3, pp. 281–295, Jun. 2006, doi: 10.1109/TEVC.2005.857610.

[59] M. H. Nadimi-Shahraki, S. Taghian, S. Mirjalili, and H. Faris, “MTDE: An effective multi-trial vector-based differential evolution algorithm and its applications for engineering design problems,” Appl. Soft Comput. J., vol. 97, Dec. 2020, doi: 10.1016/j.asoc.2020.106761.

[60] S. Mirjalili, S. M. Mirjalili, and A. Lewis, “Grey Wolf Optimizer,” Adv. Eng. Softw., vol. 69, pp. 46–61, 2014, doi: 10.1016/j.advengsoft.2013.12.007.