International Journal of Electrical and Computer Engineering (IJECE)
Vol. 10, No. 3, June 2020, pp. 3261~3274
ISSN: 2088-8708, DOI: 10.11591/ijece.v10i3.pp3261-3274

Population based optimization algorithms improvement using the predictive particles

M. M. H. Elroby[1], S. F. Mekhamer[2], H. E. A. Talaat[3], and M. A. Moustafa. Hassan[4]
[1]Electrical Engineering Department, Faculty of Engineering, Ain Shams University, Egypt
[2,3]Electrical Engineering Department, Future University, Egypt
[4]Electrical Engineering Department, Cairo University, Egypt

Article Info

Article history:

Received Jun 12, 2019
Revised Dec 2, 2019
Accepted Dec 11, 2019

Keywords:

Optimization
Particle Swarm Optimization
Population optimization
Predictive particle
Teaching Learning Based

ABSTRACT

A new efficient improvement, called Predictive Particle Modification (PPM), is proposed in this paper. This modification makes the particle look to the near area before moving toward the best solution of the group. This modification can be applied to any population algorithm. The basic philosophy of PPM is explained in detail. To evaluate the performance of PPM, it is applied to Particle Swarm Optimization (PSO) algorithm and Teaching Learning Based Optimization (TLBO) algorithm then tested using 23 standard benchmark functions. The effectiveness of these modifications are compared with the other unmodified population optimization algorithms based on the best solution, average solution, and convergence rate.

Corresponding Author:

M. M. H. Elroby,
Electrical Engineering Department,
Faculty of Engineering,
Ain Shams University, Egypt.
Email: mousaelroby@yahoo.com

1. INTRODUCTION

Recently, many Meta heuristic optimization algorithms have been developed. These include Particle Swarm Optimization (PSO) [1-5], Genetic Algorithm (GA) [6-9] , Deferential Evolution (DE) [10], Ant Colony (AC) [11], Gravitational Search algorithm (GSA) [12], Sine Cosine Algorithm (SCA) [13-15], Hybrid PSOGSA Algorithm [16], Adaptive SCA integrated with particle swarm [17], and Teaching Learning Based Optimization (TLBO) [18-20]. The same goal for them is to find the global optimum. In order to do this, a heuristic algorithm should be equipped with two main characteristics to ensure finding global optimum. These two major characteristics are exploration and exploitation. Exploration is the ability to search whole parts of the space whereas exploitation is the convergence ability to the best solution. The goal of all Meta heuristic optimization algorithms is to balance the ability of exploitation and exploration in order to find global optimum. According to [21], exploitation and exploration in evolutionary computing are not clear due to lake of a generally accepted perception. In other hand, with strengthening one ability, the other will weaken and vice versa. Because of the above-mentioned points, the existing Meta heuristic optimization algorithms are capable of solving finite set of problems. It has been proved that there is no algorithm, which can perform general enough to solve all optimization problems [22]. Many hydride optimization algorithms are to balance the overall exploration and exploitation ability.

In this study, the proposed modification increases the exploration and make the particle look to the surrounding space before affected by the best solution. The proposed modification can be applied to any population optimization algorithms. The PSO is one of the widely used population algorithms due to its simplicity, convergence speed, and ability of searching global optimum. Recently TLBO is a new efficient optimization method combine between teaching and learning phases. For the reasons listed above this

modification has been applied to PSO and TLBO. The organization of this paper is as follows: Section 2 describes the standard PSO and its exploration problem. Section 3 describes the standard TLBO. The proposed modification is presented in Section 4. Section 5 describes the results of the proposed modification. Section 6 concludes this research.

2. THE STANDARD PARTICLE SWARM OPTIMIZATION

2.1. Particle Swarm Optimization Algorithm

PSO is a population computation algorithm, which is proposed by Kennedy and Eberhart [1]. The PSO was inspired from social behavior of bird flocking. It uses a number of particles, which fly, around the search space. All particles try to find best solution. Meanwhile, they all look at the best particle in their paths. In other words, particles consider their own best solutions and the best solution has found so far. Each particle in PSO should consider the current position, the distance to pbest, the current velocity, and the distance to global best (gbest) to modify its position. PSO was modeled as follow [1]:

$$v_i^{t+1} = wv_i^t + c_1 \times rand \times (pbest_i^t - x_i^t) + c_2 \times rand \times (gbest^t - x_i^t) \quad (1)$$

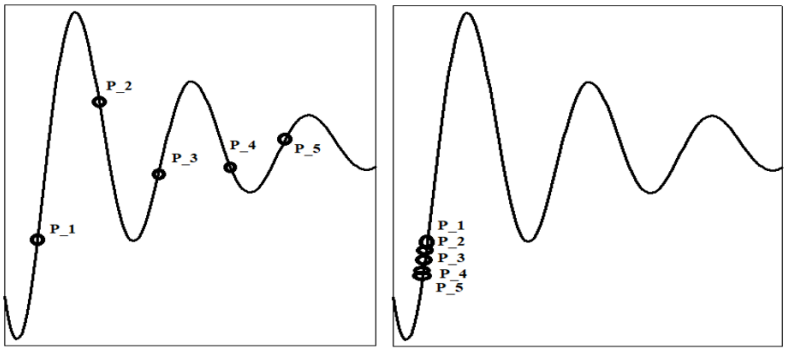
$$x_i^{t+1} = x_i^t + v_i^{t+1} \quad (2)$$

where v_i^{t+1} is the velocity of particle i at iteration t,
w is a weighting function,
c_j is a weighting factor,
rand is a random number between 0 and 1,
x_i^t is the current position of particle i at iteration t,
$pbest_i$ is the pbest of agent i at iteration t,
gbest is the best solution so far.

The first part of (1), wv_i^t provides exploration ability for PSO. The second and third parts, $c_{1\times}rand \times (pbest - x_i^t)$ and, $c_{1\times}rand \times (pbest - x_i^t)$ represent private thinking and collaboration of particles respectively [23, 24]. The PSO is initialized with randomly placing the particles in a problem space. In each iteration, the particles velocities are calculated using (1). After velocities calculating, the position of particle can be calculated as (2). This process will continue until meeting an end criterion.

2.1.1. PSO Exploration Problem

The first part of (1), wv_i^t provides PSO exploration ability. When the algorithm is started, the velocity is initialized with zero value. Thus from Equation 1, the Global Best Particle (GBP) (i.e. P1 in Figure 1 (a)) remains in its place until the best global solution is changed by a new particle. This means the global best particle cannot explore near area because it is not exited by any particle. In addition, particles that arrive from another places (P2 - P5) to the place of the global best solution with a certain velocity after a number of iteration may be damped before reaching the optimal solution as shown in Figure 1 (b). This phenomenon will be treated using PPM in Section 2.



(a) (b)

Figure 1.Particles at initial and final iteration, (a) initial iteration, (b) final iteration

3. THE STANDARD TEACHING LEARNING BASED OPTIMIZATION

The TLBO method is based on the effect of the teacher on the learners. The teacher is considered as a global best learned person $(gbest^t)$ who shares his knowledge with the learners. The process of TLBO is divided to two phase. The first phase consists of the 'Teacher Phase' and the second phase consists of the 'Learner Phase'. The 'Teacher Phase' means learning from the teacher and the 'Learner Phase' means learning through the interaction between learners. TLBO was modeled as follows [18]:

3.1. Teacher Phase

A learner learns from teacher by moving its mean to teacher value. Learner modification is expressed as:

$T_F = round[1 + rand(0,1)]$
$mean\ difference^{\ t} = r(gbest^t - T_F M^t)$
Where M^t is the mean of the learner and '$gbest^t$' is the global best (the teacher) at any iteration t.
For $i = 1 : number\ of\ learnears$
$X_i^{t+1} = X_i^t + mean\ difference^{\ t}$
$Accept\ X_i^{t+1}\ if\ it\ gives\ a\ better\ function\ value.$
End

3.2. Learner Phase

A learner learns new something if the other learner has better knowledge than him. Learner modification is expressed as:

For $i = 1 : number\ of\ learnears$
$Randomly\ select\ two\ learners\ X_i^t\ and\ X_j^t, where\ i \neq j$
If $f(X_i^t) < f(X_j^t)$
$X_i^{t+1} = X_i^t + rand(X_i^t - X_j^t)$
Else
$X_i^{t+1} = X_i^t + rand(X_j^t - X_i^t)$
End
$Accept\ X_i^{t+1} if\ it\ gives\ a\ better\ function\ value.$
End

4. PREDICTIVE PARTICLE

The main idea of the PPM based on that each iteration the particle should look at its near area and see if it have a value best than the GBP or not. If it have value better than GBP, it will be the GBP. The PPM can remedy non-exiting GBP (P1 in Figure 1 (a)) and not wait until excitation from another particle. In addition, it can improve the vision of the particle before movement toward GBP and overcome the jump over narrow area leaving goloabal solution.

Consider the initial values of the particles P1 to P5, which are shown in Figure 2. In the next iteration, these particles will move toward P1 (as it is the GBP at this moment) and take positions P1, P2′ to P5′. In addition, the P3 may jump to P3′ without converge to gbest especially when the fitness function have narrow area with high deep value. In addition, the P1 still in its position as it is GBP. These phenomena can be treated if the particle try to find a best solution (target) from near area before move to GBP as shown in Figure 3. This can be done using the numerical gradient with a definite target. Assume the fitness function (F) is a linear function near the particle position in matrix form:

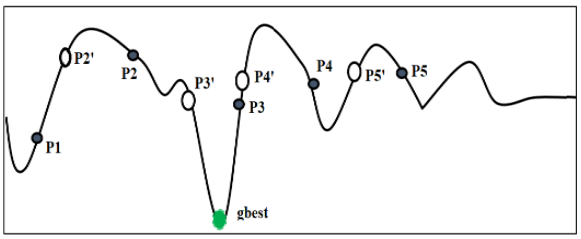


Figure 2. Particles movement

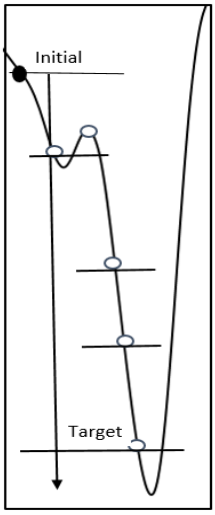


Figure 3. Initial and target of the particle

$$F = AX + b \tag{3}$$

Using numerical gradient method:

$$X_{new} = X_{old} - R * \frac{dF}{dX} \tag{4}$$

where

$$\frac{dF}{dX} = \begin{bmatrix} \Delta F/\Delta x_1 \\ \\ \Delta F/\Delta x_n \end{bmatrix} = A'$$

X_{new} is the new postion of the particle in column form
X_{old} is the current position of the particle
R is the step size
$\Delta F/\Delta x_i$ is calculated numerically near X_{old} by change only x_i

From (3):

$$F_{old} = AX_{old} + b \tag{5}$$

$$F_{new} = AX_{new} + b \tag{6}$$

From (5) and (6) by substraction:

$$X_{new} = -\frac{F_{old} - Fnew}{A} + X_{old} \tag{7}$$

where

F_{old} is the current fitinenss value
F_{new} is the new fitinenss value

From (4) and (7).

$$R = \frac{F_{old} - F_{new}}{A * \frac{dF}{dX}} = \frac{F_{old} - F_{new}}{(\frac{dF}{dX})' * \frac{dF}{dX}} \tag{8}$$

$$X_{new} = X_{old} - \frac{F_{old} - F_{new}}{(\frac{dF}{dX})' * \frac{dF}{dX}} * dF/dX \tag{9}$$

If F_i is the current fitness value of the particle and F_t is the target fitness of the particle (less than gbest value). It is nice to dived search steps to N steps as follows:

$$Assume\ \text{dist} = F_i - F_t \tag{10}$$

for each step

$$\Delta X = \frac{\text{dist/N}}{(\frac{dF}{dX})' * \frac{dF}{dX}} * \frac{dF}{dX} \tag{11}$$

$$X_{new} = X_{old} - \Delta X \tag{12}$$

The complete PPM algorithm before moving to GBP is shown in Table 1. In addition, the Modified PSO (MPSO) and Modified TLBO (MTLBO) are shown in Table 2 and Table 3 respectively.

Table 1. Gradient algorithm

Set particle gradient parameter:
$F_t <$ gbest
$X_{old} =$ current position of particle
$dist = F_i - F_t$
$intialize$ Vtemp $= 0$
Execute gradiant algorithm:
For N step
$X_{new} = X_{old} - \Delta X$ according to (12)
X_{new} =max(X_{new}, xmin);
X_{new} = min(X_{new}, xmax);
If F(X_{new}) < F(X_{new})
$Xtemp = X_{new}$
Vtemp=ΔX
Else
$Xtemp = X_t - 2 *$ Vtemp
$Xtemp$=max($Xtemp$, xmin);
$Xtemp$= min($Xtemp$, xmax);
Vtemp=Vtemp
End
End
Update particle position :

If $F(Xtemp) < gbest$
$x_i^{t+1} = Xtemp$
$v_i^{t+1} =$*Vtemp*
End

Table 2. Modified PSO

For *each particle*
initialize particle
End
Choose the particle with the best fitness value of all the particles as the gbest
Do
For *each particle*
Update particle position according to
$v_i^{t+1} = wv_i^t + c_2 \times rand \times (gbest - x_i^t)$
$x_i^{t+1} = x_i^t + v_i^{t+1}$
gradient algorithm as shown in Table 1
End
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
While maximum iterations or minimum error criteria is not attained

Table 3. Modified TLBO

For each particle
initialize particle
End
Choose the particle with the best fitness value of all the particles as the gbest
Do
1) Teacher phase
$T_F = round[1 + rand(0,1)]$
$mean\ difference^{\ t} = r(gbest^t - T_F M^t)$

For $i = 1 : number\ of\ learnears$
$X_i^{t+1} = X_i^t + mean\ difference^{\ t}$
gradient algorithm as shown in Table. 1 for i $Accept\ X_i^{t+1}\ if\ it\ gives\ a\ better\ function\ value.$
End
2) learner phase
For $i = 1 : number\ of\ learnears$
$Randomly\ select\ two\ learners\ X_i^t\ and\ X_j^t, where\ i \neq j$
gradient algorithm as shown in Table 1 for i and j
If $f(X_i^t) < f(X_j^t)$
$X_i^{t+1} = X_i^t + rand(X_i^t - X_j^t)$
Else
$X_i^{t+1} = X_i^t + rand(X_j^t - X_i^t)$
End
gradient algorithm as shown in Table 1 for i
$Accept\ X_i^{t+1} if\ it\ gives\ a\ better\ function\ value.$
End
Choose the particle with the best fitness value of all the particles as the gbest
While maximum iterations or minimum error criteria is not attained

5. EXPERIMENTAL RESULTS AND DISCUSSION

The standard PSO, PSOSGSA, SCA, TLBO, MPSO, and MTLBO with the parameter in Table 4 [25-28] have executed 30 independent runs over each benchmark function for statistical analysis. As shown in Table 5, MPSO and MTLBO outperformed all of the other algorithms with regard to the quality of the solutions for all functions. In contrast, the other algorithms produced poor results on certain functions and accurate results on others. This finding reflects the efficient performance of the MPSO and MTLBO in comparison with the other unmodifeied algorithms. In addition, Figure 4 to Figure 11 show a comparison between MPSO and MTLBO and all the other algorithms for the convergence rate for the fitness versus the iterations. These figures show that MPSO and MTLBO outperforms all the other unmodifeied algorithms in terms of the convergence speed with an accurate solutio

Table 4. Algorithms parameter

Algorithm	Parameter
PSO	C1=C2=2 wdamp=0.9
PSOGSA	G0=1, C1=0.5, C2=1.5
SCA	a = 2, r2=(2*pi)*rand , r3=2*rand, r4=rand
TLBO	TF=randi([1 2])
MPSOA	C1=C2=2, wdamp=0.9 , N=5
MTLBO	G0=1, C1=0.5, C2=1.5, N=5
MaxVelocity=0.2*(VarMax-VarMin) , MinVelocity= —MaxVelocity	

Table 5. Benchmark functions

Function	n	Range	PSO	PSOGSA	SCA	TLBO	MPSO	MTLBO
$F_1 = \sum_{i=1}^{n} x_i^2$	30	[-100,100]	12.6934	2.6127e-04	70.1582	9.4739e-10	3.7604e-20	1.0693e-310
$F_2 = \sum_{i=1}^{n} \lvert x_i \rvert + \prod_{i=1}^{n} x_i$	30	[-10,10]	0.86537	10.0018	1.86028	1.4461e-05	0.0013288	3.1831e-25
$F_3 = \sum_{i=1}^{n} (\sum_{j=1}^{i} x_j)^2$	30	[-100,100]	138.401	0.073822	44.1612	0.0043885	0.031916	8.5545e-33
$F_4 = \max_i \lvert x_i \rvert , 1 < i < n$	30	[-100,100]	4.4334	0.033062	1.7684	0.00014051	4.6443e-21	3.2386e-14
$F_5 = \sum_{i=1}^{n} [100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2]$	30	[-30,30]	761.6989	7.5496	427.0597	6.7747	5.5657	0.035228
$F_6 = \sum_{i=1}^{n} (x_i + 0.5)^2$	30	[-100,100]	23.7811	0.0001977	20.3904	5.9683e-06	5.4363e-17	6.7631e-27
$F_7 = \sum_{i=1}^{n} i x_i^4 + ran[0,1)$	30	[-1.28,1.28]	0.016338	0.053117	0.029228	0.003001	0.0068485	0.00059692
$F_8 = \sum_{i=1}^{n} -x_i \sin(\sqrt{\lvert x_i \rvert})$	30	[-500,500]	-2759.9392	-2818.879	-1900.6632	-2809.0376	-3004.0262	-3367.6955
$F_9 = \sum_{i=1}^{n} [x_i^2 - 10\cos(2\pi x_i) + 10]$	30	[-5.12,5.12]	10.3042	9.9651	36.8504	22.5791	9.9635	1.3882
$F_{10} = 20\exp\left(-0.2\sqrt{\frac{\sum_{i=1}^{n} x_i^2}{n}}\right) - \exp\left(\frac{1}{n}\sum_{i=1}^{n} \cos(2\pi x_i)\right) + 20 + e$	30	[-32,32]	3.6428	0.024135	7.2123	1.4591e-05	0.00033049	9.1038e-13

Table 5. Benchmark functions (*continue*)

Function	n	Range	PSO	PSOGSA	SCA	TLBO	MPSO	MTLBO
$F_{11} = \frac{1}{4000}\sum_{i=1}^{n} x_i^2 - \prod_i^n cos\left(\frac{x_i}{\sqrt{i}}\right) + 1$	30	[-600,600]	1.3209	0.11893	1.2887	0.21989	0.4499	0
$F_{12} = \frac{\pi}{n}\{10\sin(\pi x_i) + \sum_{i=1}^{n}(x_i - 1)^2[1 + 10\sin^2(\pi x_{i+1})] + (x_n - 1)^2\} + \sum_{i=1}^{n} u(x_i, 10,100,4)$	30	[-50,50]	0.54574	1.4131	1.128	8.4362e-06	0.0031873	1.8935e-13
$F_{13} = 0.1\{\sin^2(3\pi x_1) + \sum_{i=1}^{n}(x_i - 1)^2[1 + \sin^2(3\pi x_i + 1)] + (x_n - 1)^2[1 + \sin^2(2\pi x_n)]\} + \sum_{i=1}^{n} u(x_i, 5,100,4)$	30	[-50,50]	1.7699	2.8238e-05	1.4988	0.0026299	0.052191	5.1608e-13
$F_{14} = \left(\frac{1}{100} + \sum_{j=1}^{25}\frac{1}{j+\sum_{i=1}^{2}(x_i - a_{ij})^2}\right)^{-1}$	2	[-65.536, 65.536]	1.5716	1.2012	1.0012	0.73009	0.73009	0.73009
$F_{15} = \sum_{i=1}^{11}[a_i - \frac{x_1(b_i^2 + b_i x_2)}{b_i^2 + b_i x_3 + x_4}]^2$	4	[-5, 5]	0.00078406	0.00073004	0.00319543	0.00078124	0.00039186	0.00031247
$F_{16} = 4x_1^2 - 2.1x_1^4 + \frac{1}{3}x_1^6 + x_1x_2 - 4x_2^2 + 4x_2^4$	2	[-5, 5]	-1.0316	-1.0316	-1.0316	-1.0316	-1.0316	-1.0316
$F_{17} = (x_2 - \frac{5.1}{4\pi^2}x_1^2 + \frac{5}{\pi}x_1 - 6)^2 + 10\left(1 - \frac{1}{8\pi}\right)cosx_1 + 10$	2	[-5, 5]	0.39789	0.39789	0.42789	0.39789	0.39789	0.39789
$F_{18} = [1 + (x_1 + x_2 + 1)^2(19 - 14x_1 + 3x_1^2 - 14x_2 + 6x_1x_2 + 3x_2^2)] \times [30 + (2x_1 - 3x_2)^2 \times (18 - 32x_1 + 12x_1^2 + 48x_2 - 36x_1x_2 + 27x_2^2)]$	2	[-2, 2]	3.0001	3	3	3	3	3
$F_{19} = -\sum_{i=1}^{4} c_i \exp(-\sum_{j=1}^{3} a_{ij}(x_i - p_{ij})^2)$	3	[1, 3]	-3.8628	-3.8628	-3.8516	-3.8628	-3.8628	-3.8628
$F_{20} = -\sum_{i=1}^{4} c_i \exp(-\sum_{j=1}^{6} a_{ij}(x_i - p_{ij})^2)$	6	[0, 1]	-3.3215	-3.2031	-2.9779	-3.3175	-3.322	-3.3217
$F_{21} = -\sum_{i=1}^{5}[(X - a_i)(X - a_i)^T + c_i]^{-1}$	4	[0, 10]	-10.1532	-2.6305	-4.559	-10.149	-10.1532	-9.4881
$F_{22} = -\sum_{i=1}^{7}[(X - a_i)(X - a_i)^T + c_i]^{-1}$	4	[0, 10]	-5.1288	-10.4029	-1.5161	-4.9127	-10.4007	-10.2496
$F_{23} = -\sum_{i=1}^{10}[(X - a_i)(X - a_i)^T + c_i]^{-1}$	4	[0, 10]	-5.1756	-10.5364	-1.7766	-10.5364	-10.5363	-10.4808

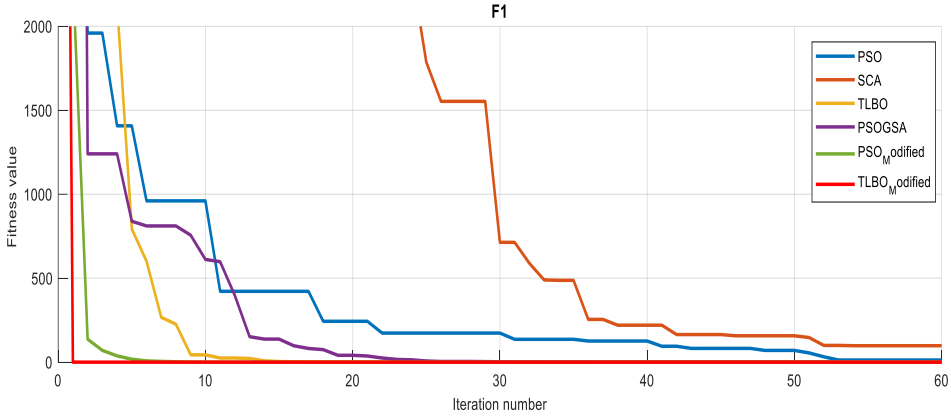


Figure 4. Converge rate curves for F1 to F3

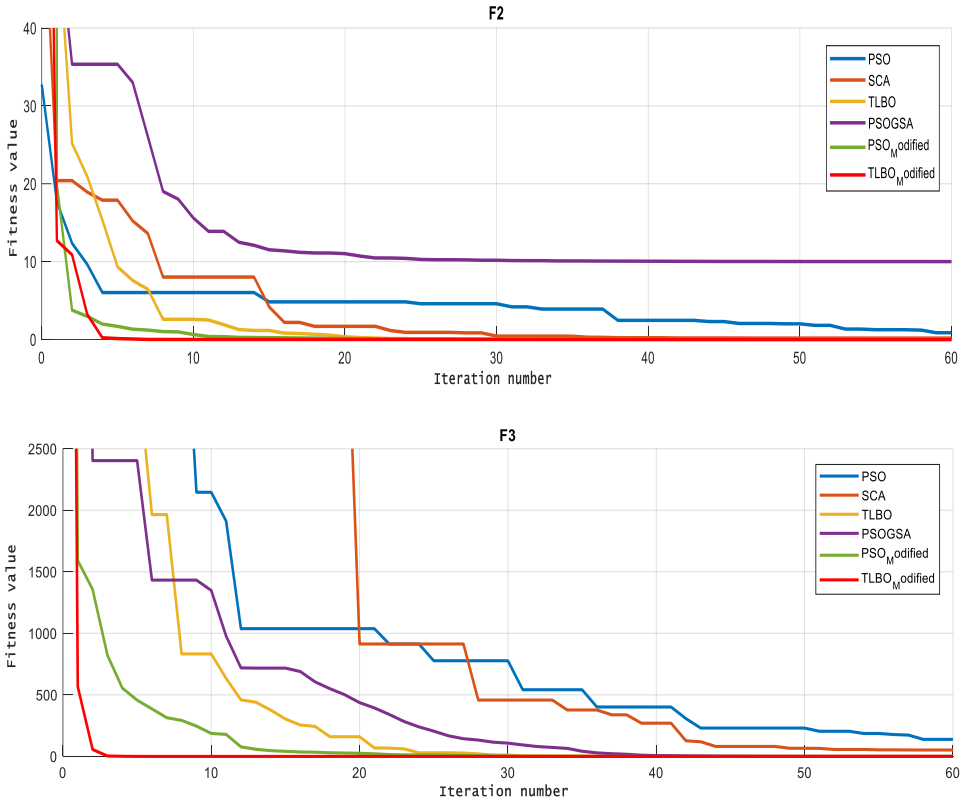


Figure 4. Converge rate curves for F1 to F3 (*continue*)

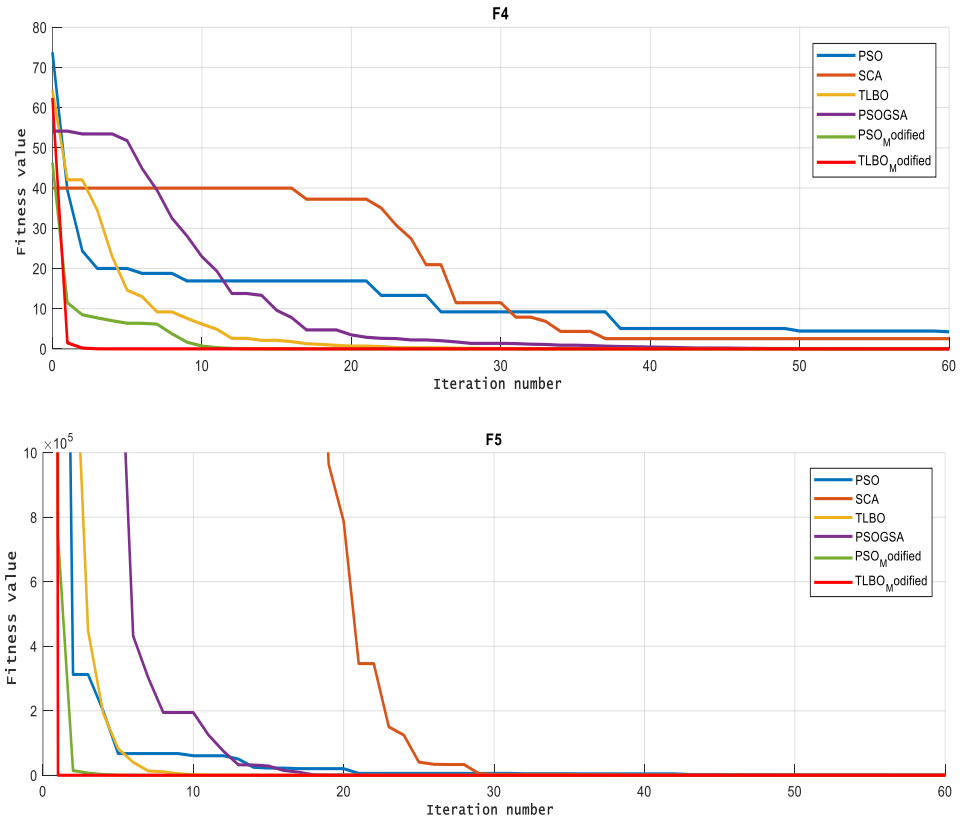


Figure 5. Converge rate curves for F4 to F6

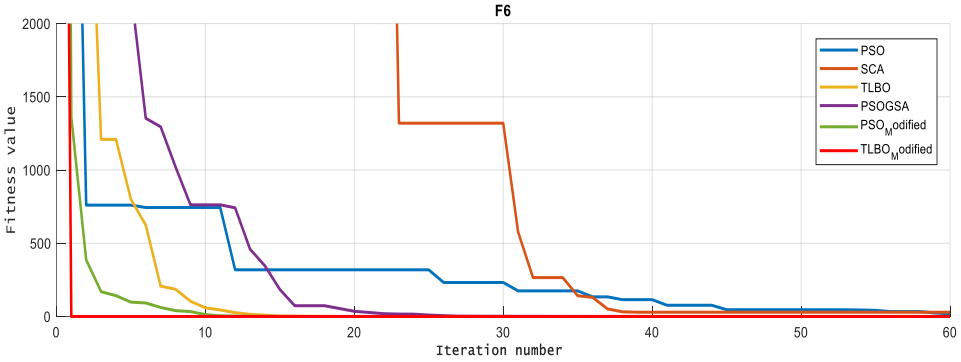


Figure 5. Converge rate curves for F4 to F6 (*continue*)

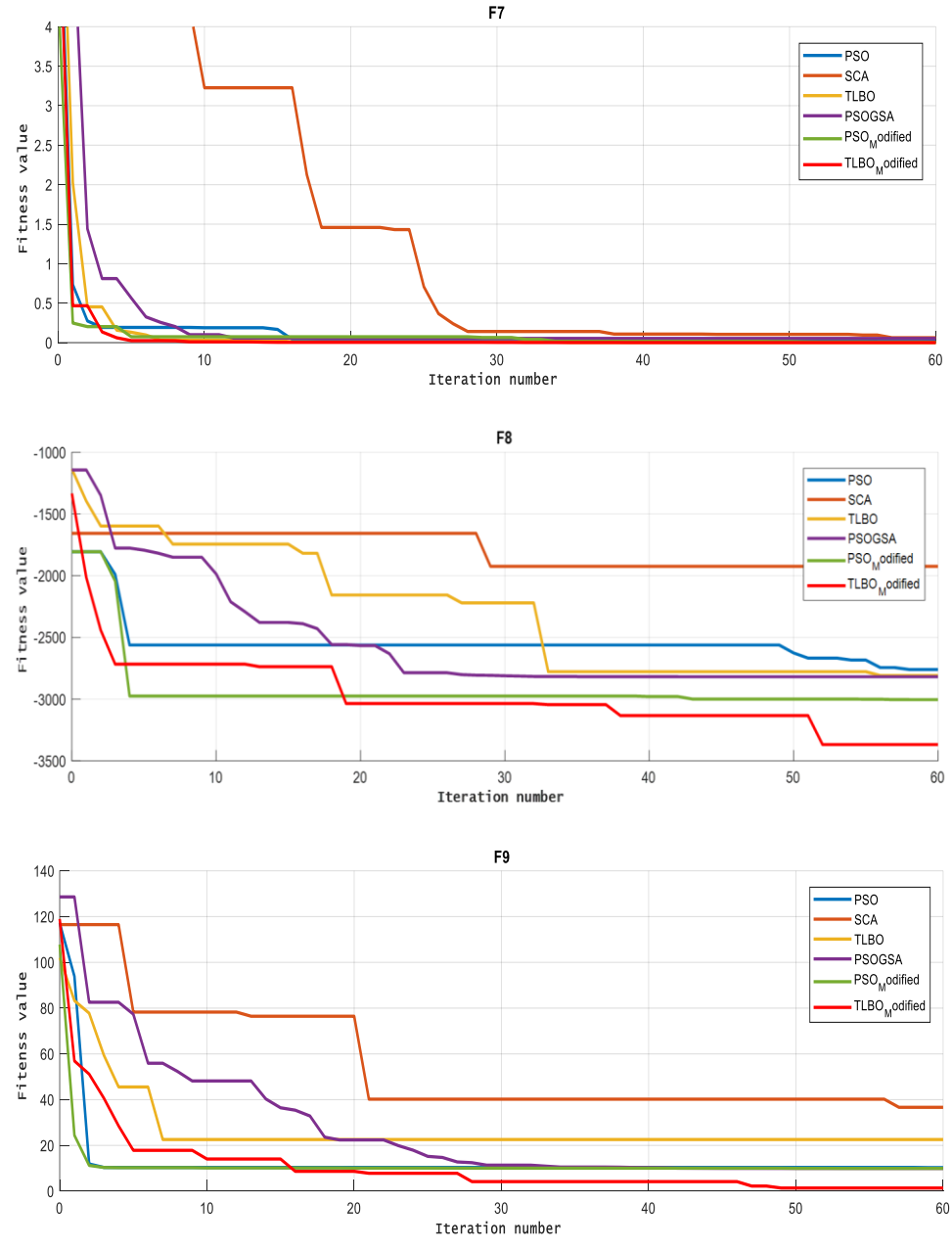


Figure 6. Converge rate curves for F7 to F9

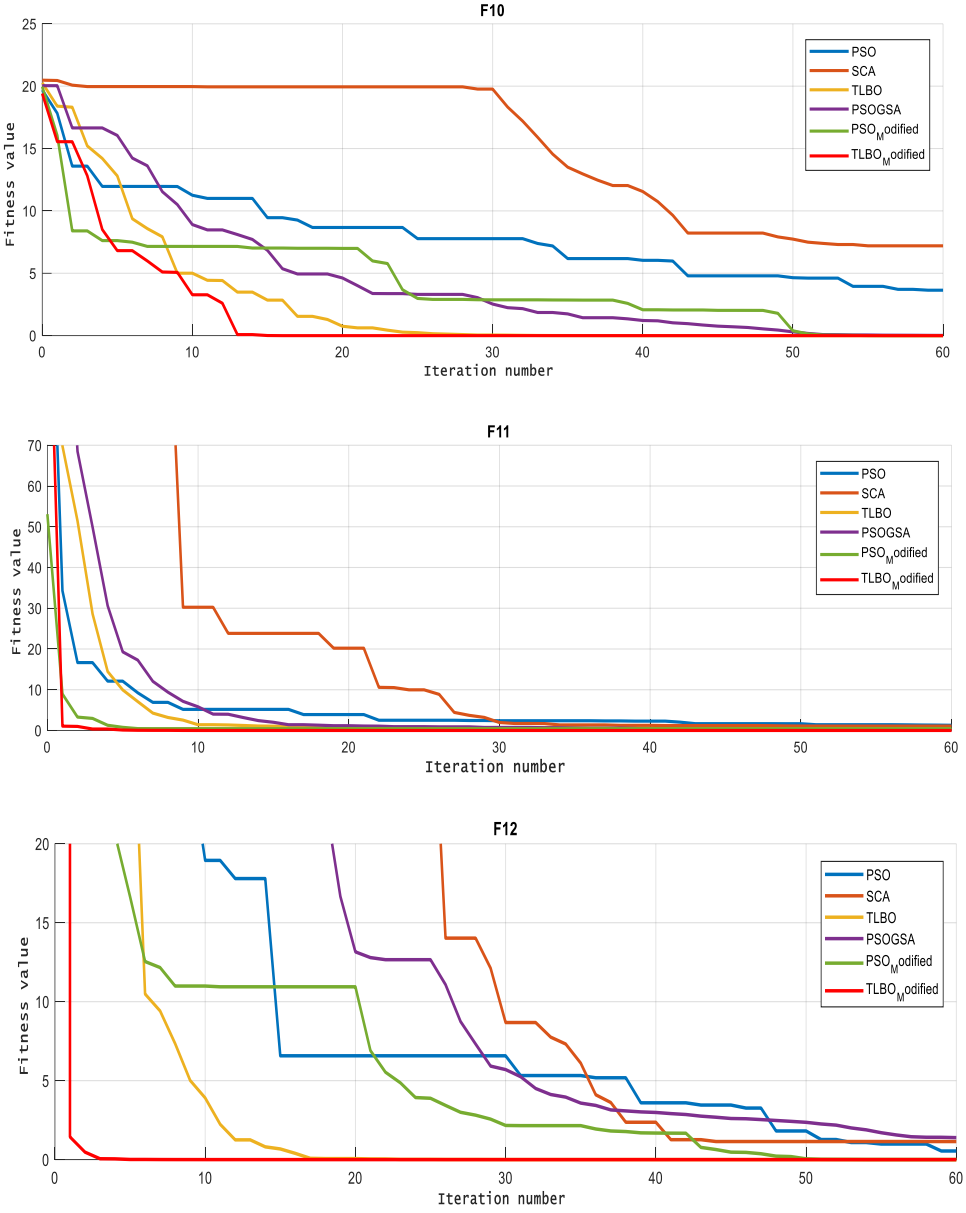


Figure 7. Converge rate curves for F10 to F12

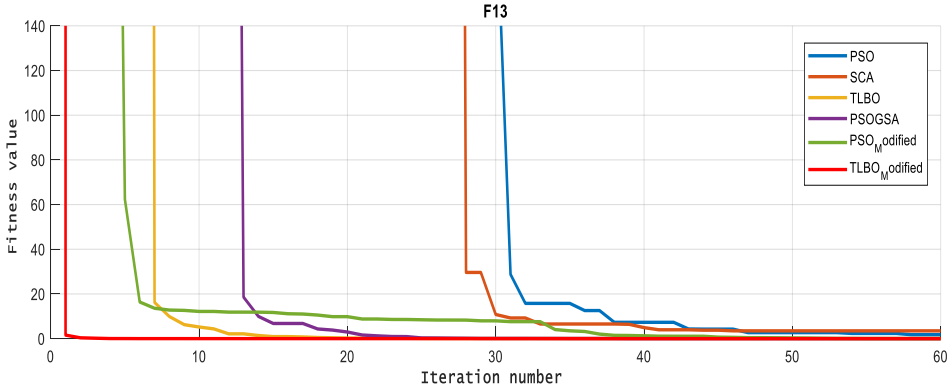


Figure 8. Converge rate curves for F13 to F15

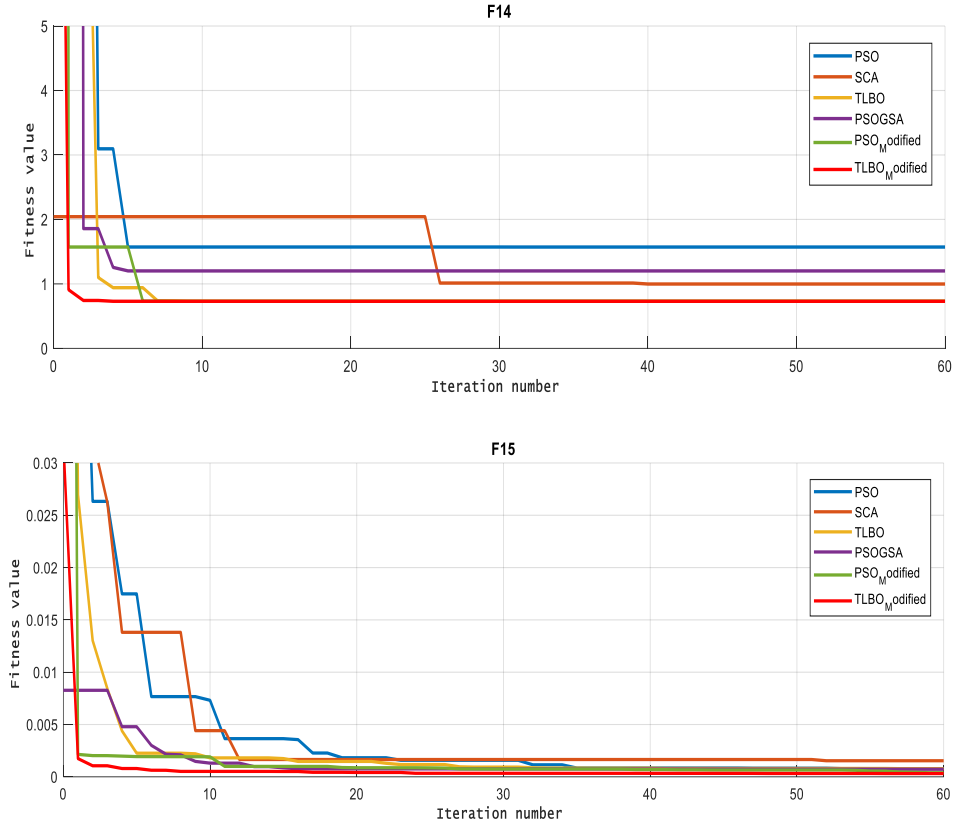


Figure 8. Converge rate curves for F13 to F15 (*continue*)

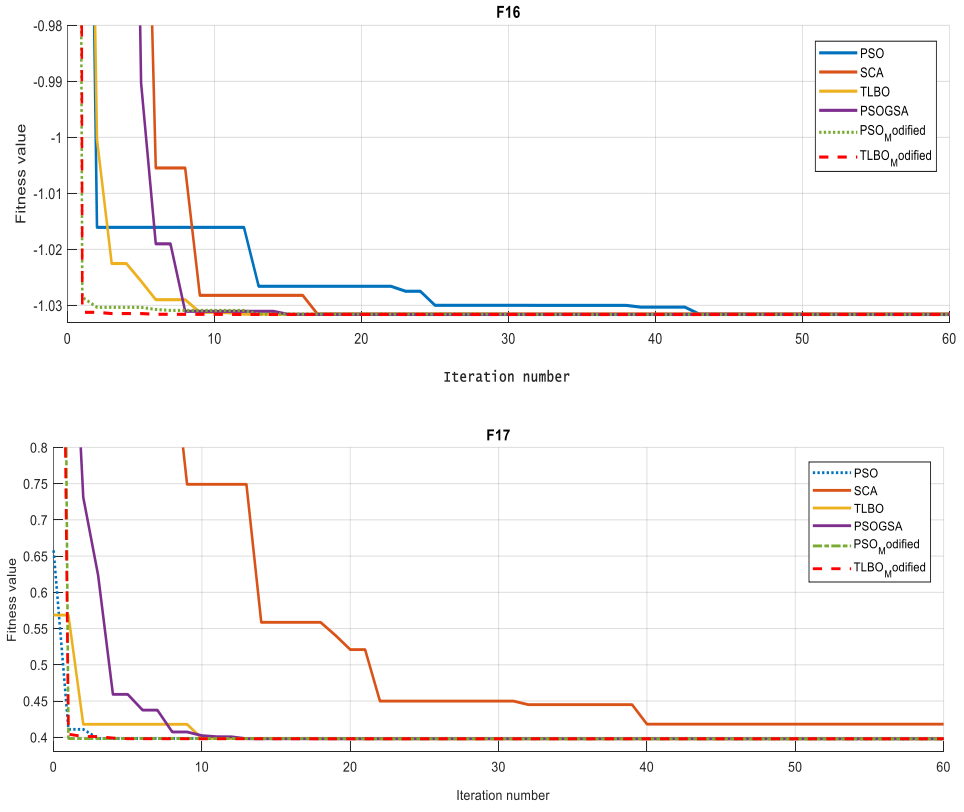


Figure 9. Converge rate curves for F16 to F18

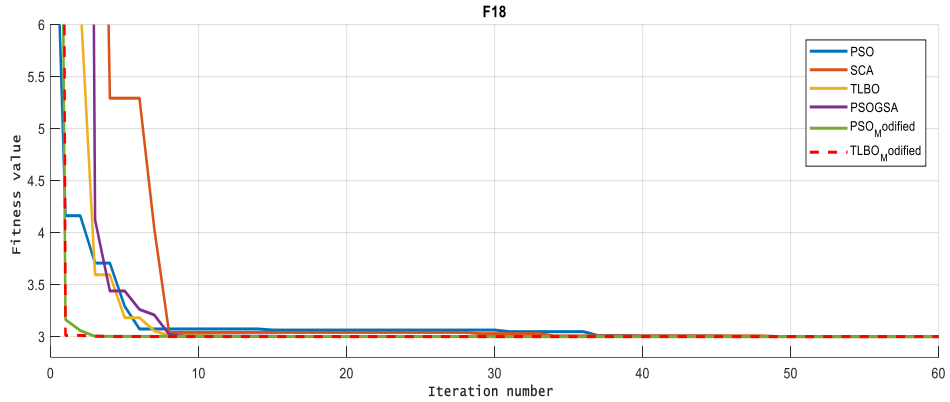


Figure 9. Converge rate curves for F16 to F18 (*continue*)

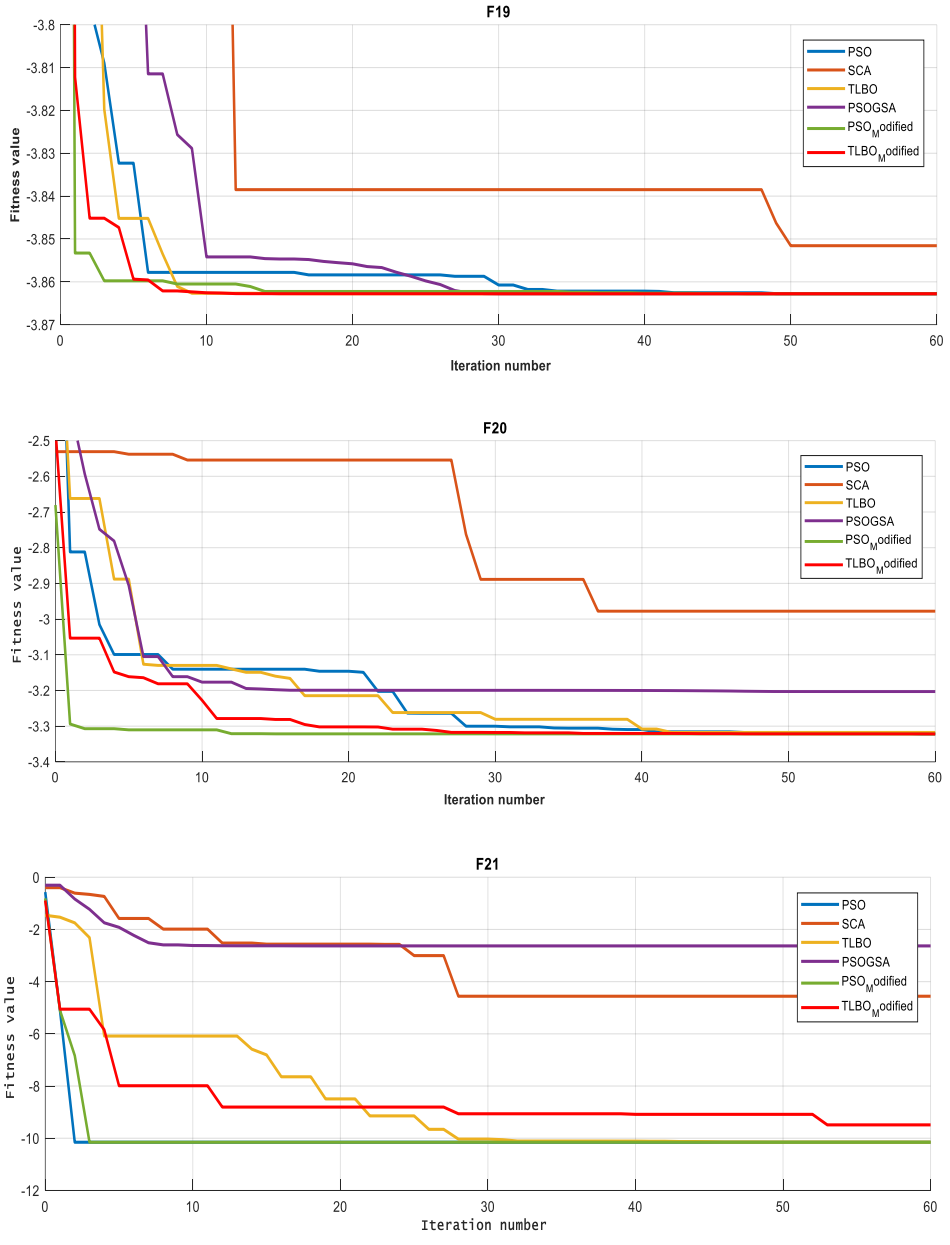


Figure 10. Converge rate curves for F19 to F21

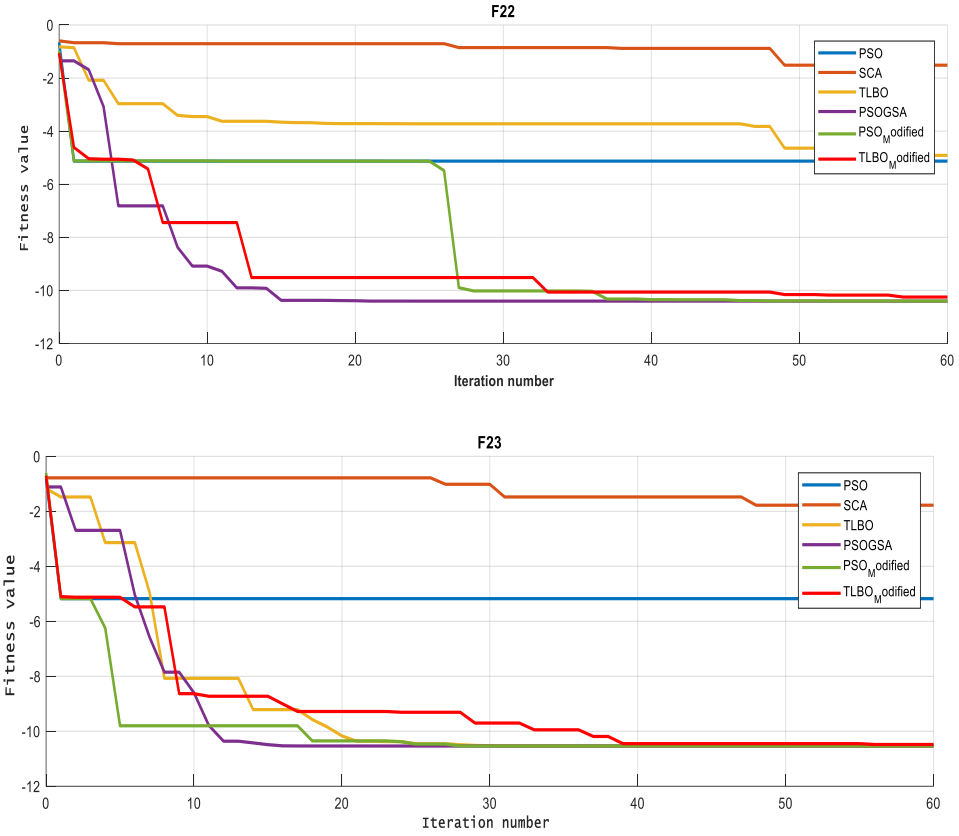


Figure 11. Converge rate curves for F22 to F23

6. CONCLUSION

In this paper, the PPM has the advantage of powerful exploration. Thus, it was necessary to enhance the population algorithms by merging it with PPM, which has the advantage of powerful exploitation. Hence, the proposed modification improves the exploration quality and maintaining fast convergence. PPM optimization was tested to find the optimal solution for standard mathematical functions, and results demonstrated improvement in solution quality and convergence rate..

REFERENCES

[1] J. Kennedy; R. Eberhart, "Particle Swarm Optimization," IEEE Int. Conf. Neural Networks, vol 4, pp. 1942–1948, 1995.
[2] I. C. Trelea, "The particle swarm optimization algorithm: Convergence analysis and parameter selection," *Inf. Process. Lett.*, vol. 85, no. 6, pp. 317-325, 2003.
[3] I. Science, "Analysis of Particle Swarm Optimization Algorithm," *Comput. Inf. Sci.*, vol. 3, pp. 180–184, 1998.
[4] R. Poli, J. Kennedy, and T. Blackwell, "Particle swarm optimization: An overview," *Swarm Intell.*, vol. 1, no. 1, pp. 33–57, 2007.
[5] F. Van Den Bergh and A. P. Engelbrecht, "A study of particle swarm optimization particle trajectories," *Inf. Sci. (Ny).*, vol. 176, no. 8, pp. 937–971, 2006.
[6] D. E. Goldberg, "Genetic Algorithms in Search, Optimization, and Machine Learning," New York Addison-Wesley, 1989.
[7] K. Deb and S. Agrawal, "Understanding interactions among genetic algorithm parameters," Found. Genet. Algorithms V, San Mateo, CA Morgan Kauffman, pp. 265–286, 1999.
[8] P. Pongcharoen, C. Hicks, P. M. Braiden, and D. J. Stewardson, "Determining optimum Genetic Algorithm parameters for scheduling the manufacturing and assembly of complex products," *Int. J. Prod. Econ.*, vol. 78, no. 3, pp. 311–322, 2002.
[9] A. H. Wright, "Genetic Algorithms for Real Parameter Optimization," Foundations of Genetic Algorithms, vol 1, pp. 205–218, 1991.
[10] R. Storn dan K. Price, "Differential Evolution – A Simple and Efficient Heuristic for Global Optimization Over Continuous Spaces," *Journal of Global Optimization*, vol. 11, no. 4, pp 341–359, 1997.
[11] D. Marco, "Ant colony optimization," *Scholarpedia*, vol. 2, no. 3, pp. 1461, 2007.

[12] E. Rashedi, H. Nezamabadi-Pour, and S. Saryazdi, "GSA: A Gravitational Search Algorithm," Information Sciences, vol. 179, no. 13, pp. 2232-2248, 2009.
[13] S. Mirjalili, "SCA: A Sine Cosine Algorithm for Solving Optimization Problems," *Knowledge-Based Syst*., vol. 96, pp. 120–133, 2016.
[14] A. I. Hafez, H. M. Zawbaa, E. Emary, and A. E. Hassanien, "Sine cosine optimization algorithm for feature selection," *Proc. 2016 Int. Symp. Innov. Intell. Syst. Appl. INISTA 2016*, 2016.
[15] M. Abd Elaziz, D. Oliva, and S. Xiong, "An improved Opposition-Based Sine Cosine Algorithm for global optimization," *Expert Syst. Appl.*, vol. 90, pp. 484–500, 2017.
[16] S. Mirjalili and S. Z. M. Hashim, "A New Hybrid PSOGSA Algorithm for Function Optimization," Proc. ICCIA 2010, Int. Conf. Comput. Inf. Appl, no. 1, pp. 374–377, 2010.
[17] M. Issa, A. E. Hassanien, D. Oliva, A. Helmi, I. Ziedan, and A. Alzohairy, "ASCA-PSO: Adaptive Sine Cosine Optimization Algorithm Integrated with Particle Swarm for Pairwise Local Sequence Alignment," *Expert Syst. Appl*, vol. 99, pp. 56–70, 2018.
[18] R. V. Rao, V. J. Savsani, and D.P Vakharia, "Teaching–Learning-Based Optimization: A Novel Method for Constrained Mechanical Design Optimization Problems," *Comput. Des.*, vol. 43, no. 3, pp. 303-315, 2011.
[19] R. V. Rao and V. Patel, "An improved teaching-learning-based optimization algorithm for solving unconstrained optimization problems," *Sci. Iran.*, vol. 20, no. 3, pp. 710–720, 2013.
[20] K. Yu, X. Wang, and Z. Wang, "An improved teaching-learning-based optimization algorithm for numerical and engineering optimization problems," *J. Intell. Manuf.*, vol. 27, no. 4, pp. 831–843, 2016.
[21] A. E. Eiben and C. A. Schippers, "On Evolutionary Exploration and Exploitation," Journal Fundamenta Informaticae, vol. 35, no. 1-4, pp. 35–50, 1998.
[22] N. Benfenatki, "La Tuberculose multirésistante," *Rev. Med. Interne*, vol. 30, pp. 268-272, 2009.
[23] G. A. F. Alfarisy, W. F. Mahmudy, and M. H. Natsir, "Good parameters for PSO in optimizing laying hen diet," *International Journal of Electrical and Computer Engineering (IJECE)*, vol. 8, no. 4, pp. 2419–2432, 2018.
[24] W. R. Abdul-Adheem, "An enhanced Particle Swarm Optimization algorithm," *International Journal of Electrical and Computer Engineering (IJECE)*, vol. 9, no. 6, pp. 4904–4907, 2019.
[25] D. B. Chen and C. X. Zhao, "Particle swarm optimization with adaptive population size and its application," *Appl. Soft Comput. J.*, vol. 9, no. 1, pp. 39–48, 2009.
[26] G. S. Basheer, M. S. Ahmad, and A. Y. C. Tang, "Intelligent Information and Database Systems - Part II," *7th Asian Conf.*, vol. 7803, pp. 549–558, 2013.
[27] J. Gardezi, "Handbook of Research on Machine Learning Innovations and Trends," no. April. 2017.
[28] A. Kaveh and T. Bakhshpoori, Metaheuristics: Outlines, MATLAB Codes and Examples, 2019.