Hindawi Publishing Corporation
Computational Intelligence and Neuroscience
Volume 2016, Article ID 4561507, 10 pages
http://dx.doi.org/10.1155/2016/4561507

Research Article

An Improved Teaching-Learning-Based Optimization with the Social Character of PSO for Global Optimization

Feng Zou, Debao Chen, and Jiangtao Wang

School of Physics and Electronic Information, Huaibei Normal University, Huaibei 235000, China

Correspondence should be addressed to Debao Chen; chendb_8@163.com

Received 24 June 2015; Accepted 18 November 2015

Academic Editor: Silvia Conforto

An improved teaching-learning-based optimization with combining of the social character of PSO (TLBO-PSO), which is considering the teacher's behavior influence on the students and the mean grade of the class, is proposed in the paper to find the global solutions of function optimization problems. In this method, the teacher phase of TLBO is modified; the new position of the individual is determined by the old position, the mean position, and the best position of current generation. The method overcomes disadvantage that the evolution of the original TLBO might stop when the mean position of students equals the position of the teacher. To decrease the computation cost of the algorithm, the process of removing the duplicate individual in original TLBO is not adopted in the improved algorithm. Moreover, the probability of local convergence of the improved method is decreased by the mutation operator. The effectiveness of the proposed method is tested on some benchmark functions, and the results are competitive with respect to some other methods.

1. Introduction

Global optimization is a concerned research area in science and engineering. Many real-world optimization applications can be formulated as global optimization problems. To efficiently solve the global optimization problems, the efficient and robust optimization algorithms are needed. The traditional methods often fail to solve complex global optimization problems. A detailed overview of the research progress in deterministic global optimization can be found in [1]. To overcome the difficulties of traditional methods, some well-known metaheuristics are developed for solving global optimization problems during the last four decades. Among existing metaheuristics, particle swarm optimization (PSO) algorithm [2] plays very important role in solving global optimization problems. PSO inspired by the social behaviors of birds has been successfully utilized to optimize continuous nonlinear functions [3], but the standard PSO algorithm (SPSO) might trap into local optima when solving complex multimodal problems. To improve the global performance of PSO, some variants are developed. Linear decreasing inertia weight particle swarm optimization (LDWPSO) [4] was introduced by Shi and Eberhart to overcome the lack of velocity control in standard PSO; the inertia weight of the algorithm decreases from a large value to a small one with increasing of evolution. To improve the convergence accuracy of PSO, some modified operators are adopted to help the swarm escape from the local optima especially when the best fitness of the swarm is not changed in some continuous iteration [5–8]. To improve the convergence speed of PSO, some modifications for updating rule of particles were presented in [9], and the improved algorithm was tested on some benchmark functions. Adaptive particle swarm optimization algorithm was introduced by Zhan to improve the performance of PSO, many operators were proposed to help the swarm jump out of the local optima and the algorithms have been evaluated on 12 benchmark functions [10, 11]. Some detail surveys of development to PSO are introduced for the interested readers in [12–14].

Though swarm intelligence optimization algorithms have been successfully used to solve global optimization problems, the main limitation of the previously mentioned algorithms is that many parameters (often more than two parameters) should be determined in updating process of individuals, and the efficiency of algorithms are usually affected by these parameters. For example, there are three parameters (w, c_1,

and c_2) that should be determined in updating equations of PSO. Moreover, the optimal parameters of the algorithms are often difficult to be determined. To decrease the effects of parameters for the algorithms, teaching-learning-based (TLBO) algorithm [15] is proposed recently, and it has been used in some real applications [16–19]. Some variants of the TLBO algorithm have been presented for improving the performance of the original TLBO [20, 21].

Under the framework of population-based optimizations, many variations of evolutionary optimization algorithms have been designed. Each of these algorithms performs well in certain cases and none of them are dominating one another. The key reason for employing the hybridization is that the hybrid algorithm can take advantage of the strengths of each individual technique while simultaneously overcoming its main limitations. On top of this idea, many hybrid algorithms have been presented [22–27]. In the paper, to improve the performance of TLBO algorithm for solving global optimization problems, an improved TLBO algorithm with combining of the social character of PSO, named TLBO-PSO, is proposed. In the improved TLBO algorithm, the teacher improves not only the performance of the mean grade of the whole class but also the performance of every student. The proposed algorithm has been evaluated on some benchmark functions, and the results are compared with some other algorithms.

The paper is organized as follows. Section 2 provides a brief description of the standard PSO algorithm. Original teaching-learning-based (TLBO) algorithm is introduced in Section 3. Section 4 provides the detail procedure of the improved teaching-learning-based optimization algorithm with combining of the social character of PSO (TLBO-PSO). Some experiments are given in Section 5. Section 6 concludes the paper.

2. Particle Swarm Optimizers

In the standard PSO (SPSO) algorithm, a swarm of particles are represented as potential solutions; each particle searches for an optimal solution to the objective function in the search space. The ith particle is associated with two vectors, that is, the velocity vector $V_i = [v_i^1, v_i^2, \ldots, v_i^D]$ and the position vector $X_i = [x_i^1, x_i^2, \ldots, x_i^D]$, where D is the dimensions of the variable. Each particle dynamically updates its position in a limited searching domain according to the best position of current iteration and the best position which it has achieved so far. The velocity and position of the ith particle are updated as follows:

$$v_i^d = w v_i^d + c_1 r_1 \left(Pbest_i^d - x_i^d\right) + c_2 r_2 \left(Gbest^d - x_i^d\right), \quad (1)$$

$$x_i^d = x_i^d + v_i^d, \quad (2)$$

where c_1 and c_2 are the acceleration coefficients, the value of them often equals 2, r_1 and r_2 are random numbers between 0 and 1. w is the inertia weight which influences the convergent character of the swarm. Large inertial weight benefits global searching performance, while small one facilitates local searching. x_i^d is the position of the ith particle on the dth dimension, $Pbest_i^d$ is the best position which the ith has achieved so far, and $Gbest^d$ is the best position of the swarm in current iteration. In general, the velocity of all the particles is limited by the maximum velocity ($V_{\max}$), and positions of them are limited by the maximum position ($P_{\max}$) and the minimum position ($P_{\min}$). To improve the performance of PSO, the method LDWPSO in which the inertia weight decreases linearly from a relatively large value to a small one is proposed [4], and the changed weight with evolution iteration is shown as follows:

$$w = w_{\max} - \text{gen} * \frac{w_{\max} - w_{\min}}{\max \text{gen}}, \quad (3)$$

where $w_{\max} = 0.9$, $w_{\min} = 0.4$ are the maximum and minimum values of inertia weight, respectively. gen is the current generation, max gen is the maximum evolutionary iteration. The adaptive weights make swarm have good global searching ability at the beginning of iterations and good local searching ability near the end of runs. Algorithm 1 presents the detail steps of LDWPSO algorithm.

3. Teaching-Learning-Based Optimization (TLBO) Algorithm

TLBO is one of the recently proposed population-based algorithms and it simulates the teaching-learning process of the class [15]. The main idea of the algorithm is based on the influence of a teacher on the output of learners in a class and the interaction between the learners. The TLBO algorithm does not require any specific parameters. TLBO requires only common controlling parameters like population size and number of generations for its working. The use of teaching in the algorithm is to improve the average grades of the class. The algorithm contains two phases: teacher phase and learner phase. The detail description of the algorithm can be found in [15]. In this paper, only the two main phases of TLBO are introduced as follows.

3.1. Teacher Phase. In the teacher phase of TLBO algorithm, the task of the teacher is to increase the mean grades of the class. Suppose that an objective function is $f(X)$ with n-dimensional variables, the ith student can be represented as $X_i = [x_{i1}, x_{i2}, \ldots, x_{in}]$. At any iteration g, assume that the population size of the class is m, the mean result of students in current iteration is $X_{g\text{mean}} = (1/m)[\sum_{i=1}^m x_{i1}, \sum_{i=1}^m x_{i2}, \ldots, \sum_{i=1}^m x_{in}]$. The student with the best fitness is chosen as the teacher of current iteration; it is represented as X_{teacher}. All the students will update their position as follows:

$$X_{i,\text{new}} = X_{i,\text{old}} + r_1 \left(X_{\text{teacher}} - T_F X_{g\text{mean}}\right), \quad (4)$$

where $X_{i,\text{new}}$ and $X_{i,\text{old}}$ are the new and the old position of the ith student and r_1 is the random number in the range $[0, 1]$. If the new solution is better than the old one, the old position of individual will be replaced by the new position. The value of T_F is randomly decided by the algorithm according to

$$T_F = \text{round}\left[1 + \text{rand}\,(0, 1)\,\{2 - 1\}\right]. \quad (5)$$

Initialize:
Initialize maxgen, $w_{\max}$, $w_{\min}$, $V_{\max}$, $P_{\max}$, $P_{\min}$, Population size (Ps), Dimension size (D) and the initial swarm
Optimize:
for gen = 1 : maxgen
Calculate the the inertia weight w according to (3)
for i = 1 : Ps
Calculate fitness of all particles;
Calculate the best position $Gbest^d$ of current generation;
Calculate the best position $Pbest_i^d$ of ith particle which it has achieved so far;
for d = 1 : D
Update the velocity of ith particle according to (1)
If $v_i^d > V_{\max}$ then $v_i^d = V_{\max}$
If $v_i^d < -V_{\max}$ then $v_i^d = -V_{\max}$
Update the position of ith particle according to (2)
If $x_i^d > P_{\max}$ then $x_i^d = P_{\max}$
If $x_i^d < P_{\min}$ then $x_i^d = P_{\min}$
end
end
end

Algorithm 1: LDWPSO algorithm.

3.2. Learner Phase. In learner phase of TLBO algorithm, learners can increase their knowledge from others. A learner interacts randomly with other learners for enhancing his or her knowledge. The learning of learner phase can be expressed as follows. For the ith individual X_i in the jth generation, randomly select the kth individual X_k which is different from X_i, and the updated formula of X_i is defined in (6) and (7).

If X_i is better than X_k according to their fitness, then

$$X_{i,\text{new}} = X_{i,\text{old}} + r_i \left(X_i - X_k\right). \tag{6}$$

Else

$$X_{i,\text{new}} = X_{i,\text{old}} + r_i \left(X_k - X_i\right), \tag{7}$$

where r_i is the random number in the range [0, 1].

If the new position $X_{i,\text{new}}$ is better than the old one $X_{i,\text{old}}$, the old position $X_{i,\text{old}}$ is replaced by the new $X_{i,\text{new}}$; otherwise, the position of the ith individual is not changed. The detail algorithm is shown in Algorithm 2.

4. Teaching-Learning-Based Optimization Algorithm with PSO (TLBO-PSO)

4.1. The Main Idea of TLBO-PSO. As previously reviewed, the main idea of TLBO is to imitate the teaching-learning process in a class. The teacher tries to disseminate knowledge among the learners to increase the knowledge level of the whole class, and the learners also study knowledge from the others to improve its grade. Algorithm displays that all individuals update their positions based on the distance between one or two times of the mean solution and the teacher in teacher phase. An individual also renewed its position based on the distance between it and a randomly selected individual from the class. Equation (4) indicates that the teacher only improves the grades of the students by using the mean grade of the class; the distance between the teacher and the students is not considered in teacher phase. In PSO, the difference between the current best individual and the individual can help the individual improve its performance. Based on this idea, the method in PSO is introduced into TLBO to improve the learning efficiency of the TLBO algorithm. The main change for TLBO is represented in updating equation. Equation (4) in TLBO is modified as

$$\begin{aligned} X_{i,\text{new}} = X_{i,\text{old}} &+ r_1 \left(X_{\text{teacher}} - T_F X_{g\text{mean}}\right) \\ &+ r_2 \left(X_{\text{teacher}} - X_{i,\text{old}}\right), \end{aligned} \tag{8}$$

where r_1, r_2 are the random number in the range [0, 1]. With this modification, the performance of TLBO algorithm might be improved.

In original TLBO algorithm, all genes of each individual should be compared with those of all other individuals for removing the duplicate individual. This operator is a heavy computation cost process, and the function evaluations required are not clearly known. In our opinion, duplication testing is not needed in every generation especially in the beginning of evolution. Assume that individuals i and k have the same genes in t generation, and the new position of these two individuals might be different when T_F is a random number (1 or 2). At the beginning of evolution, all individuals generate better positions easily. Random operator for T_F may benefit for maintaining diversity of class, but, in the anaphase of evolution, the positions of individuals might be close to each other. When the mean solution equals the best solution, the individual does not change (T_F = 1) or the large change of genes might destroy the individual in large degree (T_F = 2) so that the better individual is difficult to be generated. To decrease the computational effort of comparing all the individuals, the removing of the duplicate individual process in the original TLBO is deleted in the improved

```
Initialize N (number of learners) and D (number of dimensions)
Initialize learners X and evaluate all learners X
Donate the best learner as X_teacher and the mean of all learners X as X_gmean
while (stopping condition not met)
    for each learner X_i of the class    % Teaching phase
        T_F = round(1 + rand(0, 1))
        X_i,new = X_i,old + r_1(X_teacher − T_F X_gmean)
        Accept X_i,new if f(X_i,new) is better than f(X_i,old)
    endfor
    for each learner X_i of the class    % Learning phase
        Randomly select one learner X_k, such that i ≠ k
        if f(X_i) better f(X_k)
            X_i,new = X_i,old + r_i(X_i − X_k)
        else
            X_i,new = X_i,old + r_i(X_k − X_i)
        endif
        Accept X_i,new if f(X_i,new) is better than f(X_i,old)
    endfor
    Update X_teacher and X_gmean
endwhile
```

Algorithm 2: TLBO().

```
pop = sort(pop);
if abs(bestfit(gen) − bestfit(gen − 1)) < ε then m = m + 1; else m = 0;
    if (m == n)
        m = 0;
        for I = 2 : popsize
            if rand(1) < p_c
                k = ceil(rand(1) ∗ dimsize);
                pop(i, k) = pop(i, k) + α ∗ rands(1, 1);
                if pop(i, k) > x_max then pop(i, k) = x_max;
                if pop(i, k) < x_min then pop(i, k) = x_min;
            end
        end
    end
end
```

Algorithm 3: Subroutine for mutation.

TLBO algorithm and a mutation operator according to the best fitness of successive generations is introduced in the paper. If the best fitness of continuous n generations is not changed or changed slightly, an individual will be randomly selected according to mutation possibility p_c to be mutated. To maintain the global performance of the algorithm, the best individual is not mutated. The subroutine for the mutation is described as shown in Algorithm 3.

In Algorithm 3, n is the setting generation, p_c is the mutation possibility, and ε is a small number which is given by the designer. α is a mutation parameter between 0.01 and 0.1.

4.2. The Steps of TLBO-PSO

Step 1. Set the maximum $X_{\max}$ and minimum $X_{\min}$ of position, the maximal evolution generation genmax, mutation possibility p_c and mutation parameter α, the population size popsize, and the dimension size of the task. Initialize the initial population pop as follows:

$$\text{pop} = X_{\min} + r * \left(X_{\max} - X_{\min}\right), \tag{9}$$

where r is the random number in the range $[0, 1]$.

Step 2. Evaluate the individual, select the best individual X_{teacher} as the teacher, and calculate the mean solution $X_{g\text{mean}}$ of the population.

Step 3. For each individual, update its position according to (8). If $X_{i,\text{new}}$ is better than $X_{i,\text{old}}$, then $X_{i,\text{old}} = X_{i,\text{new}}$.

Step 4. For each individual, randomly select another individual, update its position according to (6) and (7), and choose

Table 1: Nine tested functions.

Function	Formula	Range	$f_{\min}$	Acceptance
f_1 (Sphere)	$f_1(x) = \sum_{i=1}^{D} x_i^2$	$[-100, 100]$	0	$1E-6$
f_2 (Quadric)	$f_2(x) = \sum_{i=1}^{D} \left(\sum_{j=1}^{i} x_j \right)^2$	$[-100, 100]$	0	$1E-6$
f_3 (Sum Square)	$f_3(x) = \sum_{i=1}^{D} i x_i^2$	$[-100, 100]$	0	$1E-6$
f_4 (Zakharov)	$f_4(x) = \sum_{i=1}^{D} x_i^2 + \left(\sum_{i=1}^{D} 0.5 i x_i \right)^2 + \left(\sum_{i=1}^{D} 0.5 i x_i \right)^4$	$[-10, 10]$	0	$1E-6$
f_5 (Rosenbrock)	$f_5(x) = \sum_{i=1}^{D-1} \left[100 \left(x_i^2 - x_{i+1} \right)^2 + \left(x_i - 1 \right)^2 \right]$	$[-2.048, 2.048]$	0	$1E-6$
f_6 (Ackley)	$f_6(x) = 20 - 20 \exp\left(-\frac{1}{5} \sqrt{\frac{1}{D} \sum_{i=1}^{D} x_i^2} \right) - \exp\left(\frac{1}{D} \sum_{i=1}^{D} \cos(2\pi x_i) \right) + e$	$[-32.768, 32.768]$	0	$1E-6$
f_7 (Rastrigin)	$f_7(x) = \sum_{i=1}^{D} \left(x_i^2 - 10 \cos(2\pi x_i) + 10 \right)$	$[-5.12, 5.12]$	0	$1E-6$
f_8 (Weierstrass)	$f_8(x) = \sum_{i=1}^{D} \left(\sum_{k=0}^{k_{\max}} \left[a^k \cos\left(2\pi b^k (x_i + 0.5) \right) \right] \right) - D \sum_{k=0}^{k_{\max}} \left[a^k \cos\left(2\pi b^k \times 0.5 \right) \right]$ $a = 0.5 \quad b = 3 \quad k_{\max} = 20$	$[-0.5, 0.5]$	0	$1E-1$
f_9 (Griewank)	$f_9(x) = \sum_{i=1}^{D} \frac{x_i^2}{4000} - \prod_{i=1}^{n} \cos\left(\frac{x_i}{\sqrt{i}} \right) + 1$	$[-600, 600]$	0	$1E-1$

the better solution from $X_{i,\text{old}}$ and $X_{i,\text{new}}$ as the new position of the individual.

Step 5. Execute mutation operator for the population according to Algorithm 3.

Step 6. If the ended condition of TLBO-PSO is not satisfied, the algorithm will go back to Step 2, or it is terminated.

5. Simulation Experiments

To test the performance of the improved TLBO-PSO algorithm, nine benchmark functions are simulated in this section. These nine benchmark functions are listed in Table 1. In particular, in order to compare the performance of the proposed TLBO-PSO with variants of TLBO and PSO, CLPSO [22], TLBO [15], and ETLBO [21] are selected and simulated.

5.1. Parameters Setting. To reduce statistical errors, each function in the paper is independently simulated 30 runs, and their mean results are used in the comparison. The value of function is defined as the fitness function. All the experiments are carried out on the same machine with a Celeron 2.26 GHz CPU, 512-MB memory, and Windows XP operating system with Matlab 7.0. All functions are simulated in 10 and 30 dimensions. The nine functions are summarized in Table 1. "Range" is the lower and upper bounds of the variables. "$f_{\min}$" is the theory global minimum solution. "Acceptance" is the acceptable solutions of different functions. For CLPSO, TLBO, and ETLBO algorithms, the training parameters are the same as those used in the corresponding references except that the maximal FEs are 50000 and the size of population is 30. The size of elitism is 2 in ETLBO algorithm. When the best fitness is changed to be smaller than ε, the large value of mutation will possibly be chosen. In our experiments, the mutation possibility p_c is 0.6, ε is 0.01, and α is 0.05.

5.2. Comparisons on the Solution Accuracy. The performance of different algorithms for 10- and 30-dimensional functions in terms of the best solution, the mean (Mean), and standard deviation (STD) of the solutions obtained in the 30 independent runs is listed in Tables 2 and 3. Boldface in the tables indicates the best solution among those obtained by all four contenders.

The results for 10-dimensional functions in Table 2 display that the improved TLBO (TLBO-PSO) outperforms all the other algorithms in terms of the mean best solutions and standard deviations (STD) for functions f_1, f_2, f_3, and f_4. For function f_5, ETLBO has the best performance. For function f_6, the mean best solution of TLBO-PSO is the best, and the STD of TLBO is the smallest. For functions f_7 and f_9, CLPSO has the best performance. For function f_8, three TLBOs have the same solutions in terms of the mean best solutions and standard deviations (STD).

The results for 30-dimensional functions in Table 3 indicate that TLBO-PSO also has the best performance in terms of the mean best solutions and standard deviations (STD) for functions f_1, f_2, f_3, and f_4 with 30 dimensions. TLBO has the best performance for function f_5. For function f_6, TLBO and ETLBO have the same performance in terms of the mean best solutions and standard deviations (STD). For function f_7, the mean best solution of TLBO is the smallest, and the standard

Table 2: The mean best solutions and standard deviations by various methods for 10D functions.

Methods	CLPSO		TLBO		ETLBO		TLBO-PSO	
	Mean	STD	Mean	STD	Mean	STD	Mean	STD
f_1	$4.68E-19$	$4.90E-19$	$2.12E-185$	$0.00E+00$	$1.51E-167$	$0.00E+00$	**1.07E − 314**	**0.00E + 00**
f_2	$6.73E-01$	$4.40E-01$	$5.79E-80$	$1.71E-79$	$5.10E-78$	$1.22E-77$	**9.62E − 172**	**0.00E + 00**
f_3	$1.14E-20$	$8.02E-21$	$1.02E-185$	$0.00E+00$	$5.91E-169$	$0.00E+00$	**2.65E − 315**	**0.00E + 00**
f_4	$3.32E-03$	$3.25E-03$	$4.44E-87$	$1.36E-86$	$9.99E-85$	$1.89E-84$	**5.04E − 182**	**0.00E + 00**
f_5	$2.25E+00$	$1.00E+00$	$6.54E-01$	$6.56E-01$	**2.76E − 01**	**2.78E − 01**	$4.18E+00$	$7.46E-01$
f_6	$3.75E-10$	$2.83E-10$	$3.55E-15$	**0.00E + 00**	$3.20E-15$	$1.12E-15$	**2.84E − 15**	$1.50E-15$
f_7	**1.63E − 09**	**2.42E − 09**	$2.17E+00$	$1.86E+00$	$2.33E+00$	$1.76E+00$	$5.26E+00$	$4.63E+00$
f_8	$1.16E-11$	$1.55E-11$	**0.00E + 00**	**0.00E + 00**	**0.00E + 00**	**0.00E + 00**	**0.00E + 00**	**0.00E + 00**
f_9	**1.06E − 03**	**2.29E − 03**	$9.96E-03$	$1.02E-02$	$8.45E-03$	$1.61E-02$	$3.54E-02$	$5.27E-02$

Table 3: The mean best solutions and standard deviations by various methods for 30D functions.

Methods	CLPSO		TLBO		ETLBO		TLBO-PSO	
	Mean	STD	Mean	STD	Mean	STD	Mean	STD
f_1	$1.50E-03$	$5.04E-04$	$1.63E-152$	$2.38E-152$	$5.70E-132$	$1.13E-131$	**6.67E − 240**	**0.00E + 00**
f_2	$9.37E+03$	$2.23E+03$	$1.78E-33$	$3.34E-33$	$4.58E-33$	$5.40E-33$	**1.98E − 76**	**4.73E − 76**
f_3	$1.39E-04$	$4.19E-05$	$1.45E-152$	$2.79E-152$	$7.13E-132$	$1.78E-131$	**1.60E − 240**	**0.00E + 00**
f_4	$1.45E+02$	$4.10E+01$	$5.72E-18$	$7.60E-18$	$1.56E-16$	$3.00E-16$	**1.46E − 43**	**3.04E − 43**
f_5	$3.60E+01$	$1.03E+01$	**2.42E + 01**	**6.48E − 01**	$2.46E+01$	$7.50E-01$	$2.65E+01$	$1.18E+00$
f_6	$3.19E-02$	$1.27E-02$	**3.55E − 15**	**0.00E + 00**	$3.91E-15$	$1.12E-15$	**3.55E − 15**	**0.00E + 00**
f_7	$9.88E+00$	$2.48E+00$	**1.31E + 01**	$5.78E+00$	$1.68E+01$	$8.12E+00$	$2.41E+01$	**1.40E + 01**
f_8	$6.53E-02$	$1.08E-02$	**0.00E + 00**	**0.00E + 00**	**0.00E + 00**	**0.00E + 00**	**0.00E + 00**	**0.00E + 00**
f_9	$8.76E-03$	$3.43E-03$	**0.00E + 00**	**0.00E + 00**	**0.00E + 00**	**0.00E + 00**	**0.00E + 00**	**0.00E + 00**

Table 4: The mean FEs needed to reach an acceptable solution and reliability "ratio" being the percentage of trial runs reaching acceptable solutions for 10-dimensional functions.

Methods	CLPSO		TLBO		ETLBO		TLBO-PSO	
	mFEs	Ratios	mFEs	Ratios	mFEs	Ratios	mFEs	Ratios
f_1	27041.1	**100.0**%	2744.9	**100.0**%	3044.1	**100.0**%	**1640**	**100.0**%
f_2	NaN	0.0%	5748	**100.0**%	6106.3	**100.0**%	**2858.2**	**100.0**%
f_3	24058.3	**100.0**%	2336.1	**100.0**%	2631.1	**100.0**%	**1449.2**	**100.0**%
f_4	NaN	0.0%	**5999.3**	**100.0**%	6040.2	**100.0**%	**3084.3**	**100.0**%
f_5	NaN	0.0%	45112	20.0%	**43793.5**	**40.0**%	NaN	0.0%
f_6	38089.8	**100.0**%	4201.4	**100.0**%	4577.4	**100.0**%	**2480.4**	**100.0**%
f_7	26783.3	**100.0**%	23632.8	**50.0**%	17727.2	**50.0**%	**12833**	40.0%
f_8	41009.2	**100.0**%	6110.1	**100.0**%	6686.2	**100.0**%	**3278.9**	**100.0**%
f_9	24268	**100.0**%	4123.3	**100.0**%	5467	**100.0**%	**2796.2**	90.0%

deviation of TLBO-PSO is the smallest. Three TLBOs have the same solutions for functions f_8 and f_9.

5.3. Comparisons on the Convergence Speed and Successful Ratios. The mean number of function evolutions (FEs) is often used to measure the convergence speed of algorithms. In this paper, the mean value of FEs is used to measure the speed of all algorithms. The average FEs (when the algorithm is globally convergent) of all algorithms for nine functions with 10 and 30 dimensions are shown in Tables 4 and 5. Boldface in the tables indicates the best result among those obtained by all algorithms. When the algorithm is not globally convergent in all 30 runs, the mFEs is represented as "NaN." The successful ratios of different algorithms for the nine functions are also shown in the tables.

Tables 4 and 5 display that mFEs of TLBO-PSO are the smallest for large part of functions except that for function f_5. The merit in terms of mean FEs for 10-dimensional function f_5 with ETLBO is the best and the four algorithms cannot converge to the acceptable solution for 30-dimensional function. For function f_6, all algorithms can converge to the global optima with 100% successful ratios on 30 dimensions except

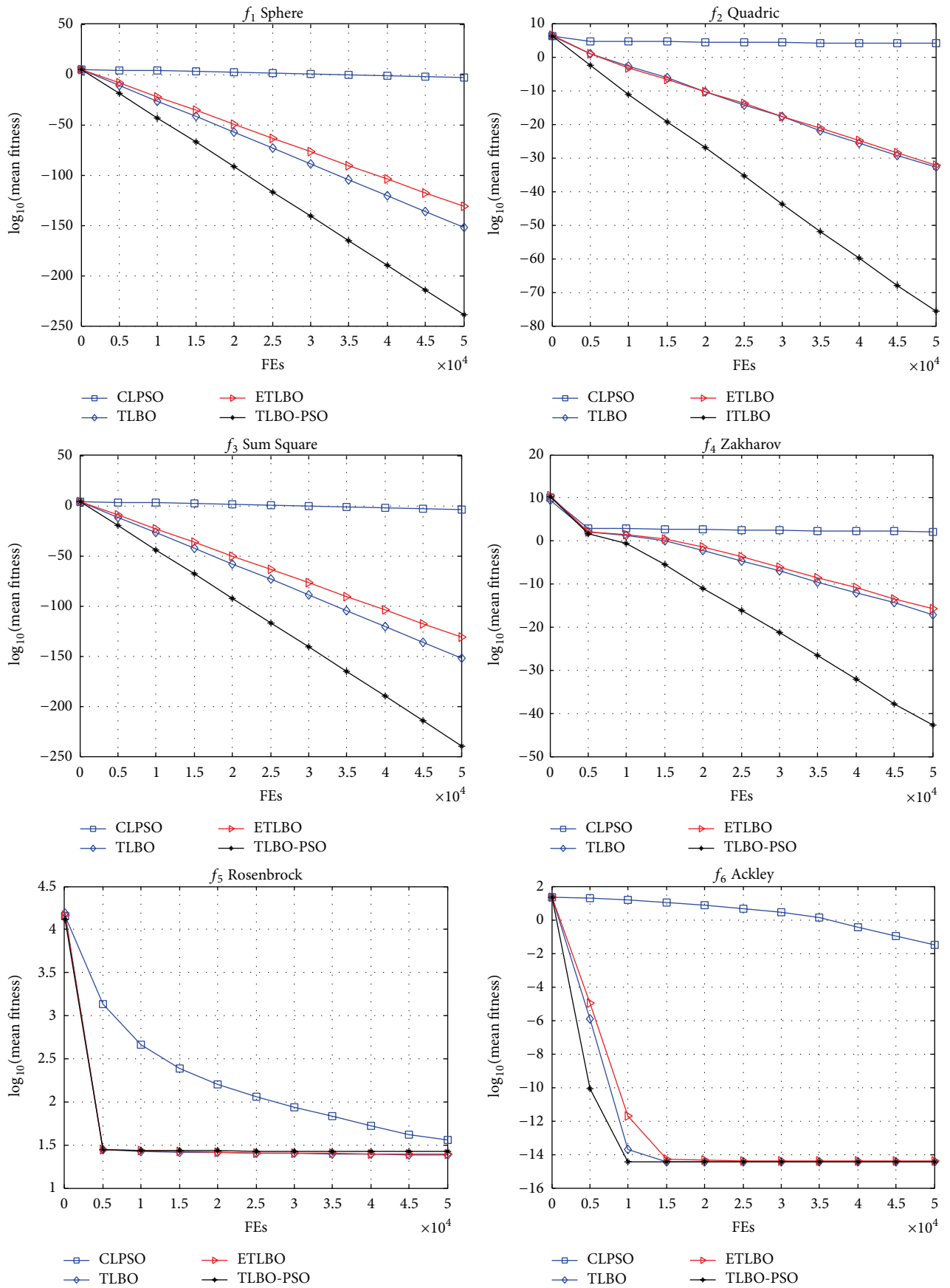

Figure 1: Continued.

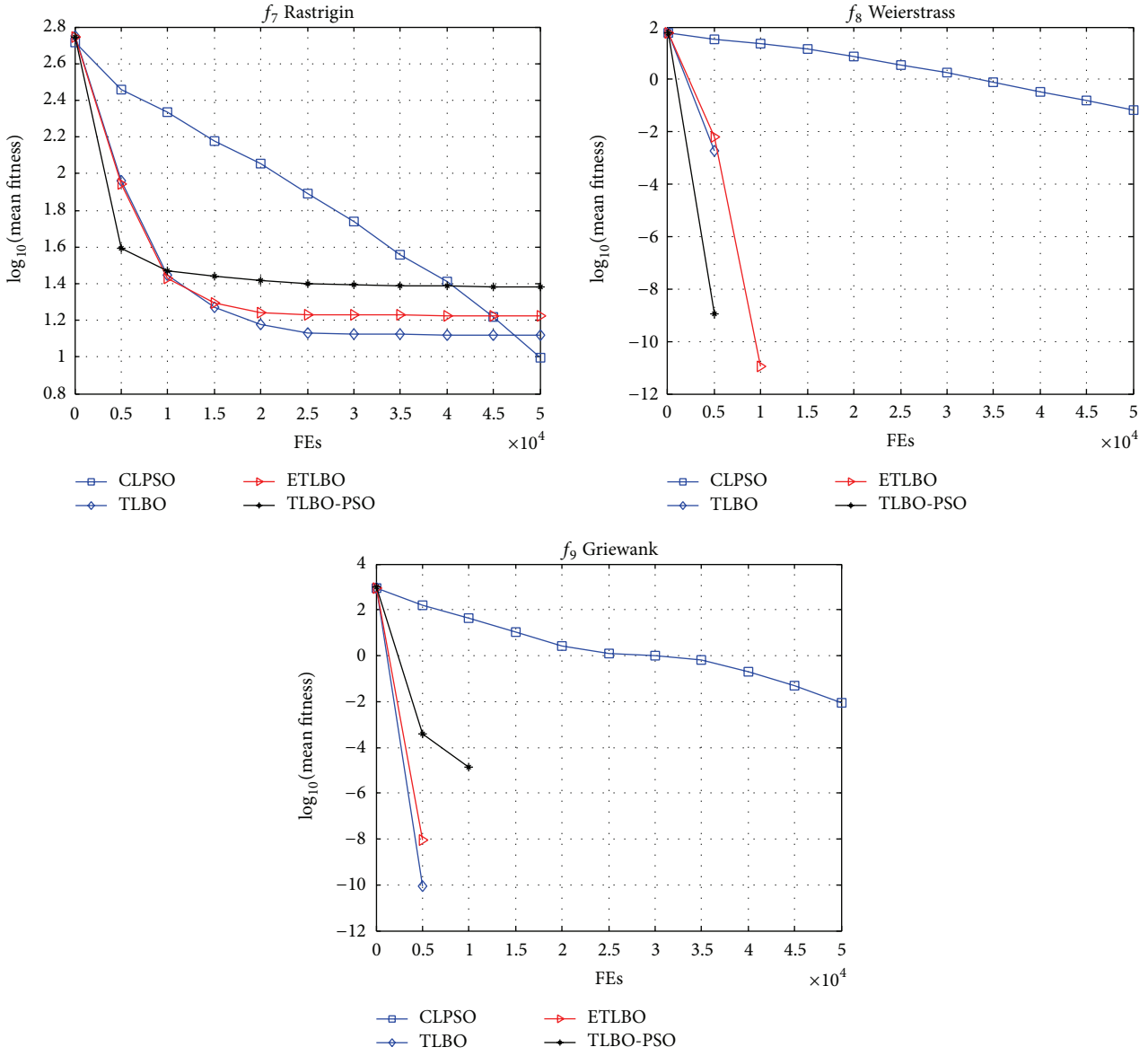

Figure 1: Convergence performance of the four different methods for 30-dimensional functions.

Table 5: The mean FEs needed to reach an acceptable solution and reliability "ratio" being the percentage of trial runs reaching acceptable solutions for 30-dimensional functions.

Methods	CLPSO		TLBO		ETLBO		TLBO-PSO	
	mFEs	Ratios	mFEs	Ratios	mFEs	Ratios	mFEs	Ratios
f_1	0.0%	3502.8	**100.0%**	4039.5	**100.0%**	**2327.9**	**100.0%**	0.0%
f_2	0.0%	13845.3	**100.0%**	13699.8	**100.0%**	**6802.7**	**100.0%**	0.0%
f_3	0.0%	3214.1	**100.0%**	3716.9	**100.0%**	**2093**	**100.0%**	0.0%
f_4	0.0%	26934.9	**100.0%**	27876.2	**100.0%**	**14067.3**	**100.0%**	0.0%
f_5	0.0%	NaN	0.0%	NaN	0.0%	NaN	0.0%	0.0%
f_6	0.0%	5039.1	**100.0%**	5744.6	**100.0%**	**3305.4**	**100.0%**	0.0%
f_7	0.0%	NaN	0.0%	NaN	0.0%	**4629**	**10.0%**	0.0%
f_8	0.0%	7188.8	**100.0%**	8125.9	**100.0%**	**4315.3**	**100.0%**	0.0%
f_9	**100.0%**	2046.8	**100.0%**	2361	**100.0%**	**1492.5**	**100.0%**	**100.0%**

CLPSO. For function f_7, CLPSO can converge to optima with 100% successful ratios for 10-dimensional function. They all cannot converge to 30-dimensional function except that the successful ratio of TLBO-PSO is 10%. The convergence speed of average best fitness of four algorithms is shown in Figure 1. The figures indicate that the TLBO-PSO has the best performance for large part of functions. According to the theorem of "no free lunch" [28], one algorithm cannot offer better performance than all the others on every aspect or on every kind of problem. This is also observed in our experimental results. For example, the merit of TLBO-PSO is worse than those of TLBO and ETLBO for 10-dimension function but it is better than large part of algorithms for other functions.

6. Conclusions

An improved TLBO-PSO algorithm which is considering the difference between the solutions of the best individual and the individual that want to be renewed is designed in the paper. The mutation operator is introduced to improve the global convergence performance the algorithm. The performance of TLBO-PSO is improved for larger part of functions especially for 30-dimension functions in terms of convergence accuracy and mFEs. The local convergence of TLBO-PSO is major caused by the lost diversity in the later stage of the evolution.

Further works include researches into adaptive selection of parameters to make the algorithm more efficient. Moreover, it needs to seek a better method to improve the TLBO algorithm for functions with optimal parameters displaced from all zeroes. Furthermore, the algorithm may be applied to constrained, dynamic optimization domain. It is expected that TLBO-PSO will be used in real-world optimization problems.

Conflict of Interests

The authors declare that there is no conflict of interests regarding the publication of this paper.

Acknowledgments

This work is partially supported by National Natural Science Foundations of China under Grants 61304082 and 61572224 and the National Science Fund for Distinguished Young Scholars under Grant 61425009, and it is partially supported by the Major Project of Natural Science Research in Anhui Province under Grant KJ2015ZD36.

References

[1] C. A. Floudas and C. E. Gounaris, "A review of recent advances in global optimization," *Journal of Global Optimization*, vol. 45, no. 1, pp. 3–38, 2009.

[2] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Proceedings of the IEEE International Conference on Neural Networks*, vol. 4, pp. 1942–1948, IEEE, Perth, Australia, November-December 1995.

[3] S. L. Sabat, L. Ali, and S. K. Udgata, "Integrated learning particle swarm optimizer for global optimization," *Applied Soft Computing*, vol. 11, no. 1, pp. 574–584, 2011.

[4] Y. Shi and R. Eberhart, "A modified particle swarm optimizer," in *Proceedings of the IEEE International Conference on Evolutionary Computation (ICEC '98)*, pp. 69–73, IEEE, Anchorage, Alaska, USA, May 1998.

[5] M. Pant, R. Thangaraj, and V. P. Singh, "Particle Swarm optimization with crossover operator and its engineering applications," *IAENG International Journal of Computer Science*, vol. 36, no. 2, 2009.

[6] T.-O. Ting, M. V. C. Rao, C. K. Loo, and S.-S. Ngu, "A new class of operators to accelerate particle swarm optimization," in *Proceedings of the Congress on Evolutionary Computation (CEC '03)*, vol. 4, pp. 2406–2410, IEEE, December 2003.

[7] C. Li, Y. Liu, A. Zhao, L. Kang, and H. Wang, "A fast particle swarm algorithm with cauchy mutation and natural selection strategy," in *Advances in Computation and Intelligence*, vol. 4683 of *Lecture Notes in Computer Science*, pp. 334–343, Springer, Berlin, Germany, 2007.

[8] H. Zhang, "An analysis of multiple particle swarm optimizers with inertia weight for multi-objective optimization," *IAENG International Journal of Computer Science*, vol. 39, no. 2, pp. 190–199, 2012.

[9] M. M. Ali and P. Kaelo, "Improved particle swarm algorithms for global optimization," *Applied Mathematics and Computation*, vol. 196, no. 2, pp. 578–593, 2008.

[10] Z.-H. Zhan, J. Zhang, Y. Li, and H. S.-H. Chung, "Adaptive particle swarm optimization," *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, vol. 39, no. 6, pp. 1362–1381, 2009.

[11] Z.-H. Zhan and J. Zhang, "Adaptive particle swarm optimization," in *Ant Colony Optimization and Swarm Intelligence*, vol. 5217 of *Lecture Notes in Computer Science*, pp. 227–234, Springer, Berlin, Germany, 2008.

[12] A. Banks, J. Vincent, and C. Anyakoha, "A review of particle swarm optimization. Part I: background and development," *Natural Computing*, vol. 6, no. 4, pp. 467–484, 2007.

[13] A. Banks, J. Vincent, and C. Anyakoha, "A review of particle swarm optimization. Part II: hybridisation, combinatorial, multicriteria and constrained optimization, and indicative applications," *Natural Computing*, vol. 7, no. 1, pp. 109–124, 2008.

[14] S. Rana, S. Jasola, and R. Kumar, "A review on particle swarm optimization algorithms and their applications to data clustering," *Artificial Intelligence Review*, vol. 35, no. 3, pp. 211–222, 2011.

[15] R. V. Rao, V. J. Savsani, and D. P. Vakharia, "Teaching-learning-based optimization: an optimization method for continuous non-linear large scale problems," *Information Sciences*, vol. 183, no. 1, pp. 1–15, 2012.

[16] R. V. Rao, V. J. Savsani, and D. P. Vakharia, "Teaching-learning-based optimization: a novel method for constrained mechanical design optimization problems," *Computer-Aided Design*, vol. 43, no. 3, pp. 303–315, 2011.

[17] R. V. Rao, V. J. Savsani, and J. Balic, "Teaching–learning-based optimization algorithm for unconstrained and constrained real-parameter optimization problems," *Engineering Optimization*, vol. 44, no. 12, pp. 1447–1462, 2012.

[18] C. S. Suresh and N. Anima, "Data clustering based on teaching-learning-based optimization," in *Swarm, Evolutionary, and Memetic Computing: Second International Conference, SEMCCO 2011, Visakhapatnam, Andhra Pradesh, India, December*

19–21, 2011, Proceedings, Part II, vol. 7077 of *Lecture Notes in Computer Science*, pp. 148–156, 2011.

[19] V. Toĝan, "Design of planar steel frames using Teaching-Learning Based Optimization," *Engineering Structures*, vol. 34, pp. 225–232, 2012.

[20] H. Hossein, N. Taher, and I. T. Seyed, "A Modified TLBO algorithm for placement of AVRs considering DGs," in *Proceedings of the 26th International Power System Conference*, Theran, Iran, October 2011.

[21] R. V. Rao and V. Patel, "An elitist teaching-learning-based optimization algorithm for solving complex constrained optimization problems," *International Journal of Industrial Engineering Computations*, vol. 3, no. 4, pp. 535–560, 2012.

[22] J. J. Liang, A. K. Qin, P. N. Suganthan, and S. Baskar, "Comprehensive learning particle swarm optimizer for global optimization of multimodal functions," *IEEE Transactions on Evolutionary Computation*, vol. 10, no. 3, pp. 281–295, 2006.

[23] W.-J. Zhang and X.-F. Xie, "DEPSO: hybrid particle swarm with differential evolution operator," in *Proceedings of the IEEE International Conference on Systems, Man and Cybernetics*, vol. 4, pp. 3816–3821, IEEE, Washington, DC, USA, October 2003.

[24] H.-J. Meng, P. Zheng, R.-Y. Wu, X.-J. Hao, and Z. Xie, "A hybrid particle swarm algorithm with embedded chaotic search," in *Proceedings of the IEEE Conference on Cybernetics and Intelligent Systems*, pp. 367–371, Singapore, December 2004.

[25] A. Kaveh and S. Talatahari, "A hybrid particle swarm and ant colony optimization for design of truss structures," *Asian Journal of Civil Engineering*, vol. 9, pp. 329–348, 2008.

[26] V. Plevris and M. Papadrakakis, "A hybrid particle swarm—gradient algorithm for global structural optimization," *Computer-Aided Civil and Infrastructure Engineering*, vol. 26, no. 1, pp. 48–68, 2011.

[27] H. Duan, Q. Luo, Y. Shi, and G. Ma, "Hybrid particle swarm optimization and genetic algorithm for multi-UAV formation reconfiguration," *IEEE Computational Intelligence Magazine*, vol. 8, no. 3, pp. 16–27, 2013.

[28] D. H. Wolpert and W. G. Macready, "No free lunch theorems for optimization," *IEEE Transactions on Evolutionary Computation*, vol. 1, no. 1, pp. 67–82, 1997.