Research Article

Bare-Bones Teaching-Learning-Based Optimization

Feng Zou, Lei Wang, Xinhong Hei, Debao Chen, Qiaoyong Jiang, and Hongye Li

1 School of Computer Science and Engineering, Xi’an University of Technology, Xi’an 710048, China
2 School of Physics and Electronic Information, Huaihe Normal University, Huaihe 235000, China

Correspondence should be addressed to Lei Wang; wangleeei@163.com

Received 20 February 2014; Accepted 7 April 2014; Published 10 June 2014

Academic Editors: S. Balochian and Y. Zhang

Copyright © 2014 Feng Zou et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Teaching-learning-based optimization (TLBO) algorithm which simulates the teaching-learning process of the classroom is one of the recently proposed swarm intelligent (SI) algorithms. In this paper, a new TLBO variant called bare-bones teaching-learning-based optimization (BBTLBO) is presented to solve the global optimization problems. In this method, each learner of teacher phase employs an interactive learning strategy, which is the hybridization of the learning strategy of teacher phase in the standard TLBO and Gaussian sampling learning based on neighborhood search, and each learner of learner phase employs the learning strategy of learner phase in the standard TLBO or the new neighborhood search strategy. To verify the performance of our approaches, 20 benchmark functions and two real-world problems are utilized. Conducted experiments can be observed that the BBTLBO performs significantly better than, or at least comparable to, TLBO and some existing bare-bones algorithms. The results indicate that the proposed algorithm is competitive to some other optimization algorithms.

1. Introduction

Many real-life optimization problems are becoming more and more complex and difficult with the development of scientific technology. So how to resolve these complex problems in an exact manner within a reasonable time cost is very important. The traditional optimization algorithms are difficult to solve these complex nonlinear problems. In recent years, nature-inspired optimization algorithms which simulate natural phenomena and have different design philosophies and characteristics, such as evolutionary algorithms [1–3] and swarm intelligence algorithms [4–7], are a research field which simulates different natural phenomena to solve a wide range of problems. In these algorithms the convergence rate of the algorithm is given prime importance for solving real-world optimization problems. The ability of the algorithms to obtain the global optima value is one aspect and the faster convergence is the other aspect.

As a stochastic search scheme, TLBO [8, 9] is a newly population-based algorithm based on swarm intelligence and has characters of simple computation and rapid convergence; it has been extended to the function optimization, engineering optimization, multiobjective optimization, clustering, and so forth [9–17]. TLBO is a parameter-free evolutionary technique and is also gaining popularity due to its ability to achieve better results in comparatively faster convergence time to genetic algorithms (GA) [1], particle swarm optimizer (PSO) [5], and artificial bee colony algorithm (ABC) [6]. However, in evolutionary computation research there have been always attempts to improve any given findings further and further. This work is an attempt to improve the convergence characteristics of TLBO further without sacrificing the accuracies obtained in TLBO and in some occasions trying to even better the accuracies. The aims of this paper are of threefold. First, authors propose an improved version of TLBO, namely, BBTLBO. Next, the proposed technique is validated on unimodal and multimodal functions based on different performance indicators. The result of BBTLBO is compared with other algorithms. Results of both the algorithms are also compared using statistical paired t-test. Thirdly, it is applied to solve the real-world optimization problem.

The remainder of this paper is organized as follows. The TLBO algorithm is introduced in Section 2. Section 3 presents a brief overview of some recently proposed
Algorithm 1: TLBO().

bare-bones algorithms. Section 4 describes the improved teaching-learning-based optimization algorithm using neighborhood search (BBTLBO), Section 5 presents the tests on several benchmark functions and the experiments are conducted along with statistical tests. The applications for training artificial neural network are shown in Section 6. Conclusions are given in Section 7.

2. Teaching-Learning-Based Optimization

Rao et al. [8, 9] first proposed a novel teaching-learning-based optimization (TLBO) inspired from the philosophy of teaching and learning. The TLBO algorithm is based on the effect of the influence of a teacher on the output of learners in a class which is considered in terms of results or grades. The process of working of TLBO is divided into two parts. The first part consists of “teacher phase” and the second part consists of “learner phase.” The “teacher phase” means learning from the teacher and the “learner phase” means learning through the interaction between learners.

A good teacher is one who brings his or her learners up to his or her level in terms of knowledge. But in practice this is not possible and a teacher can only move the mean of a class up to some extent depending on the capability of the class. This follows a random process depending on many factors. Let $M$ be the mean and let $T$ be the teacher at any iteration. $T$ will try to move mean $M$ toward its own level, so now the new mean will be $T$ designated as $M_{\text{new}}$. The solution is updated according to the difference between the existing and the new mean according to the following expression:

$$
\text{new}X = X + r \times (M_{\text{new}} - T \times M),
$$

where $TF$ is a teaching factor that decides the value of mean to be changed and $r$ is a random vector in which each element is a random number in the range $[0, 1]$. The value of $TF$ can be either 1 or 2, which is again a heuristic step and decided randomly with equal probability as

$$
TF = \text{round}[1 + \text{rand}(0, 1)].
$$

Learners increase their knowledge by two different means: one through input from the teacher and the other through interaction between themselves. A learner interacts randomly with other learners with the help of group discussions, presentations, formal communications, and so forth. A learner learns something new if the other learner has more knowledge than him or her. Learner modification is expressed as

$$
\text{new}X_i = \begin{cases} 
X_i + r \times (X_i - X_j) & \text{if } f(X_i) < f(X_j) \\
X_i + r \times (X_j - X_i) & \text{otherwise}
\end{cases}
$$

As explained above, the pseudocode for the implementation of TLBO is summarized in Algorithm 1.
3. Bare-Bones Algorithm

In this section, we only presented a brief overview of some recently proposed bare-bones algorithms.

3.1. BBPSO and BBExp. PSO is a swarm intelligence-based algorithm, which is inspired by the behavior of birds flocking [5]. In PSO, each particle is attracted by its personal best position \(p_{\text{best}}\) and the global best position \(g_{\text{best}}\) found so far. Theoretical studies [18, 19] proved that each particle converges to the weighted average of \(p_{\text{best}}\) and \(g_{\text{best}}\):

\[
\lim_{t \to \infty} X_i(t) = \frac{c_1 \cdot g_{\text{best}} + c_2 \cdot p_{\text{best}}}{c_1 + c_2},
\]

where \(c_1\) and \(c_2\) are two leaning factors in PSO.

Based on the convergence characteristic of PSO, Kennedy [20] proposed a new PSO variant called bare-bones PSO (BBPSO). Bare-bones PSO retains the standard PSO social communication but replaces dynamical particle update with sampling from a probability distribution based on \(g_{\text{best}}\) and \(p_{\text{best}}\), as follows:

\[
x_{i,j}(t + 1) = N \left( \frac{g_{\text{best}} + p_{\text{best},i,j}(t)}{2}, \left| g_{\text{best}} - p_{\text{best},i,j}(t) \right| \right),
\]

where \(x_{i,j}(t + 1)\) is the \(j\)th dimension of the \(i\)th particle in the population and \(N\) represents a Gaussian distribution with mean \(g_{\text{best}} + p_{\text{best},i,j}(t))\)/2 and standard deviation \(|g_{\text{best}} - p_{\text{best},i,j}(t)|\).

Kennedy [20] proposed also an alternative version of the BBPSO, denoted by BBExp, where (5) is replaced by:

\[
x_{i,j}(t + 1) = \begin{cases} N \left( \frac{g_{\text{best}} + p_{\text{best},i,j}(t)}{2}, \left| g_{\text{best}} - p_{\text{best},i,j}(t) \right| \right) & \text{rand}(0, 1) > 0.5 \\ p_{\text{best},i,j}(t) & \text{otherwise}, \end{cases}
\]

where \(\text{rand}(0, 1)\) is a random value within \([0, 1]\) for the \(j\)th dimension. For the alternative mechanism, there is a 50% chance that the search process is focusing on the previous best positions.

3.2. BBDE, GBDE, and MGBDE. Inspired by the BBPSO and DE, Omran et al. [21] proposed a new and efficient DE variant, called bare-bones differential evolution (BBDE). The BBDE is a new, almost parameter-free optimization algorithm that is a hybrid of the bare-bones particle swarm optimizer and differential evolution. Differential evolution is used to mutate, for each particle, the attractor associated with that particle, defined as a weighted average of its personal and neighborhood best positions. For the BBDE, the individual is updated as follows:

\[
x_{i,j}(t + 1) = \begin{cases} p_{\text{best},i,j}(t) + r_2 \cdot \left( x_{i_1,j}(t) - x_{i_2,j}(t) \right) & \text{rand}(0, 1) > \text{CR} \\ p_{\text{best},i,j}(t) & \text{otherwise}, \end{cases}
\]

where \(i_1, i_2,\) and \(i_3\) are three indices chosen from the set \(\{1, 2, \ldots, N\}\) with \(i_1 \neq i_2 \neq i_3\), \(\text{rand}(0, 1)\) is a random value within \([0, 1]\) for the \(j\)th dimension, and \(p_{\text{best},i,j}(t)\) is defined by

\[
p_{\text{best},i,j}(t + 1) = r_{1,j} \cdot p_{\text{best},i,j}(t) + \left( 1 - r_{2,j} \right) g_{\text{best},i,j}(t),
\]

where \(p_{\text{best}}\) and \(g_{\text{best}}\) are personal best position and the global best position, \(r_{1,j}\) is a random value within \([0, 1]\) for the \(j\)th dimension.

Based on the idea that the Gaussian sampling is a fine tuning procedure which starts during exploration and is continued to exploitation, Wang et al. [22] proposed a new parameter-free DE algorithm, called GBDE. In the GBDE, the mutation strategy uses a Gaussian sampling method which is defined by

\[
v_{i,j}(t + 1) = \begin{cases} N \left( \frac{X_{\text{best},i,j}(t) + X_{i,j}(t)}{2}, \text{rand}(0, 1) \leq \text{CR} \vee j = j_{\text{rand}} \\ \left| X_{\text{best},i,j}(t) - X_{i,j}(t) \right| \end{cases} \text{otherwise},
\]

where \(N\) represents a Gaussian distribution with mean \((X_{\text{best},i,j}(t)+X_{i,j}(t))/2\) and standard deviation \(|X_{\text{best},i,j}(t)-X_{i,j}(t)|\) and CR is the probability of crossover.

To balance the global search ability and convergence rate, Wang et al. [22] proposed a modified GBDE (called MGBDE). The mutation strategy uses a hybridization of GBDE and DE/best/1 as follows:

\[
v_{i,j}(t + 1) = \begin{cases} X_{\text{best},i,j}(t) + F \cdot \left( X_{i,j}(t) - X_{\text{best},i,j}(t) \right) & \text{rand}(0, 1) \leq 0.5 \\ N \left( \frac{X_{\text{best},i,j}(t) + X_{i,j}(t)}{2}, \left| X_{\text{best},i,j}(t) - X_{i,j}(t) \right| \right) & \text{otherwise}. \end{cases}
\]

4. Proposed Algorithm: BBTLBO

The bare-bones PSO utilizes this information by sampling candidate solutions, normally distributed around the formally derived attractor point. That is, the new position is generated by a Gaussian distribution for sampling the search space based on the \(g_{\text{best}}\) and the \(p_{\text{best}}\) at the current iteration. As a result, the new position will be centered around the previous best at the current iteration. For the BBTLBO, the Gaussian sampling is a fine tuning procedure which starts during exploration and is continued to exploitation. This can be beneficial for the search of many evolutionary optimization algorithms. Additionally, the bare-bones PSO has no parameters to be tuned.

Based on a previous explanation, a new bare-bones TLBO (BBTLBO) with neighborhood search is proposed in this
Begin

Initialize learners size (NP), dimension (D), and hybridization factor (u)

Calculate the NTeacher and NMean of each learner

Modify each learner $X_i$ in the class

$$
V_1 = X_i + r \times (NTeacher - TF \times NMean) \\
V_2 = N((NTeacher + X_i)/2, (NTeacher - X_i))
$$

$$
newX_i = u \times newX_1 + (1 - u) \times newX_2
$$

Teacher phase

If $newX_i$ better $X_i$ then

Yes:

$X_i = newX_i$

No:

$X_i = X_i$

new $X_i$ better $X_i$

Denote the NTeacher and randomly select a $X_k$ for each $X_i$

Learner phase

If $rand(0, 1) < 0.5$ then

Yes:

The original TLBO learning

No:

Neighborhood search strategy

If $newX_i$ better $X_i$ then

Yes:

$X_i = newX_i$

No:

$X_i = X_i$

No:

Termination criteria satisfied

gen = gen + 1

End

Figure 1: Flow chart showing the working of BBTLBO algorithm.

In fact, for TLBO, if the new learner has a better function value than that of the old learner, it is replaced with the old one in the memory. Otherwise, the old one is retained in the memory. In other words, a greedy selection mechanism is employed as the selection operation between the old and the candidate one. Hence, the new teacher and the new learner are the global best ($g_{best}$) and learner’s personal best ($p_{best}$) found so far, respectively. The complete flowchart of the BBTLBO algorithm is shown in Figure 1.

4.1. Neighborhood Search. It is known that birds of a feather flock together and people of a mind fall into the same group. Just like evolutionary algorithms themselves, the notion of neighborhood is inspired by nature. Neighborhood technique is an efficient method to maintain diversity of the solutions. It plays an important role in evolutionary algorithms and is often introduced by researchers in order to allow maintenance of a population of diverse individuals and improve the exploration capability of population-based heuristic algorithms [23–26]. In fact, learners with similar interests form different learning groups. Because of his or her favor characteristic, the learner maybe learns from the excellent individual in the learning group.

For the implementation of grouping, various types of connected distances may be used. Here we have used a ring topology [27] based on the indexes of learners for the sake of simplicity. In a ring topology, the first individual is the neighbor of the last individual and vice versa. Based on the ring topology, a $k$-neighborhood radius is defined, where $k$ is a predefined integer number. For each individual,
its $k$-neighborhood radius consists of $2k + 1$ individuals (including oneself), which are $X_{i-k}, \ldots, X_i, \ldots, X_{i+k}$. That is, the neighborhood size is $2k + 1$ for a $k$-neighborhood. For simplicity, $k$ is set to 1 (Figure 2) in our algorithm. This means that there are 3 individuals in each learning group. Once groups are constructed, we can utilize them for updating the learners of the corresponding group.

4.2. Teacher Phase. To balance the global and local search ability, a modified interactive learning strategy is proposed in teacher phase. In this learning phase, each learner employs an interactive learning strategy (the hybridization of the learning strategy of teacher phase in the standard TLBO and Gaussian sampling learning) based on neighborhood search.

In BBTLBO, the updating formula of the learning for a learner $X_i$ in teacher phase is proposed by the hybridization of the learning strategy of teacher phase and the Gaussian sampling learning as follows:

$$V_{1,j}(t+1) = X_{i,j}(t) + \text{rand}(0,1) \cdot \left( N_{\text{Teacher},i,j}(t) - \text{TF} \cdot N_{\text{Mean},i,j}(t) \right),$$

$$V_{2,j}(t+1) = N \left( \frac{N_{\text{Teacher},i,j}(t) + N_{\text{Mean},i,j}(t)}{2} \right),$$

$$\text{new}X_{i,j}(t+1) = u \cdot V_{1,j}(t+1) + (1 - u) \cdot V_{2,j}(t+1),$$

where $u$ called the hybridization factor is a random number in the range $[0, 1]$ for the $j$th dimension, $N_{\text{Teacher}}$ and $N_{\text{Mean}}$ are the existing neighborhood best solution and the neighborhood mean solution of each learner, and TF is a teaching factor which can be either 1 or 2 randomly.

In the BBTLBO, there is a $(u \ast 100)$% chance that the $j$th dimension of the $i$th learner in the population follows the behavior of the learning strategy of teacher phase, while the remaining $(100 - u \ast 100)$% follow the search behavior of the Gaussian sampling in teacher phase. This will be helpful to balance the advantages of fast convergence rate (the attraction of the learning strategy of teacher phase) and exploration (the Gaussian sampling) in BBTLBO.

4.3. Learner Phase. At the same time, in the learner phase, a learner interacts randomly with other learners for enhancing his or her knowledge in the class. This learning method can be treated as the global search strategy (shown in (3)).

In this paper, we introduce a new learning strategy in which each learner learns from the neighborhood teacher and the other learner selected randomly of his or her corresponding neighborhood in learner phase. This learning method can be treated as the neighborhood search strategy. Let $\text{new}X_i$ represent the interactive learning result of the learner $X_i$. This neighborhood search strategy can be expressed as follows:

$$\text{new}X_{i,j} = X_{i,j} + r_1 \cdot \left( N_{\text{Teacher},i,j} - X_{i,j} \right) + r_2 \cdot \left( X_{i,j} - X_{k,j} \right),$$

where $r_1$ and $r_2$ are random vectors in which each element is a random number in the range $[0, 1]$, $N_{\text{Teacher}}$ is the teacher of the learner $X_i$’s corresponding neighborhood, and the learner $X_k$ is selected randomly from the learner’s corresponding neighborhood.

In BBTLBO, each learner is probabilistically learning by means of the global search strategy or the neighborhood search strategy in learner phase. That is, about 50% of learners in the population execute the learning strategy of learner phase in the standard TLBO (shown in (3)), while the remaining 50% execute neighborhood search strategy (shown in (12)). This will be helpful to balance the global search and local search in learner phase.

Moreover, compared to the original TLBO, BBTLBO only modifies the learning strategies. Therefore, both the original TLBO and BBTLBO have the same time complexity $O(NP \cdot D \cdot \text{Gen}_{\text{max}})$, where NP is the number of the population, $D$ is the number of dimensions, and $\text{Gen}_{\text{max}}$ is the maximum number of generations.

As explained above, the pseudocode for the implementation of BBTLBO is summarized in Algorithm 2.

5. Functions Optimization

In this section, to illustrate the effectiveness of the proposed method, 20 benchmark functions are used to test the efficiency of BBTLBO. To compare the search performance of BBTLBO with some other methods, other different algorithms are also simulated in the paper.

5.1. Benchmark Functions. The details of 20 benchmark functions are shown in Table 1. Among 20 benchmark functions, $F_1$ to $F_5$ are unimodal functions, and $F_{10}$ to $F_{20}$ are multimodal functions. The searching range and theory optima for all functions are also shown in Table 1.

5.2. Parameter Settings. All the experiments are carried out on the same machine with a Celoron 2.26 GHz CPU, 2 GB memory, and Windows XP operating system with Matlab 7.9.
(1) Begin
(2) Initialize \( N \) (number of learners), \( D \) (number of dimensions) and hybridization factor \( u \)
(3) Initialize learners \( X \) and evaluate all learners \( X \)
(4) while (stopping condition not met)
(5) for each learner \( X_i \) of the class % Teaching phase
(6) \( TF = \text{round}(1 + \text{rand}(0,1)) \)
(7) Donate the \( N \) Teacher and the \( N \) Mean in its neighborhood for each learner
(8) Updating each learner according (11)
(9) Accept new\( X_i \) if \( f(\text{new}X_i) \) is better than \( f(X_i) \)
(10) endfor
(11) for each learner \( X_i \) of the class % Learning phase
(12) Randomly select one learner \( X_k \), such that \( i \neq k \)
(13) if \( \text{rand}(0,1) < 0.5 \)
(14) Updating each learner according (3)
(15) else
(16) Donate the \( N \) Teacher in its neighborhood for each learner
(17) Updating each learner according (12)
(18) endif
(19) Accept new\( X_i \) if \( f(\text{new}X_i) \) is better than \( f(X_i) \)
(20) endfor
(21) endwhile
(22) end

Algorithm 2: BBTLBO().

For the purpose of reducing statistical errors, each algorithm is independently simulated 50 times. For all algorithms, the population size was set to 20. Population-based stochastic algorithms use the same stopping criterion, that is, reaching a certain number of function evaluations (FEs).

5.3. Effect of Variation in Parameter \( u \). The hybridization factor \( u \) is set to \( \{0.0, 0.1, 0.3, 0.5, 0.7, 0.9, 1.0\} \). Comparative tests have been performed using different \( u \). In our experiment, the maximal FEs are used as ended condition of algorithm, namely, 40,000 for all test functions. Table 2 shows the mean optimum solutions and the standard deviation of the solutions obtained using different hybridization factor \( u \) in the 50 independent runs. The best results among the algorithms are shown in bold. Figure 3 presents the representative convergence graphs of different benchmark functions in terms of the mean fitness values achieved by using different hybridization factor \( u \) on all test functions. Due to the tight space limitation, some sample graphs are illustrated.

The comparisons in Table 2 and Figure 3 show that when the hybridization factor \( u \) is set to 0.9, BBTLBO offers the best performance on 20 test functions. Hence, the hybridization factor \( u \) is set to 0.9 in the following experiments.

5.4. Comparison of BBTLBO with Some Similar Bare-Bones Algorithms. In this section, we compare BBTLBO with five other recently proposed three bare-bones DE variants and two bare-bones PSO algorithms. Our experiment includes two series of comparisons in terms of the solution accuracy and the solution convergence (convergence speed and success rate). We compared the performance of BBTLBO with other similar bare-bones algorithms, including BBPSO [20], BBExp [20], BBDE [21], GBDE [22], and MGBDE [22].

5.4.1. Comparisons on the Solution Accuracy. In our experiment, the maximal FEs are used as ended condition of algorithm, namely, 40,000 for all test functions. The results are shown in Table 3 in terms of the mean optimum solution and the standard deviation of the solutions obtained in the 50 independent runs by each algorithm on 20 test functions. The best results among the algorithms are shown in bold. Figure 4 presents the convergence graphs of different benchmark functions in terms of the mean fitness values achieved by 7 algorithms for 50 independent runs. Due to the tight space limitation, some sample graphs are illustrated.

From Table 3 it can be observed that the mean optimum solution and the standard deviation of all algorithms perform well for the functions \( F_{15} \) and \( F_{17} \). Although BBExp performs better than BBTLBO on function \( F_9 \) and MGBDE performs better than BBTLBO on function \( F_{20} \), our approach BBTLBO achieves better results than other algorithms on the rest of test functions. Table 3 and Figure 4 conclude that the BBTLBO has a good performance of the solution accuracy for test functions in this paper.

5.4.2. Comparison of the Convergence Speed and SR. In order to compare the convergence speed and successful rate (SR) of different algorithms, we select a threshold value of the objective function for each test function. For other functions, the threshold values are listed in Table 4. In our experiment, the stopping criterion is that each algorithm is terminated when the best fitness value so far is below the predefined threshold value (\( T \) Value) or the number of FEs reaches to
the maximal FEs 40,000. The results are shown in Table 4 in terms of the mean number of FEs (MFEs) required to converge to the threshold and successful rate (SR) in the 50 independent runs. "NaN" represents that no runs of the corresponding algorithm converged below the predefined threshold before meeting the maximum number of FEs. The best results among the six algorithms are shown in boldface.

From Table 5 it can be observed that all algorithms hardly converge to the threshold for unimodal functions $F_1$, $F_2$, $F_3$, and $F_4$ and multimodal functions $F_{11}$, $F_{12}$, and $F_{14}$. BBTLBO converges to the threshold except for functions $F_3$, $F_9$, and $F_{14}$. From the results of total average FEs, BBTLBO converges faster than other algorithms on all unimodal functions and the majority of multimodal functions except for functions $F_{15}$, $F_{16}$, $F_{19}$, and $F_{20}$. The acceleration rates between BBTLBO and other algorithms are mostly 10 for functions $F_1$, $F_2$, $F_4$, $F_7$, $F_9$, $F_{10}$, and $F_{13}$. From the results of total average SR, BBTLBO achieves the highest SR for those test functions of which BBTLBO successfully converges to the threshold value. It can be concluded that the BBTLBO has a good performance of convergence speed and successful rate (SR) of the solutions for test functions in this paper.

### Table 1: Details of numerical benchmarks used.

| Function | Formula | $D$ | Range | Optima |
|----------|---------|-----|-------|--------|
| Sphere $F_1(x) = \sum_{i=1}^{D} x_i^2$ | 30 | $[-100, 100]$ | 0 |
| Sum square $F_2(x) = \sum_{i=1}^{D} x_i^2$ | 30 | $[-100, 100]$ | 0 |
| Quadric $F_3(x) = \sum_{i=1}^{D} x_i^2 + \text{random}(0, 1)$ | 30 | $[-1.28, 1.28]$ | 0 |
| Step $F_4(x) = \sum_{i=1}^{D} (|x_i + 0.5|)^2$ | 30 | $[-100, 100]$ | 0 |
| Schwefel 1.2 $F_5(x) = \sum_{i=1}^{D} (\sqrt{i} x_i)^2$ | 30 | $[-100, 100]$ | 0 |
| Schwefel 2.21 $F_6(x) = \max \{ |x_i|, 1 \leq i \leq D \}$ | 30 | $[-100, 100]$ | 0 |
| Schwefel 2.22 $F_7(x) = \sum_{i=1}^{D} |x_i| + \sum_{i=1}^{D} |x_i|$ | 30 | $[-10, 10]$ | 0 |
| Zakharov $F_8(x) = \sum_{i=1}^{D} (x_i^2 - x_i + 1)^2 + (x_i - 1)^2$ | 30 | $[-100, 100]$ | 0 |
| Rosenbrock $F_9(x) = \sum_{i=1}^{D} \left[ 100(x_i^2 - x_{i+1})^2 + (x_i - 1)^2 \right]$ | 30 | $[-2.048, 2.048]$ | 0 |
| Ackley $F_{10}(x) = 20 - 20 \exp \left( -\frac{1}{5} \sqrt{\frac{1}{D} \sum_{i=1}^{D} x_i^2} \right) - \exp \left( \frac{1}{D} \sum_{i=1}^{D} \cos \left( 2\pi x_i \right) \right) + e$ | 30 | $[-32, 32]$ | 0 |
| Rastrigin $F_{11}(x) = \sum_{i=1}^{D} (x_i^2 - 10 \cos (2\pi x_i) + 10)$ | 30 | $[-5.12, 5.12]$ | 0 |
| Weierstrass $F_{12}(x) = \sum_{i=1}^{D} \left[ 0.5 \cos \left( 2\pi b \frac{(x_i + 0.5)}{2} \right) \right]$ | 30 | $[-0.5, 0.5]$ | 0 |
| $a = 0.5 \ b = 3 \ k_{\text{max}} = 20$ |
| Griewank $F_{13}(x) = \sum_{i=1}^{D} \left( \frac{x_i^2}{4000} - \cos \left( \frac{x_i}{\sqrt{1}} \right) + 1 \right)$ | 30 | $[-600, 600]$ | 0 |
| Schwefel $F_{14}(x) = 418.9829 D + \sum_{i=1}^{D} (-x_i \sin \sqrt{|x_i|})$ | 30 | $[-500, 500]$ | 0 |
| Bohachevsky1 $F_{15}(x) = x_1^2 + 2x_2^2 - 0.3 \cos \left( 3\pi x_1 \right) - 0.4 \cos \left( 4\pi x_2 \right) + 0.7$ | 2 | $[-100, 100]$ | 0 |
| Bohachevsky2 $F_{16}(x) = x_1^2 + 2x_2^2 - 0.3 \cos \left( 3\pi x_1 \right) + \cos \left( 4\pi x_2 \right) + 0.3$ | 2 | $[-100, 100]$ | 0 |
| Bohachevsky3 $F_{17}(x) = x_1^2 + 2x_2^2 - 0.3 \cos \left( 3\pi x_1 \right) + \cos \left( 4\pi x_2 \right)$ | 2 | $[-100, 100]$ | 0 |
| Shkel5 $F_{18}(x) = -\sum_{i=1}^{5} \left( x_i - a_i \right)^2 + c_i$ | 4 | $[0, 10]$ | $-10.1532$ |
| Shkel7 $F_{19}(x) = -\sum_{i=1}^{7} \left( x_i - a_i \right)^2 + c_i$ | 4 | $[0, 10]$ | $-10.4029$ |
| Shkel10 $F_{20}(x) = -\sum_{i=1}^{10} \left( x_i - a_i \right)^2 + c_i$ | 4 | $[0, 10]$ | $-10.5364$ |
Table 2: Comparisons mean ± std of the solutions using different $u$.

| Run | BBTLO $(u = 0.0)$ | BBTLO $(u = 0.1)$ | BBTLO $(u = 0.3)$ | BBTLO $(u = 0.5)$ | BBTLO $(u = 0.7)$ | BBTLO $(u = 0.9)$ | BBTLO $(u = 1.0)$ |
|-----|------------------|------------------|------------------|------------------|------------------|------------------|------------------|
| $F_1$ | $1.75e-001 ± 1.21e+000$ | $6.89e-071 ± 1.01e+000$ | $1.2e-163 ± 000$ | $1.21e-256 ± 000$ | $0.0 ± 0.0$ | $0.0 ± 0.0$ | $0.0 ± 0.0$ |
| $F_2$ | $8.98e-005 ± 5.73e+004$ | $5.62e-069 ± 2.72e+068$ | $2.20e-161 ± 1.12e+160$ | $2.43e-254 ± 000$ | $0.0 ± 0.0$ | $0.0 ± 0.0$ | $0.0 ± 0.0$ |
| $F_3$ | $1.20e-001 ± 6.34e+002$ | $5.91e-003 ± 1.44e+003$ | $1.01e-003 ± 3.48e+004$ | $4.35e-004 ± 1.97e+004$ | $2.35e-004 ± 1.30e-004$ | $2.27e-004 ± 1.26e-004$ | $1.99e-004 ± 1.13e-004$ |
| $F_4$ | $7.65e+002 ± 5.83e+002$ | $4.80e-001 ± 8.86e-001$ | $0.0 ± 0.0$ | $0.0 ± 0.0$ | $0.0 ± 0.0$ | $0.0 ± 0.0$ | $0.0 ± 0.0$ |
| $F_5$ | $5.58e+002 ± 6.58e+002$ | $1.87e-028 ± 5.73e-028$ | $3.53e-054 ± 1.86e-053$ | $3.69e-073 ± 6.74e-072$ | $9.53e-096 ± 6.10e-095$ | $9.16e-110 ± 434e-099$ | $2.56e-100 ± 1.30e-099$ |
| $F_6$ | $2.51e+001 ± 5.34e+000$ | $6.67e-021 ± 8.81e-021$ | $2.81e-061 ± 6.35e-061$ | $8.22e-100 ± 1.80e-099$ | $8.18e-137 ± 1.41e-136$ | $3.63e-154 ± 1.34e-153$ | $8.86e-147 ± 5.22e-146$ |
| $F_7$ | $1.37e-003 ± 9.54e-003$ | $8.72e-043 ± 1.52e-042$ | $5.68e-088 ± 8.76e-088$ | $1.01e-133 ± 2.38e-133$ | $2.60e-175 ± 000$ | $1.16e-188 ± 000$ | $8.33e-190 ± 000$ |
| $F_8$ | $2.41e+000 ± 3.07e+000$ | $1.32e-019 ± 2.98e-019$ | $2.13e-028 ± 7.69e-028$ | $3.44e-037 ± 1.24e-036$ | $2.20e-050 ± 9.12e-050$ | $1.07e-056 ± 4.39e-056$ | $2.03e-049 ± 8.94e-049$ |
| $F_9$ | $2.66e+001 ± 1.79e+000$ | $2.72e+001 ± 3.17e+000$ | $2.77e+001 ± 3.18e+000$ | $2.83e+001 ± 2.78e+000$ | $2.84e+001 ± 2.67e+000$ | $2.85e+001 ± 3.41e+000$ | $2.80e+001 ± 3.87e+000$ |
| $F_{10}$ | $8.30e+000 ± 1.76e+000$ | $1.77e+001 ± 6.10e+000$ | $5.90e-015 ± 1.70e-015$ | $3.55e-015 ± 000$ | $3.55e-015 ± 000$ | $3.55e-015 ± 000$ | $3.55e-015 ± 000$ |
| $F_{11}$ | $3.74e+001 ± 9.05e+000$ | $3.33e+001 ± 1.18e+001$ | $2.71e+001 ± 8.04e+000$ | $1.89e+001 ± 1.14e+001$ | $5.73e+000 ± 1.06e+001$ | $0.0 ± 0.0$ | $0.0 ± 0.0$ |
| $F_{12}$ | $8.15e+000 ± 1.93e+000$ | $3.38e-001 ± 1.16e+000$ | $0.0 ± 0.0$ | $0.0 ± 0.0$ | $0.0 ± 0.0$ | $0.0 ± 0.0$ | $0.0 ± 0.0$ |
| $F_{13}$ | $5.06e-001 ± 8.08e-001$ | $6.52e-003 ± 8.86e-003$ | $1.78e-003 ± 3.68e-003$ | $0.0 ± 0.0$ | $0.0 ± 0.0$ | $0.0 ± 0.0$ | $0.0 ± 0.0$ |
| $F_{14}$ | $4.33e+003 ± 6.79e+002$ | $4.67e+003 ± 6.10e+002$ | $5.17e+003 ± 6.68e+002$ | $5.59e+003 ± 6.85e+002$ | $5.53e+003 ± 7.30e+002$ | $5.53e+003 ± 7.80e+002$ | $5.40e+003 ± 6.53e+002$ |
| $F_{15}$ | $0.0 ± 0.0$ | $0.0 ± 0.0$ | $0.0 ± 0.0$ | $0.0 ± 0.0$ | $0.0 ± 0.0$ | $0.0 ± 0.0$ | $0.0 ± 0.0$ |
| $F_{16}$ | $0.0 ± 0.0$ | $0.0 ± 0.0$ | $0.0 ± 0.0$ | $0.0 ± 0.0$ | $0.0 ± 0.0$ | $0.0 ± 0.0$ | $0.0 ± 0.0$ |
| $F_{17}$ | $0.0 ± 0.0$ | $0.0 ± 0.0$ | $0.0 ± 0.0$ | $0.0 ± 0.0$ | $0.0 ± 0.0$ | $0.0 ± 0.0$ | $0.0 ± 0.0$ |
| $F_{18}$ | $-7.71e+000 ± 3.47e+000$ | $-8.06e+000 ± 3.39e+000$ | $-9.64e+000 ± 1.81e+000$ | $-9.65e+000 ± 1.76e+000$ | $-1.02e+001 ± 6.77e-003$ | $-9.85e+000 ± 1.22e+000$ | $-9.93e+000 ± 1.12e+000$ |
| $F_{19}$ | $-7.69e+000 ± 3.52e+000$ | $-8.13e+000 ± 3.36e+000$ | $-9.87e+000 ± 1.83e+000$ | $-1.03e+001 ± 9.45e-001$ | $-9.76e+000 ± 1.95e+000$ | $-9.82e+000 ± 1.78e+000$ | $-9.61e+000 ± 1.99e+000$ |
| $F_{20}$ | $-8.12e+000 ± 3.53e+000$ | $-9.38e+000 ± 2.69e+000$ | $-1.01e+001 ± 1.65e+000$ | $-1.01e+001 ± 1.61e+000$ | $-9.70e+000 ± 2.28e+000$ | $-9.44e+000 ± 2.43e+000$ | $-1.00e+001 ± 1.69e+000$ |
Table 3: Comparisons mean ± std of the solutions using different algorithms.

| Fun | BBPSO         | BBExp         | BDE | GBDE            | MGBDE            | BRTLBO            |
|-----|---------------|---------------|-----|-----------------|-----------------|------------------|
| F1  | 5.44e-027 ± 1.87e-026 | 2.62e-024 ± 5.00e-024 | 3.90e-035 ± 2.00e-034 | 4.35e-022 ± 1.13e-021 | 3.35e-035 ± 2.11e-034 | 0.0 ± 0.0          |
| F2  | 13800 ± 2.11e+004 | 1000 ± 4.63e+003 | 6.20e-021 ± 4.38e-020 | 1400 ± 4.52e+003 | 1.28e-032 ± 8.37e-032 | 0.0 ± 0.0          |
| F3  | 1.32e+000 ± 3.18e+000 | 2.22e-002 ± 7.55e-003 | 1.64e-002 ± 9.57e-003 | 2.49e-002 ± 9.88e-003 | 1.16e-002 ± 5.26e-003 | 2.27e-004 ± 1.26e-004 |
| F4  | 5.60e+000 ± 9.28e+000 | 9.60e-001 ± 4.27e+000 | 7.89e+001 ± 3.05e+002 | 8.40e-001 ± 9.12e-001 | 1.08e+000 ± 1.28e+000 | 0.0 ± 0.0          |
| F5  | 1.24e+004 ± 6.66e+003 | 4.41e+003 ± 3.37e+003 | 2.09e+000 ± 4.00e+000 | 5.36e+003 ± 3.26e+003 | 7.57e+002 ± 1.16e+003 | 2.16e-115 ± 1.10e-114 |
| F6  | 1.67e+001 ± 9.19e+000 | 1.20e+000 ± 5.22e-001 | 1.39e+001 ± 4.47e+000 | 3.60e-001 ± 1.95e-001 | 1.10e+000 ± 2.94e+000 | 3.63e-154 ± 1.34e-153 |
| F7  | 2.34e+001 ± 1.32e+001 | 1.00e+000 ± 3.03e+000 | 4.06e-019 ± 2.15e-018 | 6.00e-001 ± 2.40e+000 | 2.00e-001 ± 1.41e+000 | 1.16e-188 ± 0.00    |
| F8  | 1.87e+002 ± 1.34e+002 | 1.58e+002 ± 7.00e+001 | 1.16e-001 ± 2.35e-001 | 1.72e+002 ± 6.67e+001 | 2.49e+001 ± 1.99e+001 | 1.07e-056 ± 4.39e-056 |
| F9  | 7.07e+001 ± 1.48e+002 | 3.57e-001 ± 2.50e+001 | 2.76e+001 ± 1.06e+001 | 3.17e+001 ± 2.07e+001 | 2.76e+001 ± 1.46e+001 | 2.83e+001 ± 3.41e+001 |
| F10 | 10.6e+001 ± 9.29e+000 | 1.52e+000 ± 5.11e+000 | 1.34e+000 ± 1.15e+000 | 2.59e+000 ± 6.45e+000 | 5.54e-001 ± 2.79e+000 | 3.55e-015 ± 0.00    |
| F11 | 1.16e+002 ± 3.53e+000 | 1.81e+001 ± 7.28e+000 | 6.76e+001 ± 3.89e+001 | 1.55e+001 ± 5.96e+000 | 2.03e+001 ± 9.23e+000 | 0.0 ± 0.0          |
| F12 | 2.73e+000 ± 2.11e+000 | 1.20e-001 ± 4.42e-001 | 1.73e+000 ± 1.32e+000 | 1.21e-001 ± 3.37e-001 | 5.17e-001 ± 8.67e-001 | 0.0 ± 0.0          |
| F13 | 2.14e+002 ± 4.11e-002 | 2.34e+002 ± 4.29e+003 | 4.07e-002 ± 4.89e+002 | 3.08e-003 ± 7.42e+003 | 4.63e-003 ± 7.16e+003 | 0.0 ± 0.0          |
| F14 | 3.64e+003 ± 6.28e+002 | 2.58e+003 ± 5.51e+002 | 2.30e+003 ± 4.09e+002 | 2.49e+003 ± 5.41e+002 | 2.60e+003 ± 5.05e+002 | 5.58e+003 ± 7.80e+002 |
| F15 | 0.0 ± 0.0 | 0.0 ± 0.0 | 0.0 ± 0.0 | 0.0 ± 0.0 | 0.0 ± 0.0 | 0.0 ± 0.0          |
| F16 | 4.37e-003 ± 3.09e-002 | 0.0 ± 0.0 | 0.0 ± 0.0 | 0.0 ± 0.0 | 0.0 ± 0.0 | 0.0 ± 0.0          |
| F17 | 0.0 ± 0.0 | 0.0 ± 0.0 | 0.0 ± 0.0 | 0.0 ± 0.0 | 0.0 ± 0.0 | 0.0 ± 0.0          |
| F18 | -5.60e+000 ± 3.41e+000 | -7.90e+000 ± 2.74e+000 | -7.09e+000 ± 3.33e+000 | -7.63e+000 ± 2.86e+000 | -8.01e+000 ± 3.00e+000 | -9.85e+000 ± 1.22e+000 |
| F19 | -5.97e+000 ± 3.31e+000 | -7.87e+000 ± 3.03e+000 | -6.21e+000 ± 3.66e+000 | -8.60e+000 ± 2.68e+000 | -8.37e+000 ± 2.90e+000 | -9.82e+000 ± 1.78e+000 |
| F20 | -5.81e+000 ± 3.65e+000 | -9.40e+000 ± 2.42e+000 | -6.02e+000 ± 3.77e+000 | -9.46e+000 ± 2.24e+000 | -9.38e+000 ± 2.51e+000 | -9.41e+000 ± 2.43e+000 |
5.5. Comparison of BBTLBO with DE Variants, PSO Variants, and Some TLBO Variants. In this section, we compared the performance of BBTLBO with other optimization algorithms, including jDE [28], SaDE [29], PSOcFLocal [27], PSOWFIPS [30], and TLBO [8, 9]. In our experiment, the maximal FEs are used as the stopping criterion of all algorithms, namely, 40,000 for all test functions. The results are shown in Table 5 in terms of the mean optimum solution and the standard deviation of the solutions obtained in the 50 independent runs by each algorithm on 20 test functions, where “w/t/l” summarizes the competition results among BBTLBO and other algorithms. The best results among the algorithms are shown in boldface.

The comparisons in Table 5 show that all algorithms perform well for $F_{15}$, $F_{16}$, and $F_{17}$. Although SaDE outperforms BBTLBO on $F_{14}$, PSOcFLocal outperforms BBTLBO on $F_8$ and PSOWFIPS outperforms BBTLBO on $F_{19}$ and $F_{20}$, and BBTLBO offers the highest accuracy on functions $F_3$, $F_4$, $F_5$, $F_7$, $F_9$, $F_{10}$, $F_{11}$, and $F_{18}$. “w/t/l” shows that BBTLBO offers well accuracy for the majority of test functions in this paper.

Figure 3: Comparison of the performance curves using different $u$. 

(a) $F_7$ Schwefel 2.22

(b) $F_8$ Zakharov

(c) $F_{18}$ Shekel5

(d) $F_{11}$ Rastrigin
Table 5 concludes that BBTLBO has a good performance of the solution accuracy for all unimodal optimization problems and most complex multimodal optimization problems.

6. Two Real-World Optimization Problems

In this section, to show the effectiveness of the proposed method, the proposed BBTLBO algorithm is applied to estimate parameters of two real-world problems.

6.1. Nonlinear Function Approximation. The artificial neural network trained by our BBTLBO algorithm is a three-layer feed-forward network and the basic structure of the proposed scheme is depicted in Figure 5. The inputs are connected to all the hidden units, which in turn all connected to all
Table 4: The mean number of FEs and SR with acceptable solutions using different algorithms.

| Fun | t value | BBPSO MFEs | BBExp MFEs | BBDE MFEs | GBDE MFEs | MGBDE MFEs | BBTLBO MFEs |
|-----|---------|------------|------------|-----------|-----------|------------|-------------|
|     |         | SR         | SR         | SR        | SR        | SR         | SR          |
| $F_1$ | $1E−8$ | 15922 100 | 17727 100 | 11042 100 | 19214 100 | 11440 100 | 1390 100    |
| $F_2$ | $1E−8$ | 17515 54  | 19179 94  | 12243 100 | 20592 90  | 12634 100 | 1500 100    |
| $F_3$ | $1E−8$ | NaN 0 NaN | NaN 0 NaN | NaN NaN 0 | NaN NaN 0 | NaN NaN 0 | NaN NaN     |
| $F_4$ | $1E−8$ | 11710 24 | 8120 84   | 3634 6    | 7343 40   | 4704 34   | 525 100     |
| $F_5$ | $1E−8$ | NaN 0 NaN | NaN 0 NaN | NaN NaN 0 | NaN NaN 0 | NaN NaN 0 | NaN NaN     |
| $F_6$ | $1E−8$ | 17540 6 | 21191 90 | 17314 100 | 22684 94 | 15322 98 | 2144 100    |
| $F_7$ | $1E−8$ | NaN 0 NaN | NaN 0 NaN | NaN NaN 0 | NaN NaN 0 | NaN NaN 0 | NaN NaN     |
| $F_8$ | $1E−8$ | NaN 0 NaN | NaN 0 NaN | NaN NaN 0 | NaN NaN 0 | NaN NaN 0 | NaN NaN     |
| $F_9$ | $1E−2$ | 24647 26 | 27598 90 | 18273 26 | 18182 52 | 17200 80 | NaN         |
| $F_{10}$ | $1E−8$ | 17073 62 | 18404 42 | 1115 40 | 6548 76 | 5441 80 | 2822 82    |
| $F_{11}$ | $1E−8$ | NaN 0 NaN | NaN 0 NaN | NaN NaN 0 | NaN NaN 0 | NaN NaN 0 | NaN NaN     |
| $F_{12}$ | $1E−8$ | 1190 36 | 8928 74 | 11048 16 | 22951 64 | 14786 58 | 1470 100    |
| $F_{13}$ | $1E−8$ | 568 100 | 1176 100 | 1254 100 | 1294 100 | 1343 100 | 1308 100    |
| $F_{14}$ | $1E−8$ | 657 98 | 1251 100 | 1274 100 | 1251 100 | 1206 100 | 799 100     |
| $F_{15}$ | $1E−8$ | 995 100 | 2626 100 | 1487 100 | 2759 100 | 1921 100 | 973 100     |
| $F_{16}$ | $1E−8$ | 15922 100 | 17727 100 | 11042 100 | 19214 100 | 11440 100 | 1390 100    |
| $F_{17}$ | $1E−8$ | 17515 54 | 19179 94 | 12243 100 | 20592 90 | 12634 100 | 1500 100    |
| $F_{18}$ | $1E−8$ | NaN 0 NaN | NaN 0 NaN | NaN NaN 0 | NaN NaN 0 | NaN NaN 0 | NaN NaN     |
| $F_{19}$ | $1E−8$ | NaN 0 NaN | NaN 0 NaN | NaN NaN 0 | NaN NaN 0 | NaN NaN 0 | NaN NaN     |
| $F_{20}$ | $1E−8$ | NaN 0 NaN | NaN 0 NaN | NaN NaN 0 | NaN NaN 0 | NaN NaN 0 | NaN NaN     |

For neural network training, the aim is to find a set of weights with the smallest error measure. Here the objective function is the mean sum of squared errors (MSE) over all training patterns which is shown as follows:

$$L = (M + 1) \times N + (N + 1) \times K.$$  \hfill (13)

where $L$ is the number of all the neurons in the network. $M$ is the number of input units, $N$ is the number of hidden units, and $K$ is the number of output units.

The number (dimension) of the variables is shown as follows:

$$L = (M + 1) \times N + (N + 1) \times K.$$  \hfill (13)

For neural network training, the aim is to find a set of weights with the smallest error measure. Here the objective function is the mean sum of squared errors (MSE) over all training patterns which is shown as follows:

$$\text{MSE} = \frac{1}{Q \times K} \sum_{i=1}^{Q} \sum_{j=1}^{K} \frac{1}{2}(d_{ij} - y_{ij})^2,$$  \hfill (14)

where $Q$ is the number of training data set, $K$ is the number of output units, $d_{ij}$ is desired output, and $y_{ij}$ is output inferred from neural network.

In this example, a three-layer feed-forward ANN with one input unit, five hidden units, and one output unit is constructed to model the curve of a nonlinear function which is described by the following equation [31]:

$$y = \sin(2x) \exp(-2x).$$  \hfill (15)

In this case, activation function used in the output layer is the sigma function and activation function used in the output layer is linear. The number (dimension) of the variables is 16 for BBTLBO-based ANN. In order to train the ANN, 200 pairs of data are chosen from the real model. For each algorithm, 50 runs are performed. The other parameters are the same as those of the previous investigations. The results are shown in Table 6 in terms of the mean MSE and the standard deviation obtained in the 50 independent runs for three methods. Figure 6 shows the predicted time series for training and test using different algorithms. It can conclude that the approximation achieved by BBTLBO has good performance.

6.2. Tuning of PID Controller. The continuous form of a discrete-type PID controller with a small sampling period $\Delta t$ is described as follows [32]:

$$u[k] = K_P \cdot e[k] + K_I \cdot \sum_{i=1}^{k} e[i] \cdot \Delta t + K_D \cdot \frac{e[k] - e[k-1]}{\Delta t},$$  \hfill (16)

where $u[k]$ is the controlled output, respectively. $e[k] = r[k] - y[k]$ is the error signal, $r[k]$ and $y[k]$ are the reference signal and the system output, and $K_P$, $K_I$, and $K_D$ represent the proportional, integral and derivative gains, respectively.

For an unknown plant, the goal of this problem is to minimize the integral absolute error (IAE), which is given as follow [32, 33]:

$$f(t) = \int_{0}^{\infty} \left(\omega_1 |e(t)| + \omega_2 u^2(t)\right)dt + \omega_3 t_r,$$  \hfill (17)
Table 5: Comparisons mean ± std of the solutions using different algorithms.

| Fun | jDE       | SaDE      | PSOcLocal | PSOwFIPS | TLBO | BRTLBO |
|-----|-----------|-----------|-----------|----------|------|--------|
| 1   | 3.63e−025 ± 1.85e−024 | 7.65e−025 ± 3.34e−024 | 9.23e−018 ± 3.03e−017 | 1.01e−002 ± 5.48e−003 | 3.05e−189 ± 0.00 | 0.0 ± 0.0 |
| 2   | 1.49e−023 ± 6.69e−023 | 2.75e−025 ± 1.08e−024 | 3.68e−017 ± 5.37e−017 | 1.08e−001 ± 5.05e−002 | 1.29e−185 ± 0.00 | 0.0 ± 0.0 |
| 3   | 3.22e−002 ± 2.83e−002 | 2.08e−002 ± 1.18e−002 | 1.28e−002 ± 5.50e−003 | 1.86e−002 ± 4.39e−003 | 5.70e−004 ± 3.27e−004 | 2.27e−004 ± 1.26e−004 |
| 4   | 2.11e + 001 ± 6.74e + 001 | 0.0 ± 0.0 | 0.0 ± 0.0 | 0.0 ± 0.0 | 0.0 ± 0.0 | 0.0 ± 0.0 |
| 5   | 1.22e + 002 ± 1.37e + 002 | 4.28e + 001 ± 2.59e + 001 | 1.17e + 001 ± 9.30e + 000 | 2.60e + 003 ± 6.79e + 002 | 9.45e−043 ± 6.47e−042 | 2.16e−115 ± 1.10e−114 |
| 6   | 3.06e + 001 ± 8.50e + 000 | 2.45e + 000 ± 2.60e + 000 | 4.67e−001 ± 2.82e−001 | 2.66e + 000 ± 5.58e −001 | 2.08e−078 ± 4.30e−078 | 3.63e−154 ± 1.34e−153 |
| 7   | 8.28e−019 ± 3.49e−018 | 5.40e−016 ± 3.81e−015 | 1.34e−011 ± 1.27e−011 | 1.70e−002 ± 2.85e−003 | 3.84e−096 ± 5.53e−096 | 1.16e−188 ± 0.00 |
| 8   | 2.16e + 000 ± 4.16e + 000 | 4.88e−001 ± 5.82e−001 | 9.60e−002 ± 6.99e−002 | 5.86e−001 ± 1.70e+001 | 7.09e−022 ± 4.99e−021 | 1.07e−056 ± 4.39e−056 |
| 9   | 2.49e + 001 ± 1.05e + 001 | 2.61e + 001 ± 1.07e + 000 | 2.40e + 001 ± 1.52e + 000 | 2.65e + 001 ± 3.54e−001 | 2.55e + 001 ± 5.01e−001 | 2.83e + 001 ± 3.41e−001 |
| 10  | 5.05e−002 ± 1.05e−001 | 2.07e−001 ± 4.58e−001 | 1.94e−001 ± 4.56e−001 | 2.16e−002 ± 4.37e−003 | 3.62e−015 ± 5.02e−016 | 3.55e−015 ± 0.00 |
| 11  | 2.03e + 000 ± 1.94e + 000 | 3.86e+000 ± 1.97e+000 | 4.26e + 001 ± 1.06e + 001 | 1.15e + 002 ± 1.54e+001 | 1.55e + 001 ± 8.09e+000 | 0.0 ± 0.0 |
| 12  | 2.88e−002 ± 1.45e−001 | 6.50e−002 ± 1.87e−001 | 7.89e−002 ± 1.03e−000 | 1.36e + 000 ± 7.41e−001 | 0.0 ± 0.0 | 0.0 ± 0.0 |
| 13  | 1.87e−002 ± 3.58e−002 | 1.18e−002 ± 1.75e−002 | 1.16e−002 ± 1.58e−002 | 1.06e−001 ± 9.93e−002 | 0.0 ± 0.0 | 0.0 ± 0.0 |
| 14  | 1.93e + 002 ± 1.42e + 002 | 1.35e + 002 ± 1.26e + 002 | 4.49e + 003 ± 8.25e + 002 | 3.96e + 003 ± 8.40e+002 | 4.82e + 003 ± 6.86e+002 | 5.58e + 003 ± 7.80e+002 |
| 15  | 0.0 ± 0.0 | 0.0 ± 0.0 | 0.0 ± 0.0 | 0.0 ± 0.0 | 0.0 ± 0.0 | 0.0 ± 0.0 |
| 16  | 0.0 ± 0.0 | 0.0 ± 0.0 | 0.0 ± 0.0 | 0.0 ± 0.0 | 0.0 ± 0.0 | 0.0 ± 0.0 |
| 17  | 0.0 ± 0.0 | 0.0 ± 0.0 | 0.0 ± 0.0 | 0.0 ± 0.0 | 0.0 ± 0.0 | 0.0 ± 0.0 |
| 18  | −9.40e + 000 ± 2.10e + 000 | −9.25e + 000 ± 2.30e + 000 | −7.76e + 000 ± 3.42e + 000 | −9.79e + 000 ± 1.44e + 000 | −9.72e + 000 ± 1.42e + 000 | −9.85e + 000 ± 1.22e + 000 |
| 19  | −9.85e + 000 ± 1.90e + 000 | −9.87e + 000 ± 1.83e + 000 | −9.24e + 000 ± 2.70e + 000 | −1.04e + 001 ± 4.23e −009 | −9.22e + 000 ± 2.41e + 000 | −9.82e + 000 ± 1.78e + 000 |
| 20  | −9.65e + 000 ± 2.23e + 000 | −1.01e + 001 ± 1.59e + 000 | −9.63e + 000 ± 2.50e + 000 | −1.05e + 001 ± 1.01e −004 | −9.65e + 000 ± 2.23e + 000 | −9.41e + 000 ± 2.43e + 000 |

w/t/l | 13/3/4 | 12/4/4 | 13/4/3 | 12/4/4 | 11/6/3 | —
Figure 6: Comparison of the performance curves using different algorithms.

(a) Convergence curves

(b) Approximation curves

(c) Error curves

Table 6: Comparisons between BBTLBO and other algorithms on MSE.

| Algorithm | Training error | Testing error |
|-----------|----------------|---------------|
|           | Mean Std       | Mean Std      |
| TLBO      | 9.85e-004 9.26e-004 | 9.43e-004 9.18e-004 |
| BBTLBO    | 3.45e-004 2.02e-004 | 2.76e-004 1.82e-004 |

where \( e(t) \) and \( u(t) \) are used to represent the system error and the control output at time \( t \), \( t_r \) is the rising time, and \( \omega_i (i = 1, 2, 3) \) are weight coefficients.

To avoid overshooting, a penalty value is adopted in the cost function. That is, once overshooting occurs, the value of overshooting is added to the cost function, and the cost function is given as follows [32, 33]:

\[
\begin{align*}
\text{if } dy(t) < 0 & \\
f(t) = \int_0^\infty \left( \omega_1 |e(t)| + \omega_2 u^2(t) + \omega_4 |dy(t)| \right) dt + \omega_3 t_r \\
\text{else} & \\
f(t) = \int_0^\infty \left( \omega_1 |e(t)| + \omega_2 u^2(t) \right) dt + \omega_3 t_r,
\end{align*}
\]
Table 7: Comparisons of parameters of PID controllers using different algorithms.

| Algorithm | $K_p$  | $K_i$  | $K_d$  | Overshoot (%) | Peak time (s) | Rise time (s) | Cost function | CPU time (s) |
|-----------|--------|--------|--------|---------------|---------------|---------------|---------------|--------------|
| GA        | 0.11257| 0.02710| 0.28792| 2.90585       | 1.65000       | 1.05000       | 16.34555      | 7.05900      |
| PSO       | 0.11772| 0.01756| 0.27737| 1.04808       | 1.65000       | 0.65000       | 11.60773      | 6.91000      |
| BBTLBO    | 0.11605| 0.01661| 0.25803| 0.34261       | 1.80000       | 0.70000       | 11.34300      | 7.04500      |

where $\omega_4$ is a coefficient and $\omega_4 \gg \omega_1$, $dy(t) = y(t) - y(t-1)$, and $y(t)$ is the output of the controlled objective.

In our simulation, the formulas for the plant examined are given as follows [34]:

$$G(s) = \frac{1958}{s^3 + 17.89s^2 + 103.3s + 190.8}.$$  \hfill (19)

The system sampling time is $\Delta t = 0.05$ second and the control value $u$ is limited in the range of $[-10, 10]$. Other relevant system variables are $K_p \in [0, 1]$, $K_i \in [0, 1]$, and $K_d \in [0, 1]$. The weight coefficients of the cost function are set as $\omega_1 = 0.999$, $\omega_2 = 0.001$, $\omega_3 = 2$, and $\omega = 100$ in this example.

In the simulations, the step response of PID control system tuned by the proposed BBTLBO is compared with that tuned by the standard genetic algorithm (GA) and the standard PSO (PSO). The population sizes of GA, PSO, and BBTLBO are 50, and the corresponding maximum numbers of iterations are 50, 50, and 50, respectively. In addition, the crossover rate is set as 0.90 and the mutation rate is 0.10 for GA.

The optimal parameters and the corresponding performance values of the PID controllers are listed in Table 7 and the corresponding performance curves and step responses curves are given in Figures 7 and 8. It can be seen from Figure 7 and Table 7 that the PID controller tuned by BBTLBO has the minimum cost function and CPU time.

Although PID controllers tuned by PSO have a smaller peak time and rise time, their maximum overshoots are much larger than the overshoot tuned by BBTLBO. It concludes that the PID controller tuned by the BBTLBO could perform the best control performance in the simulations.

7. Conclusion

In this paper, TLBO has been extended to BBTLBO which uses the hybridization of the learning strategy in the standard TLBO and Gaussian sampling learning to balance the exploration and the exploitation in teacher phase and uses a modified mutation operation so as to eliminate the duplicate learners in learner phase. The proposed BBTLBO algorithm is utilized to optimize 20 benchmark functions and two real-world optimization problems. From the analysis and experiments, the BBTLBO algorithm significantly improves the performance of the original TLBO, although it needs to spend more CPU time than the standard TLBO algorithm in each generation. From the results compared with other algorithms on the 20 chosen test problems, it can be observed that the BBTLBO algorithm has good performance by using neighborhood search more effectively to generate better quality solutions, although the BBTLBO algorithm does not always have the best performance in all experiments cases of this paper. It can be also observed that the BBTLBO algorithm
gives the best performance on two real-world optimization problems compared with other algorithms in the paper.

Further work includes research into neighborhood search based on different topological structures. Moreover, the algorithm may be further applied to constrained, dynamic, and noisy single-objective and multiobjective optimization domain. It is expected that BBTLBO will be used to more 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 research was partially supported by the National Natural Science Foundation of China (61100173, 61100009, 61272283, and 61304082). This work is partially supported by the Natural Science Foundation of Anhui Province, China (Grant no. 1308085MF82), and the Doctoral Innovation Foundation of Xi’an University of Technology (207-002J1305).

**References**

[1] D. E. Goldberg, *Genetic Algorithms in Search Optimization and Machine Learning*, Addison-Wesley, Reading, Mass, USA, 1989.

[2] L. C. Jiao and L. Wang, “A novel genetic algorithm based on immunity,” *IEEE Transactions on Systems, Man, and Cybernetics A: Systems and Humans*, vol. 30, no. 5, pp. 552–561, 2000.

[3] R. Storn and 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.

[4] M. Dorigo and T. Stutzle, *Ant Colony Optimization*, MIT Press, 2004.

[5] J. Kennedy and R. Eberhart, “Particle swarm optimization,” in *Proceedings of the IEEE International Conference on Neural Networks*, pp. 1942–1948, December 1995.

[6] D. Karaboga and B. Basturk, “On the performance of artificial bee colony (ABC) algorithm,” *Applied Soft Computing Journal*, vol. 8, no. 1, pp. 687–697, 2008.

[7] D. Simon, “Biogeography-based optimization,” *IEEE Transactions on Evolutionary Computation*, vol. 12, no. 6, pp. 702–713, 2008.

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

[9] 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.

[10] R. V. Rao, V. J. Savsani, and D. P. Vakharia, “Teaching-learning-based optimization algorithm for unconstrained and constrained real-parameter optimization problems,” *Engineering Optimization*, vol. 44, no. 12, pp. 1447–1462, 2012.

[11] V. Toğan, “Design of planar steel frames using teaching-learning based optimization,” *Engineering Structures*, vol. 34, pp. 225–232, 2012.

[12] 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, pp. 535–560, 2012.

[13] S. O. Degertekin and M. S. Hayalioğlu, “Sizing truss structures using teaching-learning-based optimization,” *Computers and Structures*, vol. 119, pp. 177–188, 2013.

[14] R. V. Rao and V. Patel, “An improved teaching-learning-based optimization algorithm for solving unconstrained optimization problems,” *Scientia Iranica*, vol. 20, no. 3, pp. 710–720, 2013.

[15] R. V. Rao and V. Patel, “Multi-objective optimization of combined Brayton and inverse Brayton cycles using advanced optimization algorithms,” *Engineering Optimization*, vol. 44, no. 8, pp. 965–983, 2011.

[16] T. Niknam, F. Golestaneh, and M. S. Sadeghi, “Theta-multi-objective teaching-learning-based optimization for dynamic economic emission dispatch,” *IEEE Systems Journal*, vol. 6, no. 2, pp. 341–352, 2012.

[17] R. V. Rao and V. Patel, “Multi-objective optimization of heat exchangers using a modified teaching-learning-based optimization algorithm,” *Applied Mathematical Modelling*, vol. 37, no. 3, pp. 1147–1162, 2013.

[18] M. Clerc and J. Kennedy, “The particle swarm-explosion, stability, and convergence in a multidimensional complex space,” *IEEE Transactions on Evolutionary Computation*, vol. 6, no. 1, pp. 58–73, 2002.

[19] F. van den Bergh and A. P. Engelbrecht, “A study of particle swarm optimization particle trajectories,” *Information Sciences*, vol. 176, no. 8, pp. 937–971, 2006.

[20] J. Kennedy, “Bare bones particle swarms,” in *Proceedings of the Swarm Intelligence Symposium (SIS ’03)*, pp. 80–87, 2003.

[21] M. G. H. Omran, A. P. Engelbrecht, and A. Salman, “Bare bones differential evolution,” *European Journal of Operational Research*, vol. 196, no. 1, pp. 128–139, 2009.

[22] H. Wang, S. Rahnamayan, H. Sun, and M. G. H. Omran, “Gaussian bare-bones differential evolution,” *IEEE Transactions on Cybernetics*, vol. 43, no. 2, pp. 634–647, 2013.

[23] X. H. Hu and R. Eberhart, “Multiobjective optimization using dynamic neighborhood particle swarm optimization,” in *Proceedings of the Congress on Evolutionary Computation*, pp. 1677–1681, 2002.

[24] M. G. Omran, A. P. Engelbrecht, and A. Salman, “Using the ring neighborhood topology with self-adaptive differential evolution,” in *Advances in Natural Computation*, pp. 976–979, Springer, Berlin, Germany, 2006.

[25] X. Li, “Niching without niching parameters: particle swarm optimization using a ring topology,” *IEEE Transactions on Evolutionary Computation*, vol. 14, no. 1, pp. 150–169, 2010.

[26] I. Maruta, T. H. Kim, D. Song, and T. Sugie, “Synthesis of fixed-structure robust controllers using a constrained particle swarm optimizer with cyclic neighborhood topology,” *Expert Systems with Applications*, vol. 40, no. 9, pp. 3595–3605, 2013.

[27] J. Kennedy and R. Mendes, “Population structure and particle swarm performance,” in *Proceedings of the International Conference on Evolutionary Computation*, pp. 1671–1676, Honolulu, Hawaii, USA, 2002.

[28] J. Brest, S. Greiner, B. Bošković, M. Meršnik, and V. Zumer, “Self-adapting control parameters in differential evolution: a comparative study on numerical benchmark problems,” *IEEE Transactions on Evolutionary Computation*, vol. 10, no. 6, pp. 646–657, 2006.
[29] A. K. Qin, V. L. Huang, and P. N. Suganthan, “Differential evolution algorithm with strategy adaptation for global numerical optimization,” *IEEE Transactions on Evolutionary Computation*, vol. 13, no. 2, pp. 398–417, 2009.

[30] R. Mendes, J. Kennedy, and J. Neves, “The fully informed particle swarm: simpler, maybe better,” *IEEE Transactions on Evolutionary Computation*, vol. 8, no. 3, pp. 204–210, 2004.

[31] F. Herrera and M. Lozano, “Gradual distributed real-coded genetic algorithms,” *IEEE Transactions on Evolutionary Computation*, vol. 4, no. 1, pp. 43–62, 2000.

[32] J. Liu, *Advanced PID Control and MATLAB Simulation*, Electronic Industry Press, 2003.

[33] J. Zhang, J. Zhuang, H. Du, and S. Wang, “Self-organizing genetic algorithm based tuning of PID controllers,” *Information Sciences*, vol. 179, no. 7, pp. 1007–1017, 2009.

[34] R. Haber-Haber, R. Haber, M. Schmittdiel, and R. M. del Toro, “A classic solution for the control of a high-performance drilling process,” *International Journal of Machine Tools and Manufacture*, vol. 47, no. 15, pp. 2290–2297, 2007.