Copyright © 2018 Tech Science Press CMES, vol.117, no.1, pp.29-57, 2018

Improved Teaching-Learning-Based Optimization Algorithm for Modeling NO_X Emissions of a Boiler

Xia Li[1, 2], Peifeng Niu[1, *], Jianping Liu[2] and Qing Liu[2]

Abstract: An improved teaching-learning-based optimization (I-TLBO) algorithm is proposed to adjust the parameters of extreme learning machine with parallel layer perception (PELM), and a well-generalized I-TLBO-PELM model is obtained to build the model of NO_X emissions of a boiler. In the I-TLBO algorithm, there are four major highlights. Firstly, a quantum initialized population by using the qubits on Bloch sphere replaces a randomly initialized population. Secondly, two kinds of angles in Bloch sphere are generated by using cube chaos mapping. Thirdly, an adaptive control parameter is added into the teacher phase to speed up the convergent speed. And then, according to actual teaching-learning phenomenon of a classroom, students learn some knowledge not only by their teacher and classmates, but also by themselves. Therefore, a self-study strategy by using Gauss mutation is introduced after the learning phase to improve the exploration ability. Finally, we test the performance of the I-TLBO-PELM model. The experiment results show that the proposed model has better regression precision and generalization ability than eight other models.

Keywords: Bloch sphere, qubits, self-learning, improved teaching-learning-based optimization (I-TLBO) algorithm.

1 Introduction

Reducing NO_X emissions of a boiler has been paid a significant attention recently in the economic development of power plants. In order to implement the reduction of NO_X emissions, a precise model of NO_X emissions firstly needs to be built. Due to the complex nonlinear relationship between the NO_X emissions of the boiler and its influencing factors, it is difficult to build an accurate mathematical model by using mechanism modeling methods [Wang and Yan (2011)]. Artificial neural networks (ANNs) based on data-driven modeling is an effective method in solving this problem [Zhou, Cen and Fan (2004); Ilamathi, Selladurai, Balamurugan et al. (2013)]. However, the conventional neural network has the disadvantages of large amount of calculation, slow training speed, poor generalization ability and easy to fall into local minimum points. Extreme learning machine with parallel layer perception (PELM) [Tavares, Saldanha and Vieira (2015)] is

[1] School of Electrical Engineering, Yanshan University, Qinhuangdao, 066004, China.

[2] School of Mathematics and Information Science &Technology, Hebei Normal University of Science and Technology, Qinhuangdao, 066004, China.

[*] Corresponding Author: Peifeng Niu. Email: niupeifeng2014@163.com.

a new type of neural network. In the PELM, the input weights and thresholds of hidden layer in the nonlinear part are randomly generated, and then the input weights and thresholds of hidden layer in the linear part are obtained by the generalized inverse of the matrix. The PELM has some good characteristics: Low model complexity, high calculation speed and good generalization ability. So, this network can overcome the shortcomings of back propagation (BP) neural network, such as large iterative calculation amount, slow training speed and poor generalization ability. In Tavares et al. [Tavares, Saldanha and Vieira (2015)], the PELM is applied to solve twelve different regression and six classification problems. The experiment results show that the PELM with high speed can achieve very good generalization performance. Furthermore, the PELM has the same regression ability as extreme learning machine (ELM) [Huang, Zhu and Siew (2006); Huang, Zhou, Ding et al. (2012)], but it only uses just a half of hidden neurons and has a much less complex hidden representation. So, the PELM is an efficient modeling tool, which can be used to solve various regression problems in real life. Therefore, in this study, the PELM is considered to build the model of NO_X emissions of a boiler. Because the PELM randomly selects the input weights and thresholds of hidden layer in the nonlinear part, the regression precision and generalization ability may be affected. It is necessary to select the optimal input weights and thresholds in the PELM. Therefore, we need to find an efficient optimization algorithm to optimize the PELM.

Teaching-learning-based optimization (TLBO) algorithm is an intelligent optimization algorithm based on the teaching-learning in classroom [Rao, Savsani and Vakharia (2011); Rao, Savsani and Vakharia (2012)]. This algorithm needs fewer parameters setting, but it can achieve higher calculation precision. The TLBO algorithm is simple in the concept, easy to understand, and fast in the calculation speed. Therefore, the TLBO algorithm has attracted many scholars' attention and has been applied in many fields [Pawar and Rao (2013); Rao and Kalyankar (2013); Bhattacharyya and Babu (2016)]. The TLBO algorithm has many advantages, but it also inevitably has some shortcomings. When it is used to solve some global optimization problems, the diversity of the population decreases with increasing number of iterations. It is very easy to fall into the local minimum and even stagnate. On the other hand, the convergent speed in the early stage is fast, but the convergent speed in the late stage gradually becomes slow. So the effectiveness of the TLBO algorithm is affected.

To overcome the shortcomings and improve the exploration ability and the exploitation performance of the TLBO, an improved TLBO (I-TLBO) algorithm is proposed in this study. In the I-TLBO algorithm, there are four major highlights. Firstly, quantum initialization by using the qubits on Bloch sphere replaces random initialization in original TLBO. This highlight can improve the quality of the initial population. Secondly, two kinds of angles in Bloch sphere are generated by using cube chaos mapping. The introduction of the cube chaos mapping can increase the diversity of the initial population. The search ability of ergodic to the solution space is improved. Thirdly, an adaptive control parameter is added into the teacher phase to speed up the convergent speed. And then, according to actual teaching-learning phenomenon of a classroom, students learn some knowledge not only by their teacher and classmates, but also by themselves. Therefore, a self-study strategy by using Gauss mutation is introduced after the learning phase to improve the exploration ability. The effectiveness of the I-TLBO algorithm is benchmarked on eight well-known testing functions. The performance of the I-TLBO algorithm is compared with particle swarm optimization (PSO) algorithm [Kennedy and Eberhart (1995)], grey wolf optimizer (GWO) algorithm [Mirjalili, Mirjalili and Lewis (2014)], TLBO algorithm, mTLBO algorithm [Satapathy and Naik (2013)], and TLBO with crossover (C-TLBO) algorithm [Ouyang and Kong (2014)]. Wilcoxon signed rank test shows that the proposed I-TLBO algorithm is able to provide very good results compared to five other algorithms. So, the I-TLBO algorithm becomes a good selection to optimize the PELM, and a well-generalized I-TLBO-PELM model is obtained to predict NO_X emissions of a boiler.

The rest of this study is arranged as follows. In Section 2, the PELM model and the TLBO algorithm are reviewed, respectively. In Section 3, the I-TLBO algorithm is proposed. In Section 4, the experimental study shows the validity of the I-TLBO algorithm. In Section 5, the I-TLBO-PELM model is proposed and is applied to model NO_X emissions of the boiler. Finally, Section 6 concludes this study.

2 Basic concepts and related works

2.1 The model of the PELM

The PELM was proposed by Tavares et al. [Tavares, Saldanha and Vieira (2015)]. The structure of the PELM is described in Fig. 1.

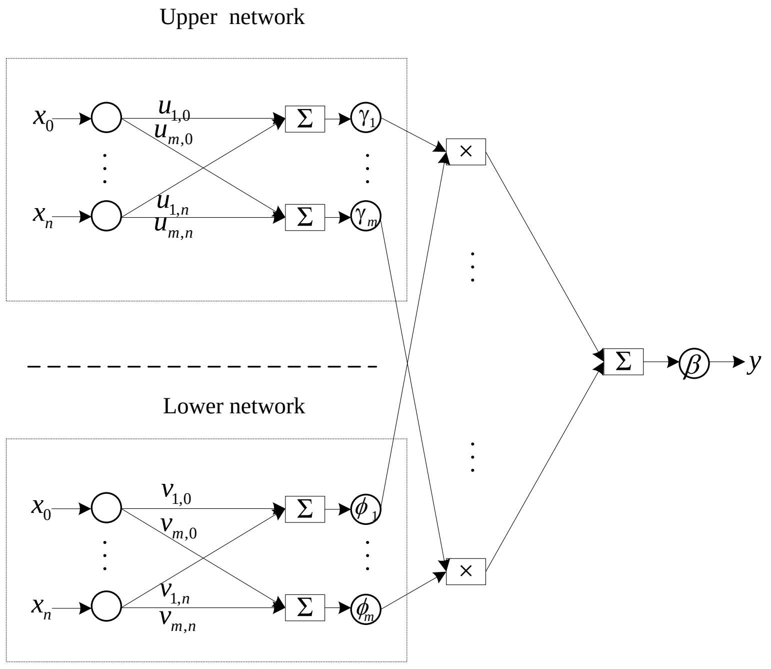


Figure 1: The structure of the PELM

Suppose a data set with N arbitrary distinct samples $\left(x_l, y_l\right) \in R^{n\times N} \times R^{1\times N}$, $l = 1, 2, \cdots, N$, the PELM has m hidden layer nodes. U is the matrix of the weights and the thresholds of hidden layer in the upper network. V is the matrix of the input weights and thresholds of hidden layer in the lower network. $\beta(\cdot), \gamma(\cdot)$ and $\phi(\cdot)$ are activation functions. Then, the PELM is mathematically modeled as

$$y_l = \beta\left(\sum_{j=1}^{m}\left[\gamma\left(a_{jl}\right)\phi\left(b_{jl}\right)\right]\right) \tag{1}$$

where a_{jl} and b_{jl} are

$$a_{jl} = \sum_{i=1}^{n+1} u_{ji} x_{il} \tag{2}$$

$$b_{jl} = \sum_{i=1}^{n+1} v_{ji} x_{il} \tag{3}$$

u_{ji} and v_{ji} are the elements of U and V; x_{il} is the i th input of the l th sample and y_l is the output of the l th sample. In the PELM, the input-output mapping is made by applying the product of functions [Tavares, Saldanha and Vieira (2015)].

As a particular case of Eq. (1), when $\beta(\cdot)$ and $\gamma(\cdot)$ are linear functions, the network output y_l is computed as

$$y_l = \sum_{j=1}^{m}\left[a_{jl}\phi\left(b_{jl}\right)\right] \tag{4}$$

Substituting Eqs. (2) and (3) into Eq. (4), we can obtain Eq. (5)

$$y_l = \sum_{j=1}^{m}\left[\sum_{i=1}^{n+1} u_{ji} x_{il}\phi\left(\sum_{i=1}^{n+1} v_{ji} x_{il}\right)\right] \tag{5}$$

In Eq. (5), v_{ji} of the nonlinear part is randomly selected in $[-1,1]$, and then u_{ji} of the linear part is obtained by the least square method.

2.2 An introduction of the TLBO algorithm

The TLBO algorithm simulates the influence of a teacher on learners in a class teaching to obtain the global optimal solution. Compared with other nature-inspired algorithms, The TLBO algorithm has the advantages of the simple principle, the few parameters and the high precision. In the TLBO, the learners are considered as the population X .The teacher is considered as the most knowledgeable person in a class and shares the knowledge to the students to improve the marks of class. The learning result of a learner is analogous to the fitness $f\left(X_i\right), i = 1:M$, where M is the size of the population. There are two parts in the TLBO: The teacher phase and the learner phase. The teacher phase means that students learn the knowledge from the teacher. The learner phase means that students learn the knowledge from the classmates by communicating with each other.

2.2.1 The teacher phase

In this stage, the teacher teaches individual knowledge to their students to improve the average level of the students in whole class. The students learn the knowledge from teachers to narrow the gap between the teachers and the students. Let X_i and $X_{new,i}$, $i=1$: M denote scores before and after study, respectively. The average level of the students in the whole class is X_{mean}. The teacher tries to raise the level of students to $X_{teacher}$ by differential teaching. Then the whole teaching process can be expressed as

$$X_{new,i} = X_i + r\left(X_{teacher} - \left(T_F \cdot X_{mean}\right)\right) \tag{6}$$

where $r \in [0,1]$ is a random number, and T_F is a teaching factor as shown below

$$T_F = round\left[1 + rand(0,1)\right] \tag{7}$$

Accept $X_{new,i}$, if it gives a better function value.

2.2.2 The learner phase

After the teacher has finished teaching, the knowledge levels of the students have been improved. However, the individual level of every student is different from others, so the students can still learn new knowledge from other superior individuals. At the stage of mutual learning among students, the student X_i randomly selects another student X_j ($i \neq j$) by contrast learning. The process of learning among the classmates is

$$X_{new,i} = \begin{cases} X_i + r_i\left(X_i - X_j\right) & f\left(X_i\right) < f\left(X_j\right) \\ X_i + r_i\left(X_j - X_i\right) & f\left(X_i\right) > f\left(X_j\right) \end{cases} \tag{8}$$

Accept $X_{new,i}$, if it gives a better function value. The algorithm will continue its iterations until reaching the maximum number of iterations.

3 An improved TLBO （I-TLBO）algorithm

The parameter setting of the TLBO algorithm is fewer and the calculation precision is higher. However, the TLBO is also easy to fall into a local minimum, and it can not find the optimal solutions. Therefore, an improved TLBO algorithm called I-TLBO algorithm is proposed. In the I-TLBO algorithm, there are four major highlights, which are expressed in detail as follows.

3.1 Quantum initialized population by using the qubits on Bloch sphere

In general, population-based optimization techniques start the optimization process with a set of random solutions, but they need sufficient individuals and iterations to find the optimal solution [Mirjalili (2016)]. Vedat et al. [Vedat and Ayşe (2008)] pointed out that initial population of high quality could reduce the number of search to reach the optimum design in the solution space. The initial solutions were coded by using Bloch spherical coordinates, which expanded the quantity of the global optimal solution and improved the probability to obtain the global optimal solution [Huo, Liu, Wang et al. (2017)]. Because the original TLBO algorithm uses a randomly initialized population, it is difficult to guarantee that the population has good quality solutions. For overcoming the defect, the population is generated by using qubits based on Bloch spherical coordinate. The detailed scheme is described in this part.

In quantum computing, a qubit represents the smallest unit of the information, whose state can be expressed by Eq. (9)

$$|\Phi\rangle = \cos(\theta/2)|0\rangle + e^{i\varphi}\sin(\theta/2)|1\rangle \tag{9}$$

where the angles φ and θ can determine the point on the Bloch sphere [Li (2014)]. Representation of the qubit on Bloch sphere is shown in Fig. 2.

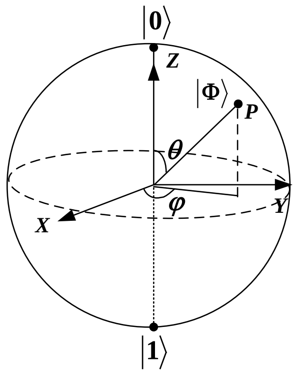


Figure 2: Representation of the qubit on Bloch sphere

According to Fig. 2, a qubit corresponds to a point on the Bloch sphere. Therefore, a qubit $|\Phi\rangle$ can be presented by using the Bloch spherical coordinate in Eq. (10).

$$|\Phi\rangle = [\cos\varphi\sin\theta \quad \sin\varphi\sin\theta \quad \cos\theta]^{\mathrm{T}} \tag{10}$$

Let P_i be i th candidate solution of the population, and the coding scheme is shown in Eq. (11).

$$P_i = \begin{vmatrix} \cos\varphi_{i1}\sin\theta_{i1} \\ \sin\varphi_{i1}\sin\theta_{i1} \\ \cos\theta_{i1} \end{vmatrix} \begin{matrix} \cdots \\ \cdots \\ \cdots \end{matrix} \begin{vmatrix} \cos\varphi_{id}\sin\theta_{id} \\ \sin\varphi_{id}\sin\theta_{id} \\ \cos\theta_{id} \end{vmatrix} \tag{11}$$

$$\varphi_{ij} = 2\pi \times rand \tag{12}$$

$$\theta_{ij} = \pi \times rand \tag{13}$$

where d is the dimension of the optimization space, $i = 1, 2, \cdots, M$ and $j = 1, 2, \cdots, d$. Each initial solution P_i corresponds to three locations on Bloch sphere, which is shown in Eq. (14). They are on X axis, Y axis and Z axis, respectively.

$$\begin{cases} P_{ix} = \left(\cos\varphi_{i1}\sin\theta_{i1},\cdots,\cos\varphi_{id}\sin\theta_{id}\right) \\ P_{iy} = \left(\sin\varphi_{i1}\sin\theta_{i1},\cdots,\sin\varphi_{id}\sin\theta_{id}\right) \\ P_{iz} = \left(\cos\theta_{i1},\cos\theta_{i2},\cdots,\cos\theta_{id}\right) \end{cases} \tag{14}$$

According to the relationship between the qubits and the coordinates of the points on Bloch sphere, the global optimal solution P^* is possibly obtained on the three circles [Li (2014)], which is shown in Fig. 3.

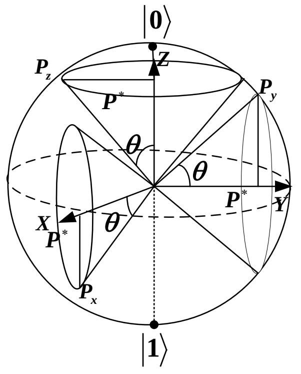


Figure 3: A point P^* corresponds to three circles on Bloch sphere

Let j th dimension of P_i be $\left[x_{ij}, y_{ij}, z_{ij}\right]^{\mathrm{T}}$ on Bloch sphere. The range of the j th dimension is $\left[a_j, b_j\right]$ in the original solution space. So, the transformation formula from Bloch sphere to the original solution space is shown in Eq. (15)

$$\begin{cases} X_{ix}^{j} = \frac{1}{2}\left[b_j\left(1+x_{ij}\right)+a_j\left(1-x_{ij}\right)\right] \\ X_{iy}^{j} = \frac{1}{2}\left[b_j\left(1+y_{ij}\right)+a_j\left(1-y_{ij}\right)\right] \\ X_{iz}^{j} = \frac{1}{2}\left[b_j\left(1+z_{ij}\right)+a_j\left(1-z_{ij}\right)\right] \end{cases} \tag{15}$$

Then select M individuals with the best fitness values as the initial population among all $3M$ candidate solutions. Because the initial solutions of high quality are possibly obtained from three coordinate axes, the number of initial solutions of high quality is extended. Thereby, the introduction of quantum mechanism increases the convergent probability. This highlight makes the I-TLBO algorithm more easily find the global optimal solution.

3.2 The angles φ_{ij} and θ_{ij} are generated by using cube chaos mapping

The angles φ_{ij} and θ_{ij} in Eqs. (12-13) are randomly generated by using the uniform distribution. Therefore, a random initialization reduces the search efficiency of the algorithm to some extent. Chaos is a kind of universal nonlinear dynamic phenomenon, and the chaotic motion can traverse all states according to its own law within a certain range. This advantage can be used as an effective method to avoid falling into a local minimum. Because of the ergodicity of the chaos, the initialized population by using chaos is diverse enough to potentially reach every mode in the multimodal functions [Gandomi and Yang (2014)]. Jothiprakash et al. [Jothiprakash and Arunkumar (2013)] generated the initial population for Genetic algorithm (GA) and Differential Evolution (DE) algorithm by chaos theory and applied it to a water resource system problem. They showed that GA and DE with initial chaotic populations outperformed those of the standard GA and DE algorithm. Furthermore, the combination of chaos and meta-heuristic optimization algorithms has a faster iterative search speed than the uniform distribution [Coelho and Mariani (2008)]. All these studies show the potential of chaos theory for improving the performance of the optimization algorithms.

The literature [Zhou, Liu and Zhao (2012)] proved that cube chaos mapping based on time series has good homogeneity. Therefore, we generate φ_{ij} and θ_{ij} by using the cube chaos mapping, which is shown in Eq. (16).

$$z_{k+1} = 4z_k^3 - 3z_k, -1 \le z_k \le 1, k = 1, 2, \cdots \tag{16}$$

The detailed implement process of the cube chaos mapping in the I-TLBO algorithm are given as follows:

Randomly generate two d-dimensional vectors $\boldsymbol{\sigma} = [\sigma_1, \sigma_2, \cdots, \sigma_d]$ and $\tau = [\tau_1, \tau_2, \cdots, \tau_d]$. Each component of these two vectors is between 0 and 1.

```
for $j = 1 : d$
  $\varphi_{1j} = \sigma_j$
  $\theta_{1j} = \tau_j$
    for $i = 1 : M - 1$
            $\varphi_{i+1j} = 4\varphi_{ij}{}^3 - 3\varphi_{ij}$
            $\theta_{i+1j} = 4\theta_{ij}{}^3 - 3\theta_{ij}$
    end
end
```

Because the introduction of cube chaos mapping can increase the diversity of the population, it expands the ergodic ability to search solution space. Therefore, the highlight improves the quality of the initial population once again and enhances the global exploration ability of the TLBO algorithm.

3.3 An Adaptive control parameter in the teacher phase

In order to balance the local exploitation ability and the global exploration ability of TLBO algorithm, an adaptive control parameter, as shown in Eq. (17), is proposed in the teacher phase.

$$w = w_{start} - (w_{start} - w_{end}) \times (t / G)^{\frac{1}{t}} \tag{17}$$

Eq. (6) is changed into Eq. (18) below in the teacher phase.

$$X_{new,i} = w \times X_i + r\left(X_{teacher} - \left(T_F \cdot X_{mean}\right)\right) \tag{18}$$

In Eq. (17), w_{start} and w_{end} are the maximum value and the minimum value of the control parameter, respectively; t is the current iteration number, and G is the maximum iteration number. A larger value of w facilitates the global exploration, while a smaller value of w facilitates the local exploitation. Selecting suitable value of w can provide a balance between the global exploration and the local exploitation. Therefore, in Eq. (17), the control parameter w nonlinearly decreases with increasing of the number of iterations. This highlight makes the algorithm advantageous to the global exploration in the early stage and the local exploitation in the late stage of the iteration.

3.4 A self-learning process by using Gauss mutation after the learner phase

In the TLBO algorithm, the students only improve their marks by learning from their teacher and classmates. In fact, the students not only rely on others to learn some knowledge, but also they often make unremitting efforts by themselves to improve their marks in actual learning. So inspired by the idea of self-learning to gain some knowledge, we add the self-learning process after the student phase. We implement the self-learning process by using Gauss mutation. This highlight is used to improve the exploration ability against the premature convergence. The self-learning process is illustrated as follows.

$$X_{new,i} = \begin{cases} X_i + randn \times h & abs(X_i) < 1 \\ X_i \times (1 + randn \times h) & abs(X_i) > 1 \end{cases} \tag{19}$$

where h is the step size; X_i is the current best solution in learner phase; $abs(X_i)$ denotes the absolute value of the elements of X_i, and $randn$ is a Gauss random number. Accept $X_{new,i}$, if it gives a better function value.

The flow chart of the I-TLBO algorithm is shown in Fig. 4. By adding above four improvements, we enhance the exploration ability and exploitation ability of the TLBO algorithm. Therefore, the optimization capability of the I-TLBO algorithm is better than the TLBO algorithm. This analysis is consistent with the following simulation results.

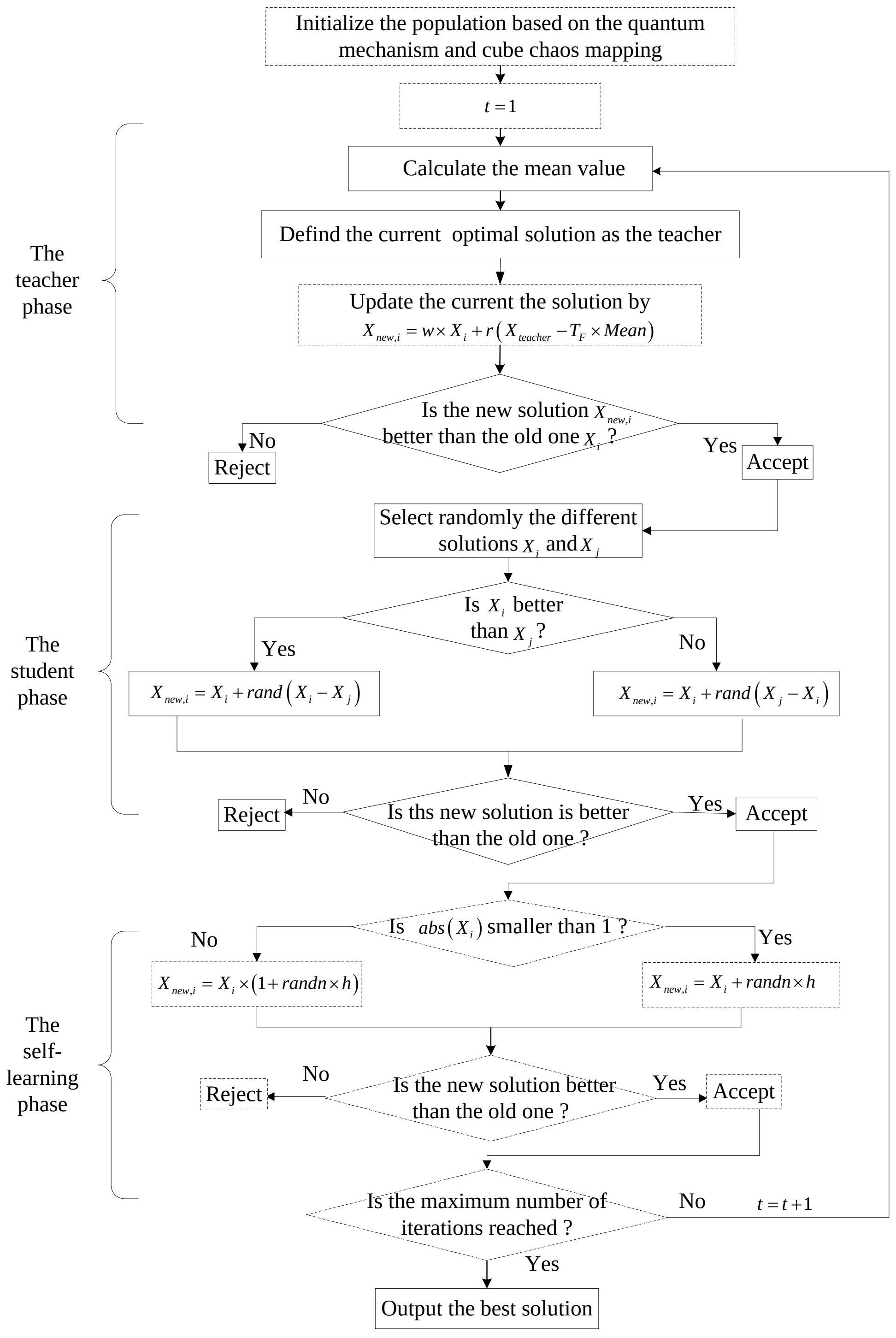

Initialize the population based on the quantum mechanism and cube chaos mapping
$t=1$
Calculate the mean value
Defind the current optimal solution as the teacher
Update the current the solution by $X_{new,i}=w\times X_i+r\left(X_{teacher}-T_F\times Mean\right)$
Is the new solution $X_{new,i}$ better than the old one X_i ?
No
Reject
Yes
Accept
The teacher phase
Select randomly the different solutions X_i and X_j
Is X_i better than X_j ?
Yes
No
$X_{new,i}=X_i+rand\left(X_i-X_j\right)$
$X_{new,i}=X_i+rand\left(X_j-X_i\right)$
The student phase
Is ths new solution is better than the old one ?
No
Reject
Yes
Accept
Is $abs\left(X_i\right)$ smaller than 1 ?
No
Yes
$X_{new,i}=X_i\times\left(1+randn\times h\right)$
$X_{new,i}=X_i+randn\times h$
The self-learning phase
Is the new solution better than the old one ?
No
Reject
Yes
Accept
Is the maximum number of iterations reached ?
No
$t=t+1$
Yes
Output the best solution

Figure 4: The flow chart of the I-TLBO algorithm

4 The simulation experiments on the benchmark functions

The optimization performance of the I-TLBO algorithm on a test of benchmark functions is compared with the PSO, GWO, TLBO, C-TLBO, and mTLBO. The performance of the proposed algorithm is verified on eight classic benchmark functions.

4.1 The classical benchmark functions

These benchmark functions [Rashedi, Nezamabadi-Pour and Saryazdi (2009); Mirjalili, Mirjalili and Lewis (2014)] are the classical functions utilized by many researchers, which are shown in Tab. 1. F_1-F_4 is unimodal benchmark functions, which are used to verify exploitation ability of fast finding the optimal solutions. The convergent speeds of unimodal benchmark functions are more interesting than the final results of optimization. F_5-F_8 are used to verify the exploration ability of finding the global optimal solutions. For the multimodal functions, the computation results are much more important. They reflect an algorithm's abilities of escaping from poor local optima and finding a good near-global optimum [Xin, Liu and Lin (2002)].

Table1: The classical benchmark functions

Benchmark function	Domain	Optimum location	Optimal value
$F_1(x)=\sum_{i=1}^{d} x_i^2$	$[-100,100]^d$	$[0]^d$	0
$F_2(x)=\max(\lvert x_i\rvert)$	$[-100,100]^d$	$[0]^d$	0
$F_3(x)=\sum_{i=1}^{d}[\lvert x_i+0.5\rvert]^2$	$[-100,100]^d$	$[0.5]^d$	0
$F_4(x)=\sum_{i=1}^{d} ix_i^2+rand(0,1)$	$[-1.28,1.28]^d$	$[0]^d$	0
$F_5(x)=-\sum_{i=1}^{d}\left(x_i\sin\left(\sqrt{\lvert x_i\rvert}\right)\right)$	$[-500,500]^d$	$[420.9687]^d$	$-418.9829\times d$
$F_6(x)=-\sum_{i=1}^{d}\left(x_i^2-10\cos(2\pi x_i)+10\right)$	$[-5.12,5.12]^d$	$[0]^d$	0
$F_7(x)=-20\exp\left(-0.2\sqrt{\frac{1}{D}\sum_{i=1}^{d}x_i^2}\right)-\exp\left(\frac{1}{D}\sum_{i=1}^{d}\cos 2\pi x_i\right)+20+e$	$[-32,32]^d$	$[0]^d$	0
$F_8(x)=\frac{1}{400}\sum_{i=1}^{d}x_i^2-\prod_{i=1}^{d}\cos\left(\frac{x_i}{\sqrt{i}}\right)+1$	$[-600,600]^d$	$[0]^d$	0

4.2 The experiment study and results analysis

All the programs are run under MATLAB 2009a environment in Windows 7, 2.2 GHZ CPU. Every experiment is repeated 10 times. The best value, the mean and the standard deviation (S.D.) of the best function values have been recorded for six algorithms. In order to fairly compare the computation results, six algorithms use the same population size $M = 30$ and the maximum number of generations $G = 500$. In the PSO, $c_1 = c_1 = 1.8$. In the C-TLBO, the crossover factor (CR) is set as 0.95, which is the same as [Ouyang and Kong (2014)]. In the I-TLBO, after a lot of experiments, the values of h, w_{start} and w_{end} are all set as 0.03, 0.08 and 0.01 respectively. The experiment results are given in Tabs. 2-3 and Fig. 5. The best results are displayed in the bold face.

Table 2: Computational results of six algorithms when $d = 30$

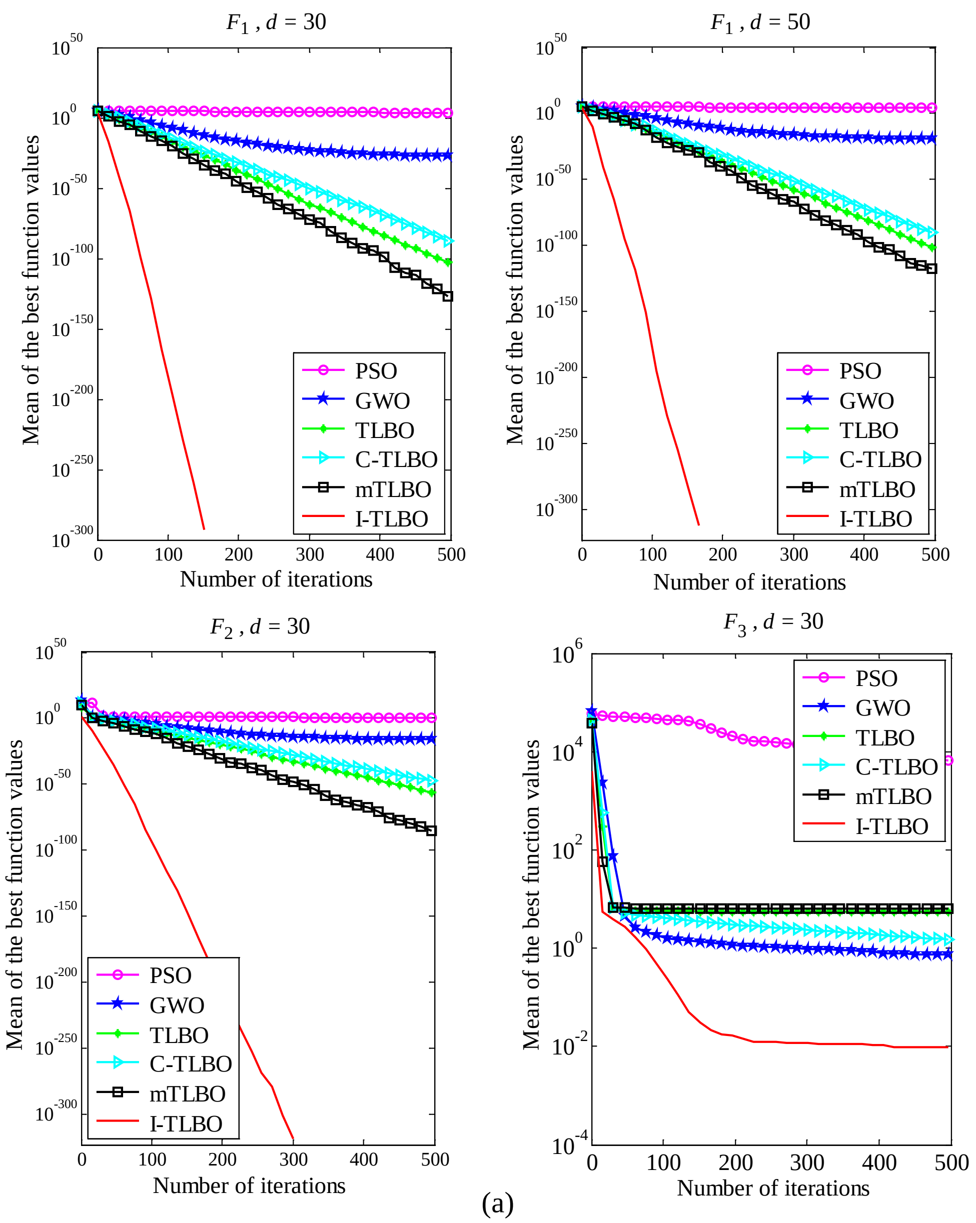
Function	P.T.	PSO	GWO	TLBO	C-TLBO	mTLBO	I-TLBO
	Best	5.6692e+002	2.5410e-028	5.6784e-109	1.1179e-097	5.4745e-137	**0**
F_1	Mean	4.5224e+003	1.5463e-027	1.8898e-104	2.8250e-089	1.5312e-128	**0**
	S.D.	2.4709e+003	1.3553e-027	4.4671e-104	8.9284e-089	3.0131e-128	**0**
	Best	2.0597e+000	3.3000e-017	3.4924e-062	1.8062e-051	9.2114e-097	**0**
F_2	Mean	4.3112e+000	8.7321e-017	9.2415e-058	3.0742e-048	1.3522e-086	**0**
	S.D.	2.6903e+000	7.3823e-017	2.8742e-057	9.4545e-048	3.3641e-086	**0**
	Best	9.6490e+002	2.5093e-001	4.5090e+000	1.0770e+000	5.8059e+000	**7.4420e-003**
F_3	Mean	6.6702e+003	6.2577e-001	5.5461e+000	1.5142e+000	6.4642e+000	**9.4372e-003**
	S.D.	4.4675e+003	3.5977e-001	6.7364e-001	4.1869e-001	4.5061e-001	**1.8115e-003**
	Best	7.1834e-004	3.4349e-004	1.4381e-003	6.8791e-004	8.5543e-004	**8.4759e-006**
F_4	Mean	2.0932e-003	2.1347e-003	3.4714e-003	1.8940e-003	2.5537e-003	**1.2209e-003**
	S.D.	1.0063e-003	1.4812e-003	1.3236e-003	1.2337e-003	1.5676e-003	**9.2795e-004**
	Best	-5.3674e+003	-6.9367e+003	-5.7710e+003	-8.5501e+003	-5.1081e+003	**-9.6351e+003**
F_5	Mean	-3.8689e+003	-6.1003e+003	-4.6600e+003	-7.4291e+003	-4.4785e+003	**-8.5263e+003**
	S.D.	7.3830e+002	7.2657e+002	5.5730e+002	7.8395e+002	**3.7110e+002**	5.8876e+002
	Best	5.7551e+001	5.6843e-014	**0**	**0**	**0**	**0**
F_6	Mean	9.8204e+001	1.8409e+000	**0**	1.1527e+001	**0**	**0**
	S.D.	2.0059e+001	3.3256e+000	**0**	2.3457e+001	**0**	**0**
	Best	1.9450e+001	7.5495e-014	4.4409e-015	4.4409e-015	**8.8818e-016**	**8.8818e-016**
F_7	Mean	1.9772e+001	9.0772e-014	6.2172e-015	7.6383e-015	**8.8818e-016**	**8.8818e-016**
	S.D.	1.7093e-001	1.2313e-014	1.8724e-015	3.1107e-015	**0**	**0**
	Best	3.0690e+002	**0**	**0**	**0**	**0**	**0**
F_8	Mean	4.4463e+002	1.0024e-003	5.7398e-003	7.8008e-004	**0**	**0**
	S.D.	1.0398e+002	3.1698e-003	1.8151e-002	2.4668e-003	**0**	**0**

As seen from Tab. 2, when $d = 30$, the I-TLBO algorithm presents the best value and the lowest mean in all eight functions and the lowest S.D. in seven functions except F_5. The mTLBO algorithm shows the lowest S.D. on F_5, but the best value and the mean are much worse than the I-TLBO algorithm. The mTLBO algorithm also presents the best value, the lowest mean and S.D. on F_6-F_8. The TLBO algorithm also presents the best value, the lowest mean and S.D. on F_6. However, from Fig. 5, we can see that the convergent speed of the I-TLBO algorithm is much faster than the mTLBO and the TLBO on F_6-F_8.

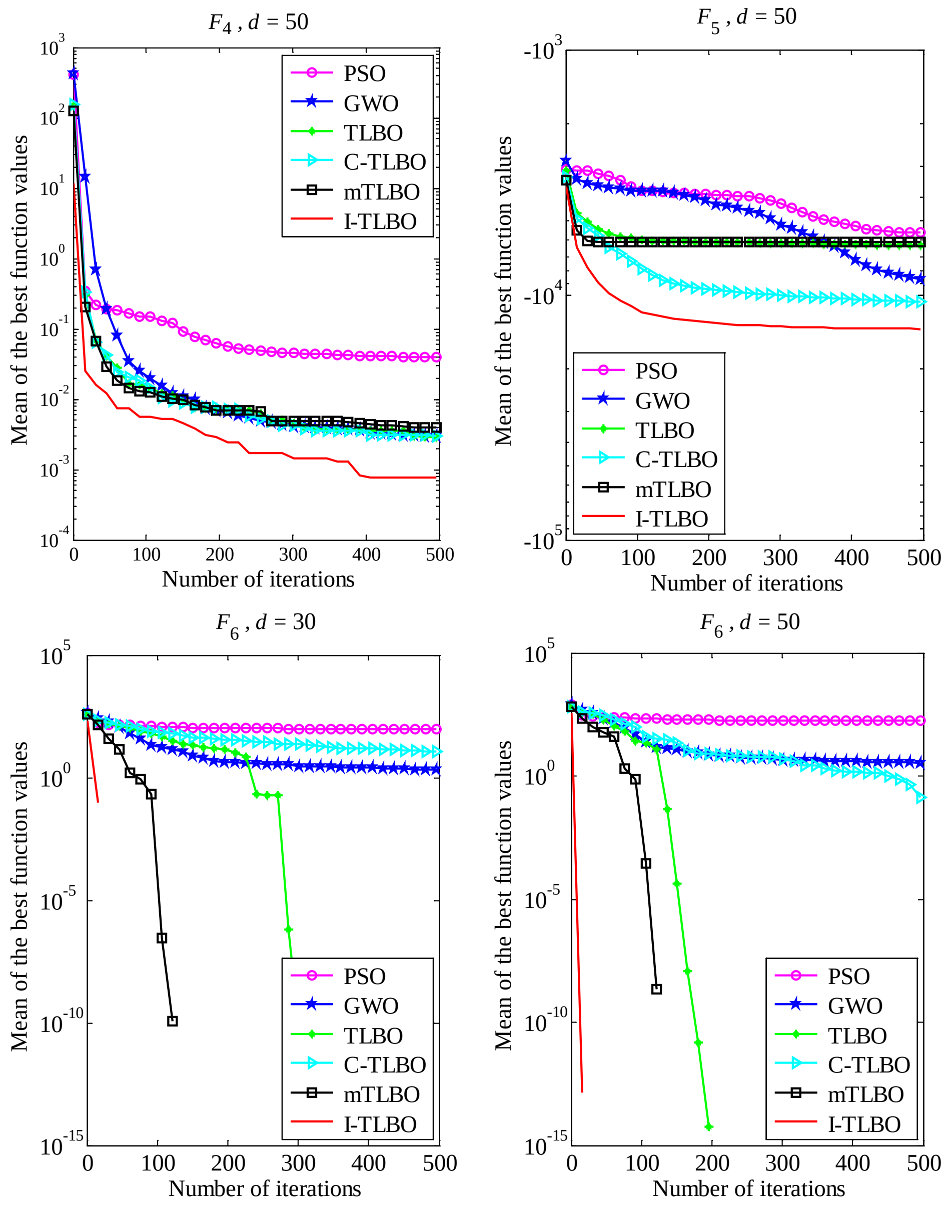
As seen from Tab. 3, when $d = 50$, the I-TLBO algorithm presents the best value and the lowest mean in all eight functions and the lowest S.D. in seven functions except F_5. The mTLBO algorithm shows the lowest S.D. on F_5, but the best value and the mean are much worse than the I-TLBO algorithm. The TLBO algorithm also presents the best value, the lowest mean and S.D. on F_6 and F_8. The mTLBO algorithm also presents the best value, the lowest mean and S.D. on F_6-F_8. However, from Fig. 5, we can see that the convergent speed of the I-TLBO algorithm is much faster than the TLBO and the mTLBO on F_6-F_8..

Table 3: Computational results of six algorithms when $d = 50$

Function	P.T.	PSO	GWO	TLBO	C-TLBO	mTLBO	I-TLBO
	Best	7.7681e+003	9.0536e-021	1.3064e-110	1.8895e-096	4.7345e-136	**0**
F_1	Mean	1.7562e+004	4.9485e-020	2.9328e-103	8.0931e-092	6.0380e-119	**0**
	S.D.	7.7786e+003	6.7452e-020	5.3523e-103	1.7123e-091	1.9094e-118	**0**
	Best	9.0643e+000	1.0774e-012	6.6841e-066	1.4529e-052	3.7106e-092	**0**
F_2	Mean	1.8680e+001	2.6337e-012	3.4967e-060	6.2358e-048	3.5912e-084	**0**
	S.D.	5.7413e+000	1.2405e-012	4.5889e-060	1.1680e-047	1.1350e-083	**0**
	Best	6.6713e+003	1.8754e+000	8.6676e+000	3.8844e+000	1.0617e+001	**2.5374e-002**
F_3	Mean	1.8603e+004	2.7127e+000	1.0235e+001	5.0728e+000	1.1701e+001	**3.5439e-002**
	S.D.	7.6168e+003	5.3399e-001	7.8459e-001	7.5808e-001	5.9991e-001	**5.7472e-003**
	Best	8.8815e-003	1.5034e-003	2.4495e-004	1.0035e-003	2.8945e-004	**1.9541e-004**
F_4	Mean	3.9382e-002	2.9301e-003	2.9070e-003	3.0162e-003	3.9057e-003	**7.5882e-004**
	S.D.	3.2274e-002	9.3282e-004	1.7938e-003	2.7191e-003	2.4299e-003	**5.3823e-004**
	Best	-8.2867e+003	-9.7367e+003	-8.3468e+003	-1.2760e+004	-7.3004e+003	**-1.6543e+004**
F_5	Mean	-5.5771e+003	-8.6260e+003	-6.2701e+003	-1.0673e+004	-6.1143e+003	**-1.3778e+004**
	S.D.	1.1607e+003	1.0109e+003	9.6979e+002	1.0542e+003	**6.4453e+002**	1.3340e+003
	Best	1.2180e+002	1.1369e-012	**0**	**0**	**0**	**0**
F_6	Mean	1.7827e+002	3.4459e+000	**0**	1.3082e-001	**0**	**0**
	S.D.	3.0794e+001	3.5364e+000	**0**	4.1368e-001	**0**	**0**
	Best	1.9631e+001	3.1027e-011	4.4409e-015	4.4409e-015	**8.8818e-016**	**8.8818e-016**
F_7	Mean	1.9857e+001	4.6950e-011	6.5725e-015	6.9278e-015	**8.8818e-016**	**8.8818e-016**
	S.D.	9.0011e-002	1.2807e-011	1.8346e-015	1.7161e-015	**0**	**0**
	Best	5.1275e+002	**0**	**0**	**0**	**0**	**0**
F_8	Mean	9.1101e+002	8.5487e-016	**0**	1.2216e-003	**0**	**0**
	S.D.	2.4298e+002	2.7033e-015	**0**	3.8629e-003	**0**	**0**

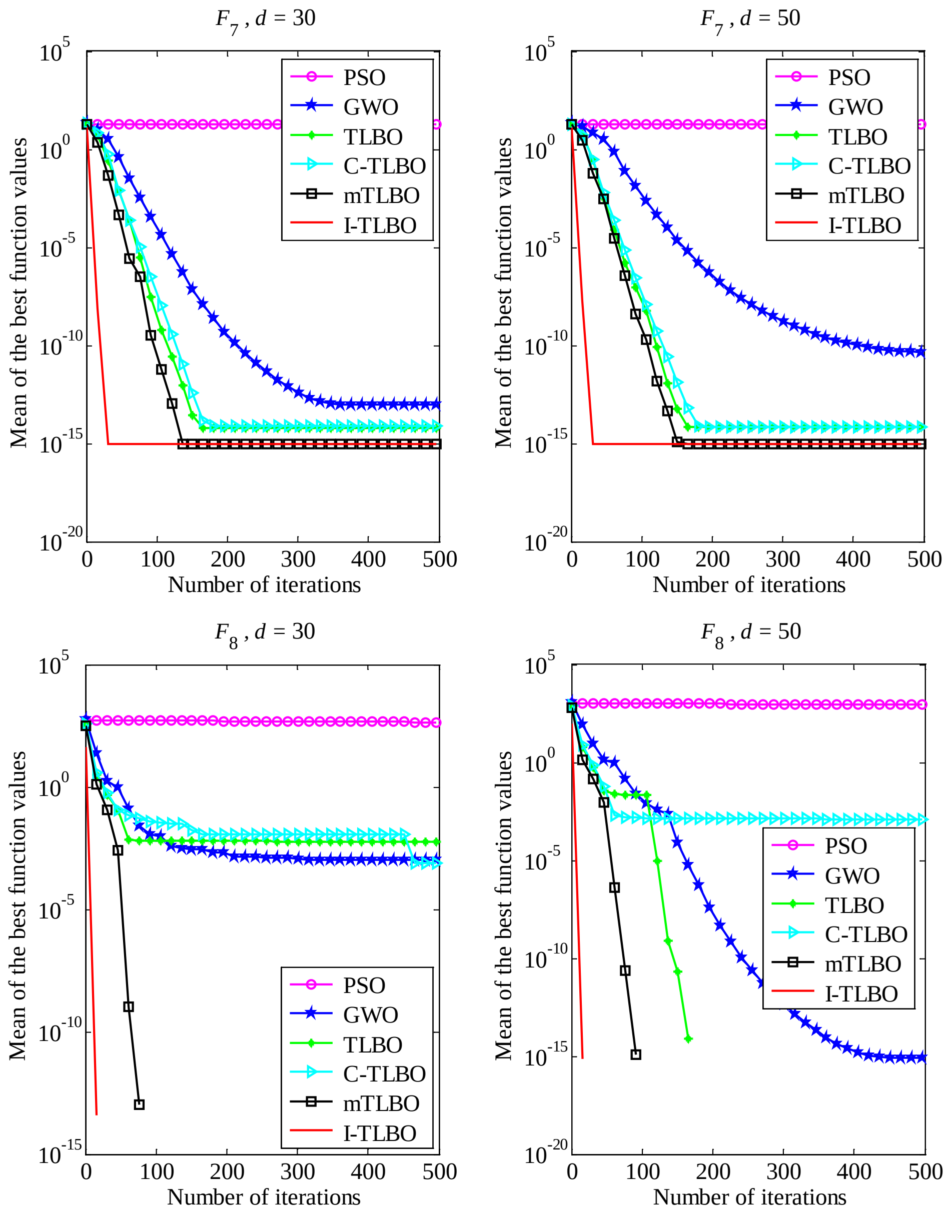

F_1 , d = 30
F_1 , d = 50
F_2 , d = 30
F_3 , d = 30
Mean of the best function values
Number of iterations
PSO
GWO
TLBO
C-TLBO
mTLBO
I-TLBO

(a)



F_4, d = 50
F_5, d = 50
F_6, d = 30
F_6, d = 50
Mean of the best function values
Number of iterations
PSO
GWO
TLBO
C-TLBO
mTLBO
I-TLBO

(b)



(c)

Figure 5: Performance comparison of six algorithms on the benchmark functions

Wilcoxon signed rank test is devoted to detecting possible differences between the I-TLBO and five other optimization algorithms. The "+", "=" and "-" denote that the performance of the I-TLBO algorithm is better than, similar to and worse than that of the corresponding algorithm, respectively. The significance level is set as 0.05. As seen from the results of statistical tests in Tabs. 4-5, the I-TLBO algorithm is significantly better than five other algorithms on the unimodal functions and the multimodal functions.

From the above analysis of the experimental results, it has been found that whether on unimodal or multimodal functions, whether $d = 30$ or $d = 50$, the I-TLBO algorithm is superior to five other optimization algorithms. Therefore, the I-TLBO algorithm is an excellent optimization algorithm, and can be applied to optimize the PELM for modeling NO_X emissions of a boiler.

Table 4: The statistical comparison of the six algorithms by Wilcoxon signed-rank test for $d = 30$

Function	I-TLBO *vs.* PSO				I-TLBO *vs.* GWO			
	p-valve	T^-	T^+	winner	*p*-valve	T^-	T^+	winner
F_1	1.9531e-003	55	0	+	1.9531e-003	55	0	+
F_2	1.9531e-003	55	0	+	1.9531e-003	55	0	+
F_3	1.9531e-003	55	0	+	3.9063e-003	54	1	+
F_4	1.3672e-002	51	4	+	4.8828e-002	47	8	+
F_5	1.9531e-003	55	0	+	1.9531e-003	55	0	+
F_6	1.9531e-003	55	0	+	1.9531e-003	55	0	+
F_7	1.9531e-003	55	0	+	1.9531e-003	55	0	+
F_8	1.9531e-003	55	0	+	1	1	0	=
Total (+/–/=)				8/0/0				7/0/1

(Continued)

Function	I-TLBO *vs.* TLBO				I-TLBO *vs.* CTLBO				I-TLBO *vs.* mTLBO			
	p-valve	T^-	T^+	winner	*p*-valve	T^-	T^+	winner	*p*-valve	T^-	T^+	winner
F_1	1.9531e-003	55	0	+	1.9531e-003	55	0	+	1.9531e-003	55	0	+
F_2	1.9531e-003	55	0	+	1.9531e-003	55	0	+	1.9531e-003	55	0	+
F_3	1.9531e-003	55	0	+	1.9531e-003	55	0	+	1.9531e-003	55	0	+
F_4	3.9063e-003	54	1	+	1.9336e-001	41	14	=	8.3984e-002	45	10	=
F_5	1.9531e-003	55	0	+	2.7344e-002	49	6	+	1.9531e-003	55	0	+
F_6	1	0	0	=	1.2500e-001	10	0	=	1	0	0	=
F_7	1.9531e-003	55	0	+	1.9531e-003	55	0	+	1	0	0	=
F_8	5.0000e-001	1	0	=	1	3	0	=	1	0	0	=
Total (+/–/=)				6/0/2				5/0/3				4/0/4

Table 5: The statistical comparison of the six algorithms by Wilcoxon signed-rank test for $d = 50$

Function	I-TLBO *vs.* PSO				I-TLBO *vs.* GWO			
	p-valve	T^-	T^+	winner	*p*-valve	T^-	T^+	winner
F_1	1.9531e-003	55	0	+	1.9531e-003	55	0	+
F_2	1.9531e-003	55	0	+	1.9531e-003	55	0	+
F_3	1.9531e-003	55	0	+	3.9063e-003	54	1	+
F_4	1.9531e-003	55	0	+	1.9531e-003	55	0	+
F_5	1.9531e-003	55	0	+	1.9531e-003	55	0	+
F_6	1.9531e-003	55	0	+	1.9531e-003	55	0	+
F_7	1.9531e-003	55	0	+	1.9531e-003	55	0	+
F_8	1.9531e-003	55	0	+	1	1	0	=
Total (+/–/=)				8/0/0				7/0/1

(Continued)

Function	I-TLBO *vs.* TLBO				I-TLBO *vs.* CTLBO				I-TLBO *vs.* mTLBO			
	p-valve	T^-	T^+	winner	*p*-valve	T^-	T^+	winner	*p*-valve	T^-	T^+	winner
F_1	1.9531e-003	55	0	+	1.9531e-003	55	0	+	1.9531e-003	55	0	+
F_2	1.9531e-003	55	0	+	1.9531e-003	55	0	+	1.9531e-003	55	0	+
F_3	1.9531e-003	55	0	+	1.9531e-003	55	0	+	1.9531e-003	55	0	+
F_4	1.9531e-003	55	0	+	9.7656e-003	52	3	+	1.9531e-003	55	0	+
F_5	1.9531e-003	55	0	+	5.8594e-003	53	2	+	1.9531e-003	55	0	+
F_6	1	0	0	=	1	1	0	=	1	0	0	=
F_7	1.9531e-003	55	0	+	1.9531e-003	55	0	+	1	0	0	=
F_8	1	0	0	=	1	1	0	=	1	0	0	=
Total (+/–/=)			6/0/2				6/0/2				5/0/3	

5 Model the NO_X emissions based on the I-TLBO algorithm and PELM

With the development of coal-fired power stations, combustion technology of low NO_X emission has become an important research direction of boiler engineering. In order to reduce pollutant emissions, NO_X emissions model firstly needs to be built. The NO_X emissions model depends on various operating parameters, such as air velocity, coal feeder rate, oxygen content in the flue gas, exhaust gas temperature. However, due to the complexity, uncertainty, strong coupling, and the nonlinearity of the combustion process, it is difficult to use the theory of thermodynamics to model the NO_X emissions. The PELM is a new intelligent modeling tool based on the history combustion data of a boiler. In the PELM, the input weights and thresholds of hidden layers in the nonlinear part are randomly generated. To improve the generalization performance of the PELM, it is necessary to select the optimal input weights and hidden thresholds. Therefore, we select proposed I-TLBO algorithm to optimize the PELM and build I-TLBO-PELM model of NO_X emissions.

5.1 An introduction of the data

There are 240 data samples collected from a boiler, which are sampled once every 30 s and listed in Tab. 6. 100% load, 75% load and 50% load have 80 samples, respectively. In this study, the total 240 cases are divided into two parts: 192 cases (64 for 100% load, 64 for 75% load, and 64 for 50% load) as the training data, and the remaining 48 cases as the testing data. The NO_X emission is as the output variable, and 20 operational conditions closely related to the output variable are as the input variables of the I-TLBO-PELM model. The 20 operational conditions are given as follows.

The boiler load (%);

The fluid bed temperature (FBT, ^oC);

The coal feeder rate (CFR, th^{-1}), including A level, B level, C level and D level;

The primary air velocity (PAV, kNm^3h^{-1}), including left level and right level;

The primary air temperature (PAT, ^oC), including left level and right level;

The secondary air velocity (SAV, kNm^3h^{-1}), including left and inside level, left and outside level, right and inside level, right and inside level.

Table 6: The boiler operating conditions

Data	load (%)	FBT (^oC)	CFR(th^{-1})				PAV(kNm^3h^{-1})		PAT(^oC)	
			A	B	C	D	Left	Right	Left	Right
1	54.008	877.622	28.466	28.756	28.534	28.474	186.985	136.176	249.287	247.397
2	54.008	877.622	28.466	28.756	28.534	28.474	237.336	176.915	249.287	247.397
3	54.117	877.962	28.932	28.892	28.95	28.831	183.552	238.48	248.786	247.397
4	53.973	878.422	29.179	29.392	29.22	29.202	219.027	275.099	248.786	246.881
…					…					…
81	71.845	868.626	52.009	44.378	20.031	54.715	222.46	201.404	262.958	260.565
82	71.845	868.626	52.009	44.378	20.031	54.715	249.237	331.401	262.958	260.565
83	71.837	867.801	51.894	44.157	19.935	54.396	260.909	259.078	262.958	260.565
84	71.806	867.912	51.759	44.026	20.013	53.687	212.389	255.417	262.958	260.565
…										
161	101.28	867.59	61.435	55.491	55.379	61.098	414.709	311.718	281.339	278.605
162	101.28	867.59	61.435	55.491	55.379	61.198	377.174	288.145	281.339	278.605
163	101.066	865.975	61.435	55.394	55.671	61.35	388.389	268.462	281.339	278.605
164	100.616	864.498	61.082	55.085	55.29	61.178	343.531	333.232	281.339	278.605
…					…					…
240	101.382	875.358	59.004	53.164	53.136	59.168	242.371	318.813	283.886	281.819

Continued

Data	SAV(kNm^3h^{-1})				SAT(^oC)		EPFA(A)		OC	EGT	NO_X
	A	B	C	D	Left	Right	Left	Right	(%)	(^oC)	$MgNm^{-3}$
1	44.126	41.367	41.164	55.661	258.233	250.204	88.419	102.685	8.647	147.803	155.02
2	44.126	41.238	40.975	55.156	258.233	250.204	88.648	102.304	8.647	147.803	155.02
3	45.103	40.938	41.704	53.785	258.233	250.204	88.763	101.922	8.647	147.803	154.257
4	46.71	40.31	43.925	54.364	257.686	249.674	88.877	100.587	8.647	147.803	157.309
…						...					…
81	72.022	77.484	76.963	57.798	272.063	261.516	86.245	106.004	5.96	148.741	155.783
82	71.34	78.664	76.748	59.223	272.063	261.516	85.825	106.652	5.96	148.741	155.783
83	74.389	82.61	77.033	57.479	272.063	261.516	86.321	105.012	5.96	148.741	155.859
84	71.15	78.514	78.764	57.081	272.063	261.516	85.864	104.02	5.96	148.741	156.469
…						…					…
161	101.937	104.988	108.895	87.504	292.292	276.819	122.94	136.291	4.734	160.848	176.381
162	101.481	106.241	108.735	88.814	292.292	276.819	125.763	134.384	4.734	160.848	176.381
163	101.089	93.541	108.998	85.024	292.292	276.819	126.03	137.321	4.734	160.848	181.492
164	101.131	99.22	107.843	87.112	292.292	276.819	125.763	134.04	4.734	160.848	183.705
…						…					…
240	99.793	110.157	110.716	88.392	294.575	279.506	104.135	0.153	3.883	165.677	131.523

The secondary air temperature (SAT, ^oC), including left level and right level;

The electricity of powder feeding machine (EPFM, A), including A level and B level;

The oxygen content in the flue gas (OC, %);

The exhaust gas temperature (EGT, ^oC).

5.2 Model the NO_X emissions by using the I-TLBO-PELM

According to Caminhas et al. [Caminhas, Vieira and Vasconcelos (2003); Tavares, Saldanha and Vieira (2015)], although the PELM requires much smaller number of hidden neurons than other ANNs, it can still obtain the good generalization capability and approximation performance. After a lot of experiments, 3 neural nodes are set in the PELM. The activation function is sigmoid function $g(x)=1/1+e^{-x}$. The maximum iteration number of six optimization algorithms is set as 100. All other parameter settings are the same as those of the Fourth part. The optimization object function is the root mean square error (RMSE) on the training data as follows:

$$f(\eta)=\sqrt{\frac{\sum_{l=1}^{N}\left\|\sum_{j=1}^{m}\left[\sum_{i=1}^{n+1}u_{ji}x_{il}\phi\left(\sum_{i=1}^{n+1}v_{ji}x_{il}\right)\right]-y_l\right\|_2^2}{N}} \tag{20}$$

$$\eta=\left[v_{11},v_{12},\cdots,v_{1n},v_{1,n+1},v_{21},v_{22},\cdots,v_{2n},v_{2,n+1},v_{m1},v_{m2},\cdots,v_{mn},v_{m,n+1}\right]$$

where N is the number of the training data and $v_{ji} \in [-1,1]$.

The detailed modeling process by using the I-TLBO-PELM is summarized in the following steps.

Step 1: Set the control parameters of the I-TLBO algorithm, such as the size of the population M, the maximum number of iterations G, the values of w_{start} and w_{end};

Step 2: Initialize the population based on the quantum mechanism and cube chaos mapping. The population is composed of M individuals;

Step 3: Evaluating the fitness $f(\boldsymbol{\eta})$ of each individual $\boldsymbol{\eta}$ by using Eq. (20);

Step 4: Optimize $\boldsymbol{\eta}$ based on the teacher phase, student phase and self-learning phase;

Step 5: If the maximum number of iterations M is reached, the I-TLBO algorithm is stopped; otherwise, renew the population of the I-TLBO algorithm and the iteration is repeated from Step 4;

Step 6: The optimal $\boldsymbol{\eta}$ is obtained, and then $\boldsymbol{\eta}$ is substituted in Eq. (5). We can obtain the optimal PELM model and then the model is applied to predict the NO_X emissions on the testing data.

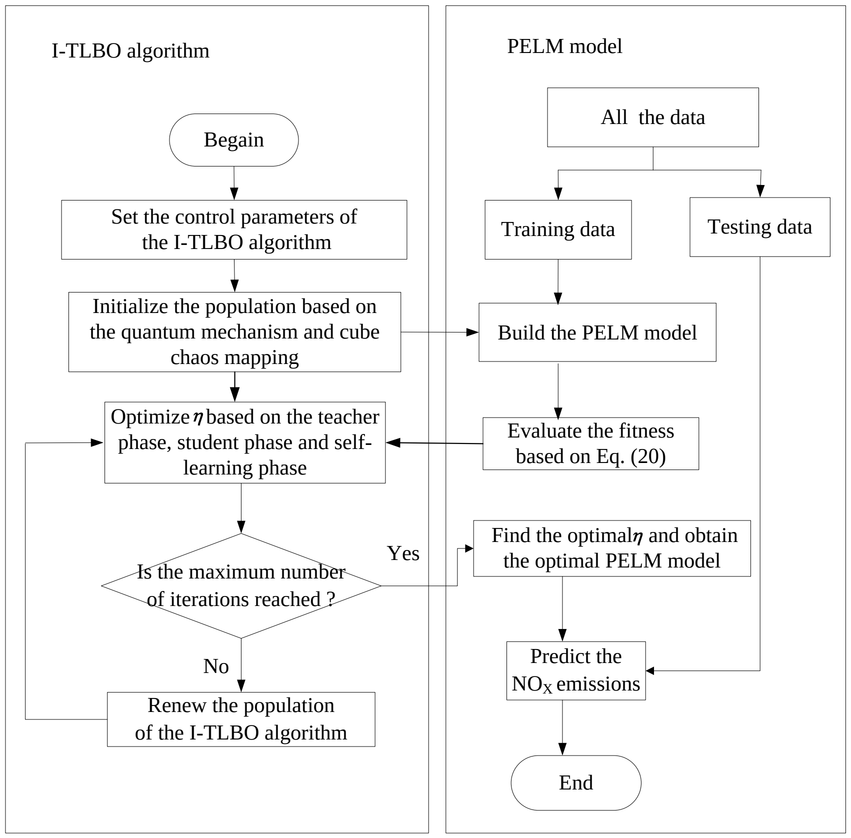

I-TLBO algorithm
PELM model
Begain
Set the control parameters of the I-TLBO algorithm
Initialize the population based on the quantum mechanism and cube chaos mapping
Optimize η based on the teacher phase, student phase and self-learning phase
Is the maximum number of iterations reached ?
Yes
No
Renew the population of the I-TLBO algorithm
All the data
Training data
Testing data
Build the PELM model
Evaluate the fitness based on Eq. (20)
Find the optimal η and obtain the optimal PELM model
Predict the NO_X emissions
End

Figure 6: The modeling process of the I-TLBO-PELM

For a more vivid observation, the realization flowchart of the proposed I-TLBO-PELM is shown in Fig. 6.

In addition, RMSE, mean absolute error (MAE) and mean absolute percentage error (MAPE) are used to evaluate the performance of the I-TLBO-PELM model.We compare with the PELM model, the PSO-PELM model, the GWO-PELM model, the TLBO-PELM model, the mTLBO-PELM model and the C-TLBO-PELM model. Additionally, in order to test the superiority of the I-TLBO-PELM model, we also compare with BP neural network and linear regression (LR). Although there are many variants of BP algorithm, a faster BP algorithm called Levenberg-Marquardt algorithm provided by MATLAB package is used in our simulations. The comparison results of the training data and the testing data are given in Tabs. 7-8 and Figs. 7-8.

5.3 The experiment results and analysis

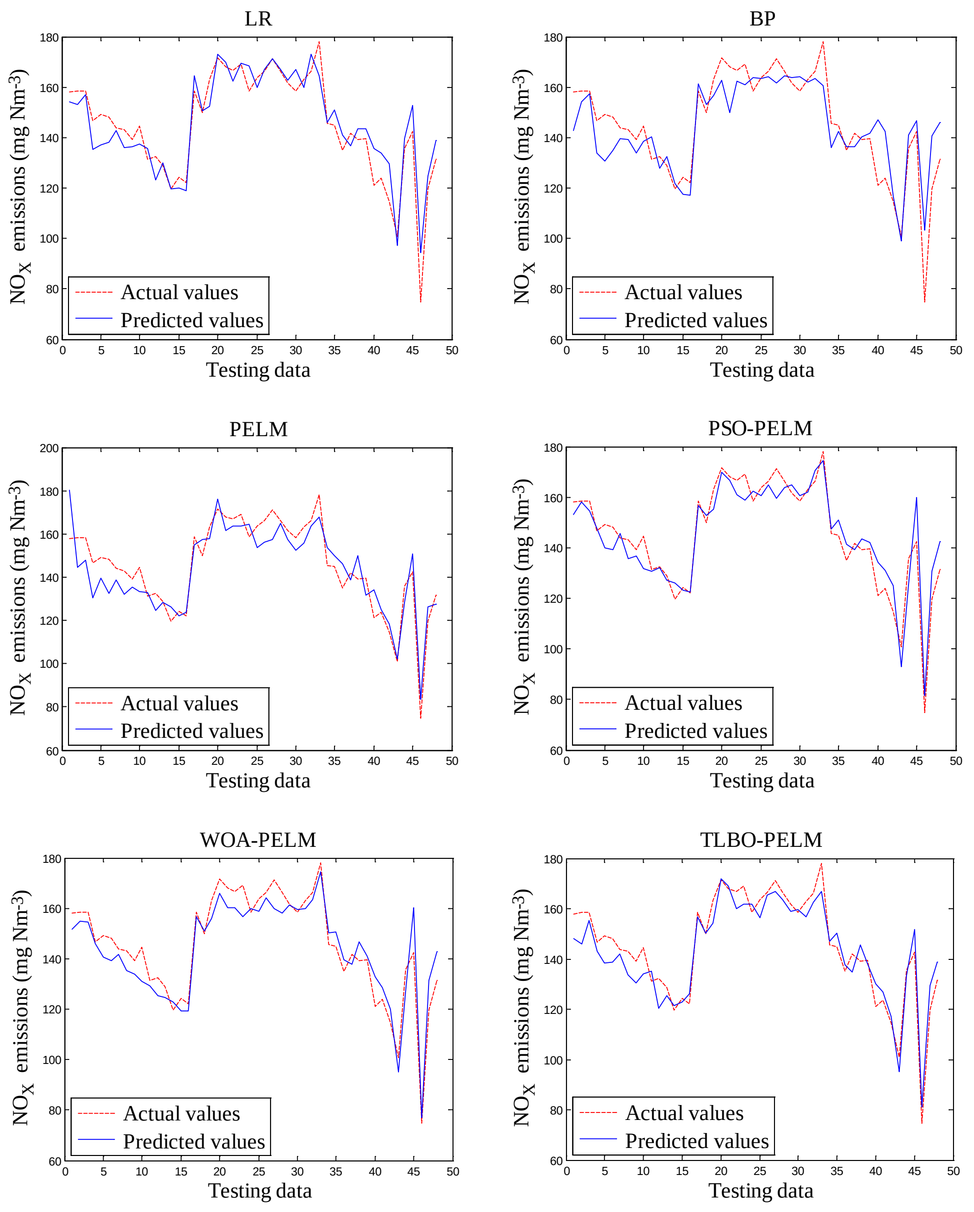
As shown in Tab. 7, for the I-TLBO-PELM model, the RMSE is 4.1763, the MAE is 3.1867, and the MAPE is 0.0228. Three performance indexes of the I-TLBO-PELM are smaller than those of eight other models. Therefore, the approximate ability of the I-TLBO-PELM is the best in nine models.

For the testing data, for the I-TLBO-PELM model in Tab. 7, we can see that the RMSE is 4.9068, the MAE is 3.9174, and the MAPE is 0.0276. Every performance index is also the smallest in nine models. So, the I-TLBO-PELM model has the best generalization and predicted performance in nine models.

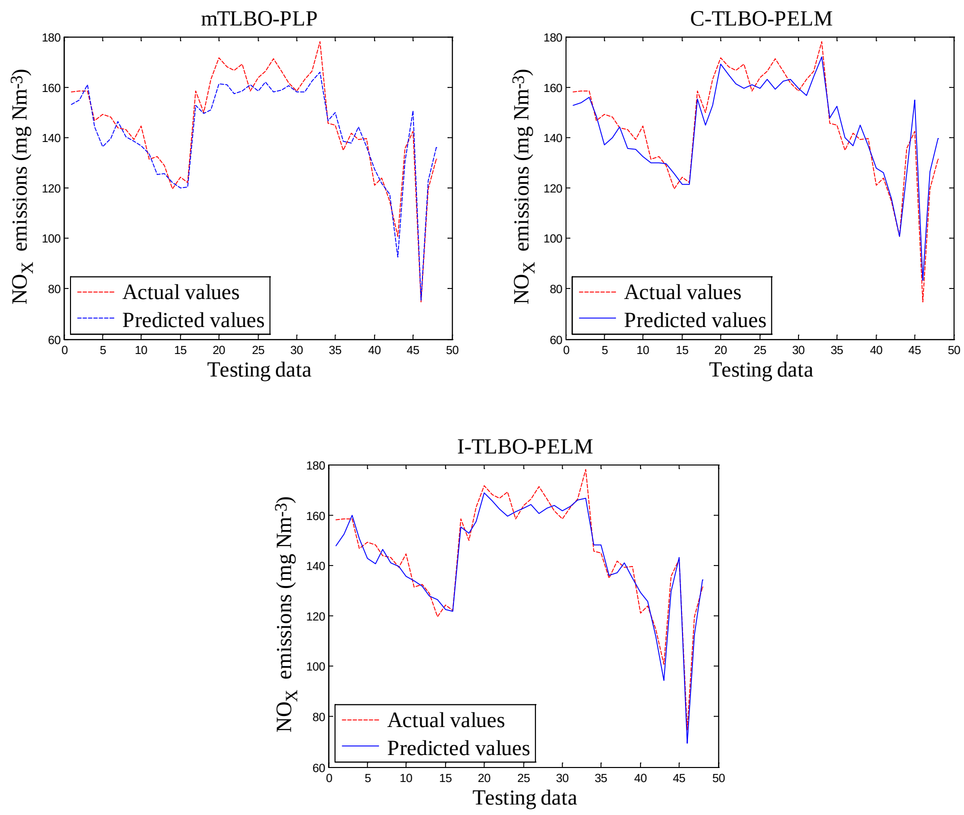
Table 7: Performance comparison for the NO_X emissions

Model	Training data			Testing data		
	RMSE	MAE	MAPE	RMSE	MAE	MAPE
LR	6.3332	4.6865	0.0332	7.2788	5.7064	0.0429
BP	7.5347	5.3627	0.0396	10.1224	7.4826	0.0562
PELM	7.3198	5.5429	0.0389	8.5144	7.2308	0.0502
PSO-PELM	4.4477	3.3120	0.0236	6.6290	5.2202	0.0381
GWO-PELM	4.4278	3.2559	0.0232	6.8367	5.7148	0.0403
TLBO-PELM	5.2355	4.0177	0.0286	6.3604	5.3264	0.0378
mTLBO-PELM	4.6954	3.5044	0.0249	6.0795	5.0516	0.0343
C-TLBO-PELM	4.4409	3.2897	0.0233	6.0324	4.9660	0.0350
I-TLBO-PELM	4.1763	3.1867	0.0228	4.9068	3.9174	0.0276

Fig. 7 shows predicted NO_X emissions of nine models on the testing data. As seen from it, the predicted values of the I-TLBO-PELM model are very closer to the actual values than eight other models. The relative errors of the testing data are given in Fig. 8. The range of the error fluctuation of the I-TLBO-PELM model is about in [-0.06, 0.06]. The ranges of eight other models are larger than the I-TLBO-PELM. Therefore, the I-TLBO-PELM is more accurate for modeling the NO_X emissions of the boiler than eight other models.

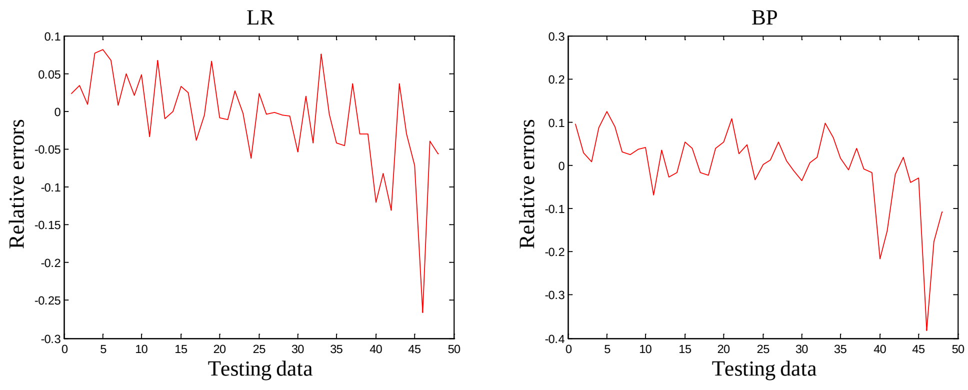

LR
BP
PELM
PSO-PELM
WOA-PELM
TLBO-PELM
NOX emissions (mg Nm-3)
Testing data
Actual values
Predicted values

(a)

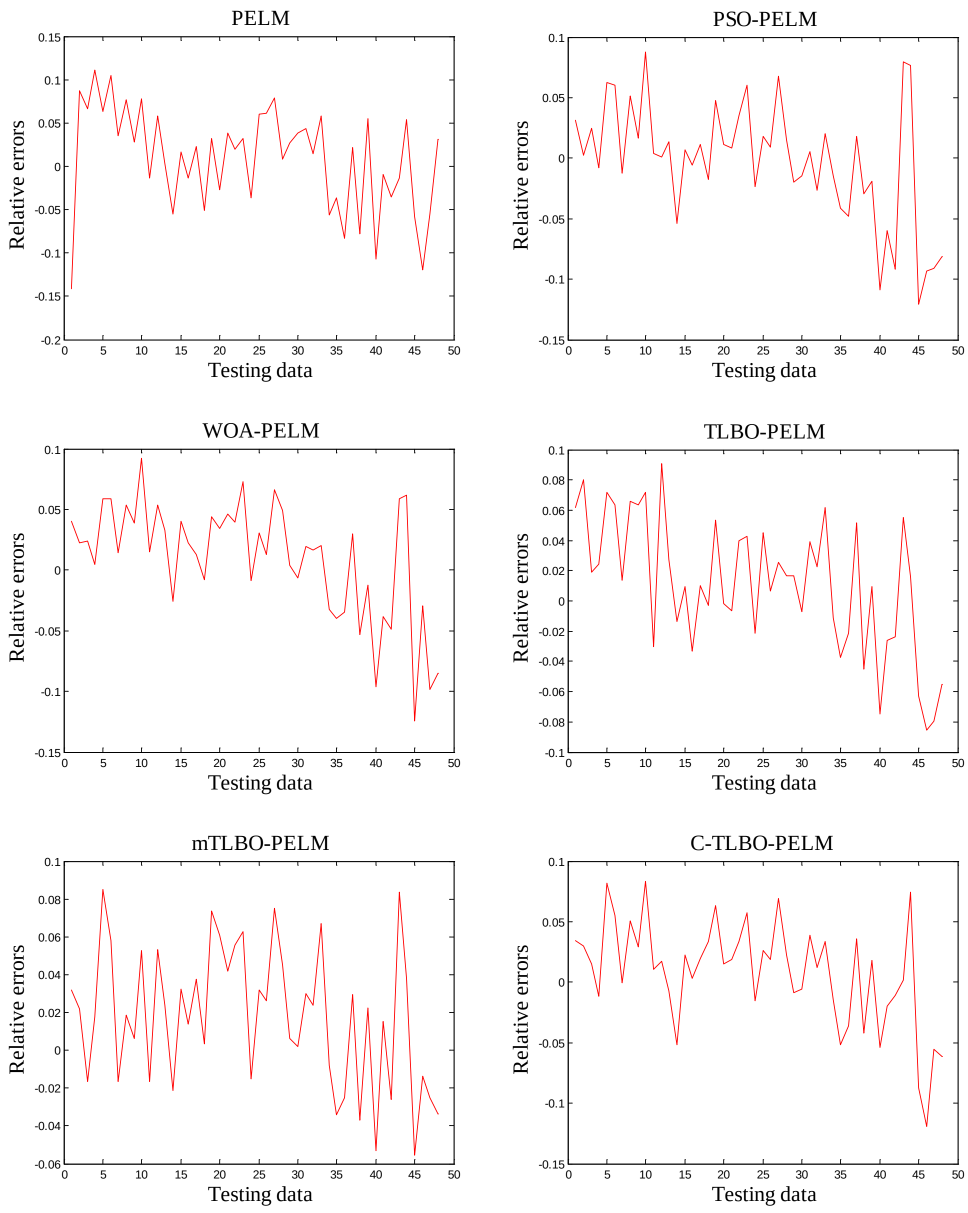


(b)

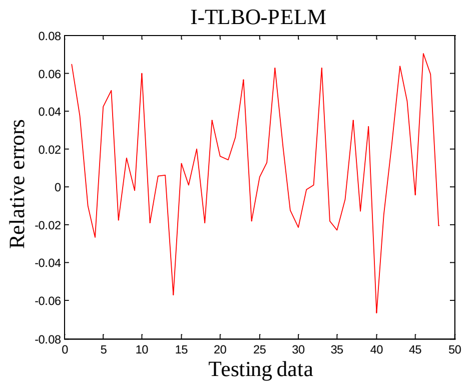
Figure 7: Predicted NO_X emissions of nine models on testing data



(a)


PELM
PSO-PELM
WOA-PELM
TLBO-PELM
mTLBO-PELM
C-TLBO-PELM
Relative errors
Testing data

(b)



(c)

Figure 8: Relative errors of nine models on testing data

Moreover, in order to further verify that I-TLBO-PELM is a better modeling tool under different experiments conditions, 5% white noise is added into the target attribute of the data. For noise-added data, the parameters of employed methods are the same as the data without noise in order to well compare the robustness of all models to perturbations [Li and Niu (2013)]. The comparison results of the training data and the testing data are shown in Tab. 8.

Table 8: Performance comparison for the NO_X emissions under the condition of 5% noise

Model	Training data			Testing data		
	RMSE	MAE	MAPE	RMSE	MAE	MAPE
LR	7.4938	5.9615	0.0423	8.1511	6.6380	0.0499
BP	8.3494	6.5325	0.0482	8.5632	7.4047	0.0536
PELM	7.2543	5.9055	0.0421	7.3135	5.9223	0.0412
PSO-PELM	6.4958	5.2664	0.0370	7.1598	6.0314	0.0439
GWO-PELM	6.0736	5.0893	0.0359	7.6978	6.5813	0.0463
TLBO-PELM	6.3924	5.1808	0.0365	7.1871	5.7742	0.0416
mTLBO-PELM	6.7622	5.3553	0.0377	6.8791	5.3532	0.0376
C-TLBO-PELM	6.3552	5.0709	0.0358	7.5353	6.1594	0.0450
I-TLBO-PELM	5.8408	4.8062	0.0336	6.3999	5.3556	0.0390

As seen from Tab. 8, whether on the training data or on the testing data, every performance index of the I-TLBO-PELM model is also the smallest in all nine models. It shows that the I-TLBO-PELM model owns a good robustness and can competently reduce the adverse effects which are caused by the perturbation.

In summarize, the proposed I-TLBO-PELM model is relative accurate. This model has very good approximate ability, generalization performance and robustness. So, the I-TLBO-PELM can provide an effective method to predict the NO_X emissions of the boiler working.

6 Conclusions

In order to improve the exploration ability and exploitation performance of the TLBO algorithm, an I-TLBO algorithm is proposed. In the I-TLBO, there are four improvements. Firstly, the quantum initialized population based on qubits replaces the randomly initialized population. Secondly, two kinds of angles in Bloch sphere are generated by using the cube chaos mapping. Thirdly, an adaptive control parameter is added into the teacher phase. And then, a self-learning process by using Gauss mutation is proposed after the learner phase. The optimization performance of the I-TLBO algorithm is verified on eight classical optimization functions. The experimental results show that the I-TLBO algorithm not only can jump out of the local optimal solution so as to achieve the global optimal solution, but also the convergent speed is the fastest in the six algorithms. Finally, a hybrid I-TLBO-PELM model is built to predict NO_X emissions of a boiler. Compared with eight other models, the I-TLBO-PELM model has good regression precision and generalization performance. Therefore, the I-TLBO-PELM model is more suitable to predict the NO_X emissions of the boiler. In the future, the I-TLBO algorithm and the I-TLBO-PELM model will be used to make combustion optimization of the boiler to reduce the NO_X emissions.

Acknowledgement: The authors would also like to acknowledge the valuable comments and suggestions from the Editors and Reviewers, which vastly contributed to improve the presentation of the paper. This work is supported by the National Natural Science Foundations of China (61573306 and 61403331), 2018 Qinhuangdao City Social Science Development Research Project (201807047 and 201807088), the Program for the Top Young Talents of Higher Learning Institutions of Hebei (BJ2017033) and the Marine Science Special Research Project of Hebei Normal University of Science and Technology (No. 2018HY021).

References

Bhattacharyya, B.; Babu, R. (2016): Teaching learning based optimization algorithm for reactive power planning. *International Journal of Electrical Power & Energy Systems*, vol. 81, pp. 248-253.

Caminhas, W. M.; Vieira, D. A. G.; Vasconcelos, J. A. (2003): Parallel layer perceptron. *Neurocomputing*, vol. 55, no. 3, pp. 771-778.

Coelho, L. D. S.; Mariani, V. C. (2008): Use of chaotic sequences in a biologically inspired algorithm for engineering design optimization. *Expert Systems with Applications*, vol. 34, no. 3, pp. 1905-1913.

Gandomi, A. H.; Yang, X. S. (2014): Chaotic bat algorithm. *Journal of Computational Science*, vol. 5, no. 2, pp. 224-232.

Ilamathi, P.; Selladurai, V.; Balamurugan, K.; Sathyanathan, V. T. (2013): ANN-GA approach for predictive modeling and optimization of NOx emission in a tangentially fired boiler. *Clean Technologies & Environmental Policy*, vol. 15, no. 1, pp. 125-131.

Huo, F. C.; Liu, Y.; Wang, D.; Sun, B. X. (2017): Bloch quantum artificial bee colony algorithm and its application in image threshold segmentation. *Signal Image and Video Processing*, vol. 11, no. 8, pp. 1585-1592.

Huang, G. B.; Zhou, H.; Ding, X.; Zhang, R. (2012): Extreme learning machine for regression and multiclass classification. *IEEE Transactions on Systems Man & Cybernetics Part B*, vol. 42, no. 2, pp. 513-529.

Huang, G. B.; Zhu, Q. Y.; Siew, C. K. (2006): Extreme learning machine: theory and applications. *Neurocomputing*, vol. 70, no. 1-3, pp. 489-501.

Jothiprakash, V.; Arunkumar, R. (2013): Optimization of hydropower reservoir using evolutionary algorithms coupled with chaos. *Water Resources Management*, vol. 27, pp. 1963-1979.

Kennedy, J.; Eberhart, R. C. (1995): Particle swarm optimization. *Proceedings of IEEE International Conference on Neural Networks*, pp. 1942-1948.

Li, G. Q.; Niu, P. F. (2013): An enhanced extreme learning machine based on ridge regression for regression. *Neural Computing and Applications*, vol. 22, pp. 803-810.

Li, P. (2014): A quantum-behaved evolutionary algorithm based on the bloch spherical search. *Communications in Nonlinear Science & Numerical Simulation*, vol. 19, no. 4, pp. 763-771.

Mirjalili, S. (2016): Sca: A sine cosine algorithm for solving optimization problems. *Knowledge-Based Systems*, vol. 96, pp. 120-133.

Mirjalili, S.; Mirjalili, S. M.; Lewis, A. (2014): Grey wolf optimizer. *Advances in Engineering Software*, vol. 69, no. 3, pp. 46-61.

Ouyang, B.; Kong, X. Y. (2014): Teaching-learning based optimization algorithm with crossover operation. *Journal of Northeastern University (Natural Science)*, vol. 35, no. 3, pp. 323-328.

Rashedi, E.; Nezamabadi-Pour, H.; Saryazdi, S. (2009): GSA: A gravitational search algorithm. *Information Sciences*, vol. 179, no. 13, pp. 2232-2248.

Rao, R. V.; Savsani, V. J.; Vakharia, D. P. (2011): Teaching-learning-based optimization: A novel method for constrained mechanical design optimization problems. *Computer-Aided Design*, vol. 43, pp. 303-315.

Rao, R. V.; Savsani, V. J.; Vakharia, D. P. (2012): Teaching-learning-based optimization: an optimization method for continuous non-linear large scale problems. *Information Sciences*, vol. 183, pp. 1-15.

Rao, R. V.; Kalyankar, V. D. (2013): Parameter optimization of modern machining processes using teaching-learning-based optimization algorithm. *Engineering Applications of Artificial Intelligence*, vol. 26, no. 1, pp. 524-531.

Pawar, P. J.; Rao, R. V. (2013): Parameter optimization of machining processes using teaching-learning-based optimization algorithm. *International Journal of Advanced Manufacturing Technology*, vol. 67, no. 5-8, pp. 995-1006.

Satapathy, S. C.; Naik, A. (2013): A modified teaching-learning-based optimization (mTLBO) for global search. *Recent Patents on Computer Science*, vol. 6, pp. 60-72.

Tavares, L. D.; Saldanha, R. R.; Vieira, D. A. G. (2015): Extreme learning machine with parallel layer perceptrons. *Neurocomputing*, vol. 166, pp. 164-171.

Vedat, T.; Ayşe, T. D. (2008): An improved genetic algorithm with initial population strategy and self-adaptive member grouping. *Computers & Structures*, vol. 86, no. 11-12, pp. 1204-1218.

Wang, Z.; Yan, M. (2011): Online adaptive least squares support vector machine and its application in utility boiler combustion optimization systems. *Journal of Process Control*, vol. 21, no. 7, pp. 1040-1048.

Xin, Y.; Liu, Y.; Lin, G. M. (2002): Evolutionary programming made faster. *IEEE Transactions on Evolutionary Computation*, vol. 3, no. 2, pp. 82-102.

Zhou, H.; Cen, K.; Fan, J. (2004): Modeling and optimization of the NOx emission characteristics of a tangentially fired boiler with artificial neural networks. *Energy*, vol. 29, no. 1, pp. 167-183.

Zhou, Y.; Liu, P. Y.; Zhao, J.; Wang, Q. L. (2012): Chaos particle swarm optimization based on the adaptive inertia weight. *Journal of Shandong University (Natural Science)*, vol. 47, no. 3, pp. 27-32.