Knowledge-Based Systems 187 (2020) 104836

Contents lists available at ScienceDirect

Knowledge-Based Systems

journal homepage: www.elsevier.com/locate/knosys

Hybrid teaching–learning-based optimization and neural network algorithm for engineering design optimization problems☆

Yiying Zhang [a], Zhigang Jin [a,*], Ye Chen [a,b]

[a] School of Electrical and Information Engineering, Tianjin University, Tianjin 300072, PR China
[b] College of Applied Science and Technology, Hainan University, Haikou 570228, PR China

HIGHLIGHTS

- A novel hybrid algorithm called TLNNA is proposed based on TLBO and NNA, which is an algorithm without any effort for fine tuning initial parameters.
- TLNNA has excellent global optimization ability of NNA and fast convergence rate of TLBO by the designed dynamic grouping mechanism.
- TLNNA is examined using 30 well-known unconstrained benchmark test functions and 4 challenging constrained engineering design problems.

ARTICLE INFO

Article history:
Received 21 December 2018
Received in revised form 4 July 2019
Accepted 6 July 2019
Available online 15 July 2019

Keywords:
Neural network algorithm
Artificial neural networks
Teaching–learning-based optimization
Engineering optimization

ABSTRACT

Neural network algorithm (NNA) is one of the newest meta-heuristic algorithms, which is inspired by biological nervous systems and artificial neural networks. Benefiting from the unique structure of artificial neural networks, NNA has good global search ability. However, slow convergence is its drawback that restricts its practical application. Teaching–learning-based optimization (TLBO) is an algorithm without any effort for fine tuning initial parameters, which has fast convergence speed while it is easy to fall into local optimum in solving complex global optimization problems. Considering the features of NNA and TLBO, an effective hybrid method based on TLBO and NNA, named TLNNA, is proposed for solving engineering optimization problems. The performance of TLNNA for 30 well-known unconstrained benchmark functions and 4 challenging engineering optimization problems is examined and the optimization results are compared with other competitive meta-heuristic algorithms. Such comparisons suggest that TLNNA has not only good global search ability of NNA but also fast convergence speed of TLBO and is more successful for most test problems in terms of solution quality and computational efficiency.

1. Introduction

Engineering design can be described as a goal-oriented, constrained, decision making process to create products that satisfy well-defined human needs [1]. Generally, a complete constrained engineering design optimization problem usually includes the following components: a certain goal (objective function), a search space (feasible solutions), some constraints (constrained conditions) and a search process (optimization techniques). Mathematically, it can be defined as

$$\begin{aligned} &\min/\max f\ (\boldsymbol{x}) \\ &\text{s.t.}\quad c_i(\boldsymbol{x}) = 0,\ i = 1, 2, \ldots, p, \\ &\qquad c_j(\boldsymbol{x}) \le 0,\ j = 1, 2, \ldots, q, \\ &\qquad \boldsymbol{l}_k \le \boldsymbol{x}_k \le \boldsymbol{u}_k,\ k = 1, 2, \ldots, D. \end{aligned} \tag{1}$$

where $\boldsymbol{x}$ is the set of all solutions including all possible values of the design variables, $f(\boldsymbol{x})$ denotes the objective function, D means the number of design variables, $c_i(\boldsymbol{x}) = 0$ are the ith equality constraints, $c_j(\boldsymbol{x}) \le 0$ are the jth inequality constraints, p and q are the number of equality constraints and inequality constraints, $\boldsymbol{l}_k$ and $\boldsymbol{u}_k$ are the lower bound and the upper bound of the decision variable $\boldsymbol{x}_k$. In general, the real design problems are very complicated, which can be reflected the following three aspects: (1) a large number of design variables and constraints need to be considered; (2) the objective functions can be related to

☆ One or more of the authors of this paper have disclosed potential or pertinent conflicts of interest, which may include receipt of payment, either direct or indirect, institutional support, or association with an entity in the biomedical field which may be perceived to have potential conflict of interest with this work. For full disclosure statements refer to https://doi.org/10.1016/j.knosys.2019.07.007.

* Corresponding author.
E-mail address: zgjin@tju.edu.cn (Z. Jin).

https://doi.org/10.1016/j.knosys.2019.07.007
0950-7051/

multiple design variables and have the non-linear characteristics; (3) the objective functions usually have a multi-peak distribution, which have many local optimal solutions and one global optimal solution.

Optimization methods for engineering design problems can be broadly divided into two major categories, i.e., deterministic methods and meta-heuristic methods. Deterministic methods are sensitive to initial values and it can be as an available option for solving the simple and ideal optimization problems while they generally fail to solve complex engineering design problems [2,3]. Compared with deterministic methods, meta-heuristic methods are new techniques for solving the engineering optimization problems and these methods have the following characteristics: using a population of individuals in the search space, using an objective function instead of function derivatives and other related knowledge, and using probabilistic rather than deterministic rules [3]. These characteristics help meta-heuristic methods have more chance to find the better solutions in solving engineering optimization problems compared to deterministic methods. The existing meta-heuristic algorithms broadly include the following categories in terms of the inspiration source:

(1) Evolutionary algorithms. These algorithms are inspired by process of biological evolution, such as different evolution (DE) [4] and biogeography-based optimization (BBO) [5].
(2) Swarm intelligence algorithms. This kind of algorithms usually are inspired by some behaviors of animals or plants, such as cuckoo search (CS) [6], grey wolf optimization (GWO) [7], whale optimization algorithm (WOA) [8], salp swarm algorithm (SSA) [9] and particle swarm optimization (PSO) [10].
(3) Human-related algorithms. These algorithms are inspired by some human activities, such as sine cosine algorithm (SCA) [11], passing vehicle search (PVS) [12] and teaching–learning based optimization (TLBO) [2].
(4) Physics-based algorithms. This type of algorithms imitate some physical phenomenon, such as water cycle algorithm (WCA) [13] and water evaporation optimization (WEO) [14].

Here, it is worth mentioning that meta-heuristic algorithms also can be divided according to their parameters. In general, the parameters of the existing meta-heuristic algorithms include common parameters and special parameters. Common parameters are essential for each meta-heuristic algorithm, such as population size and stopping criteria. As for special parameters, they are the control parameters reflecting the features of algorithms, which are required for most existing meta-heuristic algorithms, such as the crossover probability and mutation probability in DE, the discover probability in CS, the number of rivers and sea in WCA, learning coefficient in PSO and migration rate in BBO. The challenging problems for meta-heuristic algorithms with special parameters can be summarized as follows: (1) it is difficult for these algorithms to select appropriate special parameters in handling an optimization task; (2) the appropriate special parameters corresponding to the different optimization tasks usually are not the same. Thus it is very practically significant to propose new meta-heuristic algorithms without special parameters.

NNA and TLBO are two of a few meta-heuristic algorithms without special parameters. The two algorithms have their own characteristics, which can be described as follows.

As the basis of artificial intelligence, neural network technique has been growing very fast in recent years, which has been used to solve many practical problems [15–27]. Interestingly, Sadollah et al. [28] proposed a novel meta-heuristic method called neural network algorithm (NNA) in 2018, which shows neural network technique can be used to design optimization algorithm. As one of the newest meta-heuristic algorithms, NNA is inspired by artificial neural networks (ANNs) and biological nervous systems. Benefiting from the unique structure of ANNs, NNA can find the global optimum solution with the minimum possibility of getting trapped in a local optimum compared with many existing meta-heuristic algorithms and shows strong global exploration ability [28]. However, NNA has a notable drawback that it has a slow convergence speed. This defect will restrict its applications in some optimization problems with limitations like computation resource constraints.

TLBO has been developed by Rao et al. in 2011 [1,2], which is inspired by the traditional classroom teaching phenomenon. TLBO is an algorithm without any effort for fine tuning initial parameters, which has fast convergence speed. Considering these advantages, TLBO has been successfully applied to a lot of real-world optimization problems [29–34]. However, TLBO is easy to fall into a local optimum in solving complex global optimization problems [35,36].

Motivated by the characteristics of TLBO and NNA, this paper presents an effective hybrid algorithm based on teaching–learning-based optimization and neural network algorithm, named TLNNA, for solving complex engineering optimization problems. The main contribution of this paper can be stated as follows:

(1) A novel optimization technique called TLNNA is proposed based on TLBO and NNA, which is an algorithm without any effort for fine tuning initial parameters.
(2) TLNNA makes the best of the excellent global optimization ability of NNA and the fast convergence rate of TLBO by dynamic grouping mechanism.
(3) TLNNA is examined using 30 well-known unconstrained benchmark test functions and 4 challenging constrained engineering design problems.

The rest of this paper is organized as follows: Section 2 gives a brief overview of the original NNA and TLBO. Section 3 elaborates our proposed TLNNA and its technical details. The experiments results are illustrated in Section 4. Section 5 presents the conclusions and future work.

2. Related work

2.1. Neural network algorithm

NNA is a population-based optimization algorithm and its basis is ANNs. ANNs are used for prediction purposes in most cases and it tries to close the gap among the predicted solution and the given target solution by frequently adjusting the values of weight functions. However, the goal of the optimization algorithm is to find an optimum solution from the given search space of the optimization problem. In order to adapt ANNs to be suitable for using as an optimization algorithm, NNA views the current best solution as the target solution and its goal is to close the gap between the target solution and the other solutions included in the population. NNA mainly includes the following four stages:

(1) Generate new population
In NNA, every individual i has its corresponding weight vector $\boldsymbol{w}_i^t = \left[w_{i,1}^t, w_{i,2}^t, \ldots, w_{i,N}^t\right]$ that can be defined as

$$\sum_{j=1}^{N} w_{i,j}^t = 1,\ 0 < w_{i,j}^t < 1,\ i = 1, 2, \ldots, N \tag{2}$$

where N is population size and t is the current number of iterations. Fig. 1 shows the process of the population generation, which can be described as

$$\boldsymbol{x}_{\text{new},\,j}^{t+1} = \sum_{i=1}^{N} w_{i,j}^{t} \times \boldsymbol{x}_{i}^{t}, \quad j = 1, 2, \ldots, N \tag{3}$$

$$\boldsymbol{x}_{i}^{t+1} = \boldsymbol{x}_{i}^{t} + \boldsymbol{x}_{\text{new},i}^{t+1}, \qquad i = 1, 2, \ldots, N \tag{4}$$

where $\boldsymbol{x}_i^t$ is solution of the ith individual at time t and $\boldsymbol{x}_{\text{new},\,j}^{t+1}$ is the weighted solution of the jth individual at time $t+1$.

(2) Update the weight matrix
The weight matrix can be updated according to the following formulation:

$$\boldsymbol{w}_{i}^{t+1} = \boldsymbol{w}_{i}^{t} + 2 \times a \times \left(\boldsymbol{w}_{\text{opt}}^{t} - \boldsymbol{w}_{i}^{t}\right), \; i = 1, 2, \ldots, N \tag{5}$$

where the parameter a is a random number between 0 and 1 and $\boldsymbol{w}_{\text{opt}}^t$ is the target weight vector. The target weight vector $\boldsymbol{w}_{\text{opt}}^t$ and the target solution $\boldsymbol{x}_{\text{opt}}^t$ are updated simultaneously. If $\boldsymbol{x}_{\text{opt}}^t$ is equal to $\boldsymbol{x}_k^t$ ($k \in [1, N]$) at time t, $\boldsymbol{w}_{\text{opt}}^t$ is equal to $\boldsymbol{w}_k^t$.

(3) Bias operator
In the bias operator, there is a key parameter called modification factor β, which is used to determine the bias proportion and is updated according to the Eq. (6) [28].

$$\beta^{t+1} = 0.99\beta^{t} \tag{6}$$

In NNA, bias operator includes two parts, namely bias population and bias weight matrix. As for bias population, a random number N_{p} is firstly generated, which is equal to $\lceil \beta^t \times D \rceil$. Then a set P is produced by randomly choosing N_{p} integers between 0 and D. Let $\boldsymbol{l} = (l_1, l_2, \ldots, l_N)$ and $\boldsymbol{u} = (u_1, u_2, \ldots, u_N)$ are the lower and the upper bounds of design variables. Thus the bias population can be described as

$$x_{i,P(s)}^{t} = l_{P(s)} + \left(u_{P(s)} - l_{P(s)}\right) \times b, \quad s = 1, 2, \ldots, N_{\text{p}} \tag{7}$$

where b is a random number between 0 and 1. As for bias weight matrix, a random number N_{w} is firstly generated, which is equal to $\lceil \beta^t \times N \rceil$. Then N_{w} integers between 0 and N selected randomly form a set Q. Thus the bias weight matrix can be defined as

$$\boldsymbol{w}_{i,P(r)}^{t} = c, \quad r = 1, 2, \ldots, N_{\text{w}} \tag{8}$$

where c is a random number between 0 and 1.

(4) Transfer function operator
In NNA, the transfer function transfers the current solution to a new position to generate a better solution toward the current best solution, which can be expressed as

$$\boldsymbol{x}_{i}^{t+1} = \boldsymbol{x}_{i}^{t+1} + 2 \times d \times \left(\boldsymbol{x}_{\text{opt}}^{t} - \boldsymbol{x}_{i}^{t+1}\right), \; i = 1, 2, \ldots, N \tag{9}$$

where d is a random number between 0 and 1.

Based on the above stages, Fig. 2 shows the pseudo code of NNA. From Fig. 2, we can clearly see that NNA is simple and easy to implement.

2.2. Teaching–learning-based optimization

TLBO simulates the traditional classroom teaching phenomenon, which includes teacher phase and learner phase. Fig. 3 gives the pseudo code of TLBO. In terms of the optimization algorithm, TLBO can be interpreted as follows: (1) each learner denotes a solution; (2) the result of each subject means an element of the solution; (3) all solutions constitute a population; (4) the population can find better solutions by multiple iterations of teacher phase and learner phase. Then teacher phase and learner phase are presented in the following subsections.

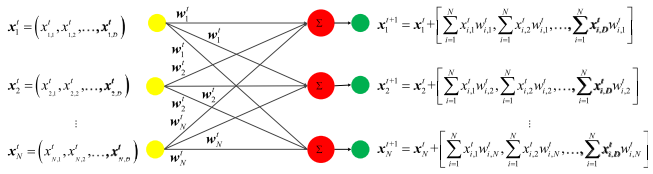

Fig. 1. The process of population generation in NNA.

2.2.1. Teacher phase

This phase means the learners gain their knowledge from their teacher. In this part, the teacher tries to increase the mean result of the whole class to his or her level. The difference between the level of the teacher and the mean result of the class can be defined as

$$\boldsymbol{M}^{t} = \frac{1}{N}\sum_{i=1}^{N} \boldsymbol{x}_{i}^{t} \tag{10}$$

$$\boldsymbol{F}_{\text{mean}}^{t} = g \times \left(\boldsymbol{x}_{\text{best}}^{t} - T_{\text{F}}\boldsymbol{M}^{t}\right), \quad j = 1, 2, \ldots, D \tag{11}$$

where g is a random number between 0 and 1, T_{F} is a heuristic step that can be 1 or 2, $\boldsymbol{M}^t$ is the mean result of the class at time t, $\boldsymbol{x}_{\text{best}}^t$ is the result of the best learner at time t and $\boldsymbol{F}_{\text{mean}}^t$ is the difference between the result of the teacher and the mean result of the class at time t.

Based on the $\boldsymbol{F}_{\text{mean}}^t$, the solutions of the population can be updated by the following formulation:

$$\boldsymbol{x}_{i}^{t+1} = \boldsymbol{x}_{i}^{t} + \boldsymbol{F}_{\text{mean}}^{t} \tag{12}$$

2.2.2. Learner phase

In addition to learning from the teacher, a learner also can gain his or her knowledge by discussing and interacting with other learners, which is called learner phase. This phase can be described as (take the minimum problem, for instance)

$$\boldsymbol{x}_{i}^{t+1} = \begin{cases} \boldsymbol{x}_{i}^{t} + m \times \left(\boldsymbol{x}_{i}^{t} - \boldsymbol{x}_{k}^{t}\right), & \text{if } f\left(\boldsymbol{x}_{i}^{t}\right) < f\left(\boldsymbol{x}_{k}^{t}\right), \quad k \neq i \\ \boldsymbol{x}_{i}^{t} + m \times \left(\boldsymbol{x}_{k}^{t} - \boldsymbol{x}_{i}^{t}\right), & \text{if } f\left(\boldsymbol{x}_{i}^{t}\right) > f\left(\boldsymbol{x}_{k}^{t}\right), \quad k \neq i \end{cases} \tag{13}$$

where m is a random number between 0 and 1 and $\boldsymbol{x}_k^t$ is the kth learner that is selected randomly.

3. The proposed TLNNA

This section is divided into two parts. The first part describes the design frame of the proposed method. The second part presents the implementation of TLNNA.

3.1. The design frame of the proposed TLNNA

As mentioned above, the proposed TLNNA is aiming at combining the fast convergence rate of TLBO and the good global search ability of NNA. Thus the difficulty is how to make the best of the advantages of NNA and TLBO. To solve this problem, the population is divided into two halves based on the fitness values of individuals. Then TLBO and NNA are performed on the best half of population and the worst half of population, respectively. The framework of the proposed TLNNA is shown in Fig. 4. From Fig. 4, this algorithm includes the following three core optimization stages:

(1) Dynamic grouping stage
In TLNNA, dynamic grouping means the population is divided into two halves based on the fitness values of individuals after each loop. The best half of population is

NNA Algorithm

1. **begin**
2. Initialize the weight matrix and the population including N solutions
3. Calculate the fitness value of each solution and then set the optimal solution and the optimal weight
4. **repeat**
5. Generate the new population $\boldsymbol{X}^{t+1}$ via Eqs. (3) and (4) and update the new weight matrix $\boldsymbol{W}^{t+1}$ via Eq.(5)
6. **for** each individual $i \in N$ **do**
7. **if** rand $\leq \beta^t$
8. Perform the bias operator for updating the solution $\boldsymbol{x}_i^{t+1}$ via Eq. (7) and the weight $\boldsymbol{w}_i^{t+1}$ via Eq. (8)
9. **else**
10. Perform the transfer function operator for updating the solution $\boldsymbol{x}_i^{t+1}$ via Eq.(9)
11. **end if**
12. **end for**
13. Generate the new modification factor β^{t+1} via Eq. (6)
14. Calculate the fitness value of each solution and find the optimal solution $\boldsymbol{x}_{\text{opt}}^{t+1}$ and the optimal weight $\boldsymbol{x}_{\text{opt}}^{t+1}$
15. **Until** (stop condition=false)
16. Post process results and visualization
17. **end**

Fig. 2. Pseudo code of standard NNA.

TLBO Algorithm

1. **begin**
2. Initialize the population including N solutions $\boldsymbol{x}_i^t$, $i = 1, 2, \ldots, N$
3. Calculate the fitness value of each solution and then select the optimal solution as the teacher $\boldsymbol{x}_{\text{best}}^t$
4. **repeat**
5. Calculate the mean result $\boldsymbol{M}^t$ according to Eq. (10)
6. **for** each individual $i \in N$ **do**
7. Perform the teacher phase according to Eqs. (11) and (12)
8. Calculate the fitness value of individual i
9. Select the best solution from new solution and old solution as the current solution
10. **end for**
11. **for** each individual $i \in N$ **do**
12. Perform the learner phase according to Eq. (13)
13. Calculate the fitness value of individual i
14. Select the best solution from new solution and old solution as the current solution
15. **end for**
16. Select the optimal solution as the teacher $\boldsymbol{x}_{\text{best}}^t$
17. **Until** (stop condition=false)
18. Post process results and visualization
19. **end**

Fig. 3. Pseudo code of TLBO.

optimized by TLBO and the worst half of population is processed by NNA. This stage is designed based on the following two reasons. Firstly, search operators with similar behavior may lead to the loss of diversity in the search space. Once the solutions get trapped in a local minimum, they are not easy to escape [37,38]. TLBO and NNA are two algorithms with different search operators. Thus TLNNA can enhance the capability of escaping from the local optimal by dynamic grouping stage. Secondly, in the search process, TLBO focus on the local search and NNA emphasizes on the global search, which can help TLNNA achieve an efficient balance between exploration and exploitation.

(2) TLBO stage

TLNNA uses TLBO to optimize the best half of population, which is mainly considered as follows. This stage makes the best of the fast convergence speed of the TLBO by optimizing the best half of population, which can accelerate greatly convergence of TLNNA.

(3) NNA stage

NNA is employed to process the worst half of population in TLNNA, which is based on the following considerations. Considering the excellent global exploration of NNA, this stage not only helps avoid the worst half of population into local optimal solutions to some extent but also makes it possible to find better global optimal solution.

3.2. The implementation of the proposed TLNNA

In this section, the implementation of the proposed TLNNA is introduced in detailed. The step-wise procedure for execution of TLNNA can be described as follows.

Step 1: Initialization information.

(1.1) Initialize parameters. The maximum number of function evaluations $N_{\max}$, population size N, the lower bounds of design variables $\boldsymbol{l}$, the upper bounds of design variables $\boldsymbol{u}$, dimension of problem D and fitness function $f(\cdot)$. The current of number iteration time t is 1, the modification factor β is 1, and the number of function evaluations N_{current} is set to 0.

(1.2) Initialize population. A random population $\boldsymbol{X}^t$ is generated on the basis of the initialization parameters, which can be described as

$$\boldsymbol{X}^t = \left[\boldsymbol{x}_1^t, \boldsymbol{x}_2^t, \ldots, \boldsymbol{x}_N^t\right]^{\mathrm{T}} = \begin{bmatrix} x_{1,1}^t & x_{1,2}^t & \cdots & x_{1,D}^t \\ x_{2,1}^t & x_{2,2}^t & \cdots & x_{2,D}^t \\ \vdots & \vdots & & \vdots \\ x_{N,1}^t & x_{N,2}^t & \cdots & x_{N,D}^t \end{bmatrix} \tag{14}$$

$$x_{i,j}^t = l_i + (u_i - l_i) \times k \tag{15}$$

where k is a random number between 0 and 1.

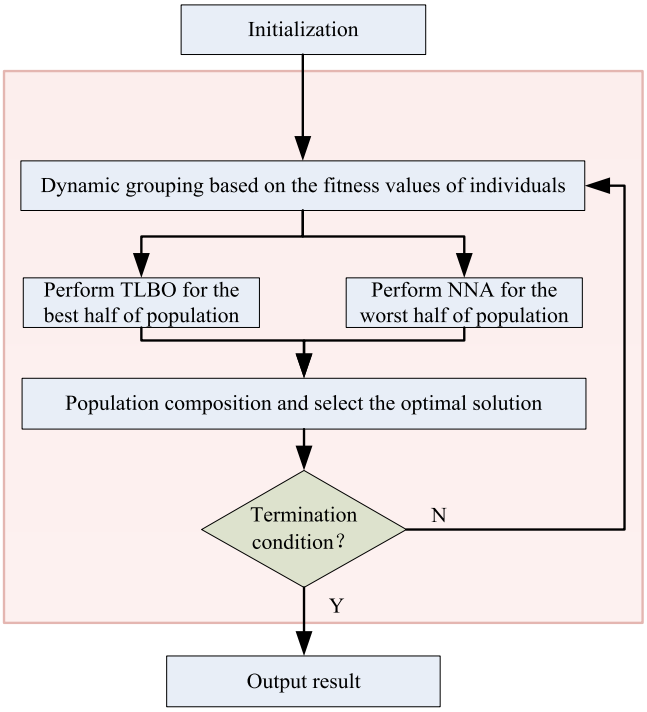

Fig. 4. The framework of TLNNA.

(1.3) Initialize weight matrix. Let E denotes the value of $0.5 * N$. In TLNNA, NNA is employed to optimization E individuals of the population in every loop. Thus the weight matrix $\boldsymbol{W}^t$ is a square matrix (E × E), which can be initialized as [28]

$$\boldsymbol{W}^t = \left[\boldsymbol{w}_1^t, \boldsymbol{w}_2^t, \ldots, \boldsymbol{w}_{\mathrm{E}}^t\right]^{\mathrm{T}} = \begin{bmatrix} w_{1,1}^t & w_{1,2}^t & \ldots & w_{1,\mathrm{E}}^t \\ w_{2,1}^t & w_{2,2}^t & \ldots & w_{2,\mathrm{E}}^t \\ \vdots & \vdots & & \vdots \\ w_{\mathrm{E},1}^t & w_{\mathrm{E},2}^t & \ldots & w_{\mathrm{E},\mathrm{E}}^t \end{bmatrix} \tag{16}$$

$$w_{i,j}^t = s, \quad i = 1, 2, \ldots, \mathrm{E};\ j = 1, 2, \ldots, \mathrm{E} \tag{17}$$

$$\boldsymbol{w}_i^t = \frac{\boldsymbol{w}_i^t}{\sum_{j=1}^{\mathrm{E}} w_{i,j}^t} \tag{18}$$

where s is a random value between 0 and 1. Eq. (17) is the normalization of matrix $\boldsymbol{W}^t$. In this research, the initial weight matrix is randomly generated according to the authors of NNA [28].

Step 2: Population evaluation.

The fitness values of individuals are calculated based on the objective function and the optimal solution $\boldsymbol{G}^t$ is selected. Then the current number of function evaluations N_{current} is updated by

$$N_{\text{current}} = N_{\text{current}} + N \tag{19}$$

Step 3: Termination criteria.

If the current number of function evaluations is greater than the maximum number of function evaluations, the algorithm stops; otherwise, go to Step 4.

Step 4: Dynamic grouping mechanism

Population individuals are sorted based on their fitness values and the population $\boldsymbol{X}^t$ is marked as $\hat{\boldsymbol{X}}^t$ after sorting. The best half of the population $\hat{\boldsymbol{X}}^t$ is marked as $\boldsymbol{P}^t$. The worst half of the population $\hat{\boldsymbol{X}}^t$ is marked as $\boldsymbol{Q}^t$. What is more, $\boldsymbol{P}^t$ and $\boldsymbol{Q}^t$ share the optimal solution to improve the convergence speed, which can be described as

$$\boldsymbol{x}_{\text{opt}}^t = \boldsymbol{G}^t, \boldsymbol{x}_{\text{best}}^t = \boldsymbol{G}^t \tag{20}$$

Step 5: Population optimization.

(5.1) Perform NNA for the population $\boldsymbol{Q}^t$. Firstly, the best solution of population $\boldsymbol{Q}^t$ is replaced by the global optimal solution $\boldsymbol{x}_{\text{opt}}^t$. Secondly, the weight matrix $\boldsymbol{W}^t$ is sorted based on the fitness values and then the target weight $\boldsymbol{w}_{\text{opt}}^t$ is selected. Thirdly, the population $\boldsymbol{Q}^t$ and the weight matrix $\boldsymbol{W}^t$ are updated according to Eqs. (3)–(4) and Eq. (5), respectively. Then if the modification factor β^t is greater than a random number between 0 and 1, the transfer function operator will be performed by Eq. (9); otherwise, the bias operator will be implemented by Eqs. (7)–(8). At last, the new population $\boldsymbol{Q}^{t+1}$ is obtained. According to the optimization process of NNA, the fitness value of every individual is calculated once (line 14 in Fig. 2). Thus the current number of function evaluations can be updated by

$$N_{\text{current}} = N_{\text{current}} + 0.5 * N \tag{21}$$

(5.2) Perform TLBO for the population $\boldsymbol{P}^t$. Firstly, the teacher phase is implemented according to Eqs. (10)–(12). Then the leaner phase is executed based on Eq. (13). At last, the new population $\boldsymbol{P}^{t+1}$ is obtained. Considering the optimization process of TLBO, the fitness value of every individual is calculated twice (lines 8 and 13 in Fig. 3). Thus the current number of function evaluations can be updated by

$$N_{\text{current}} = N_{\text{current}} + N \tag{22}$$

Step 6: Population composition.

(6.1) The population $\boldsymbol{P}^{t+1}$ and the population $\boldsymbol{Q}^{t+1}$ are combined into the population $\boldsymbol{X}^{t+1}$.

(6.2) The modification factor β^{t+1} is calculated by Eq. (6) and the number of iterations t is updated by $t = t + 1$.

Step 7: Go to Step 3.

4. Validation of TLNNA

In order to prove the validation of the proposed method, a variety of experiments are conducted in this section. This section is divided into two parts. Section 4.1 provides 30 complex unconstrained benchmark problems to be tested, with results compared against 7 other competitive meta-heuristic algorithms. Section 4.2 examines 4 challenging structural engineering design problems and the experimental results are compared with other reported solutions.

4.1. TLNNA for unconstrained benchmark functions

4.1.1. Benchmark test functions

In this research, 30 benchmark functions with various types are employed to evaluation the TLNNA, which include 15 well-known basic benchmark functions and CEC2015 test suite.

The 15 basic test functions are often used to examine the optimization performance of different algorithms [7,37,39,40],

Table 1
Basic benchmark functions.

No.	Formulation	Type	Range search	Optimal
F_1	$f(x)=\sum_{i=1}^{D} x_i^2$	Unimodal	$[-100,100]^D$	0
F_2	$f(x)=\sum_{i=1}^{D} i\times x_i^2$	Unimodal	$[-10,10]^D$	0
F_3	$f(x)=\sum_{i=1}^{D} \left(10^6\right)^{\frac{i-1}{D-1}} x_i^2$	Unimodal	$[-100,100]^D$	0
F_4	$f(x)=\sum_{i=1}^{D} i\times x_i^4$	Unimodal	$[-1.28,1.28]^D$	0
F_5	$f(x)=\max_i \{\lvert x_i\rvert, 1\le i\le D\}$	Unimodal	$[-100,100]^D$	0
F_6	$f(x)=\sum_{i=1}^{D} (\lfloor x_i+0.5\rfloor)^2$	Unimodal	$[-100,100]^D$	0
F_7	$f(x)=\sum_{i=1}^{D} i\times x_i^4+random[0,1)$	Unimodal	$[-1.28,0.64]^D$	0
F_8	$f(x)=\sum_{i=1:}^{D} \lvert x_i\rvert+\prod_{i=1}^{D}\lvert x_i\rvert$	Unimodal	$[-10,10]^D$	0
F_9	$f(x)=\sum_{i=1}^{D}\left(\sum_{j=1}^{i} x_j\right)^2$	Multi-modal	$[-100,100]^D$	0
F_{10}	$f(x)=\frac{1}{4000}\sum_{i=1}^{D} x_i^2-\prod_{i=1}^{D}\cos\frac{x_i}{\sqrt{i}}+1$	Multi-modal	$[-600,600]^D$	0
F_{11}	$f(x)=\sum_{i=1}^{D}\lvert x_i\sin(x_i)+0.1x_i\rvert$	Multi-modal	$[-10,10]^D$	0
F_{12}	$f(x)=-20\exp\left(-0.2\sqrt{\frac{1}{D}\sum_{i=1}^{D} x_i^2}\right)-\exp\left(\frac{1}{D}\sum_{i=1}^{D}\cos 2\pi x_i\right)+20+e$	Multi-modal	$[-32,32]^D$	0
F_{13}	$f(x)=\sum_{i=1}^{D} x_i^2+\left(\sum_{i=1}^{D} 0.5\times i\times x_i\right)^2+\left(\sum_{i=1}^{D} 0.5\times i\times x_i\right)^4$	Multi-modal	$[-10,10]^D$	0
F_{14}	$f(x)=0.5+\frac{\sin^2\left(\sqrt{\sum_{i=1}^{D} x_i^2}\right)-0.5}{\left(1+0.001\left(\sum_{i=1}^{D} x_i^2\right)\right)^2}$	Multi-modal	$[-100,100]^D$	0
F_{15}	$f(x)=\sum_{i=1}^{D} y_i^2-10\cos(2\pi y_i)+10,\ y_i=\begin{cases} x_i, & \lvert x_i\rvert<0.5 \\ 0.5*round(2x_i), & \lvert x_i\rvert\ge 0.5\end{cases}$	Multi-modal	$[-5.12,5.12]^D$	0

which have been listed in Table 1. These test functions include 8 unimodal functions (F_1–F_8) and 7 multi-modal functions (F_9–F_{15}).

CEC2015's benchmark functions [41] have been presented in Table 2, which consist of four categories: F_{16}–F_{17} are rotated unimodal functions, F_{18}–F_{20} are rotated and shifted multi-modal functions, F_{21}–F_{23} are hybrid functions and F_{24}–F_{30} are composition functions. In these functions, multi-modal functions are complex which are both shifted and rotated; hybrid functions consist of more than one function, which are more difficult to solve compared with multi-modal functions; composition functions have maximally complex, which combine unimodal, multi-modal and hybrid functions. In general, the benchmark functions of CEC2015 test suite are more complex compared with the basic functions.

4.1.2. Parameter settings

4.1.2.1. The applied algorithms. In order to better show the efficiency and robustness of the proposed algorithm in solving the unconstrained benchmark problems, seven competitive meta-heuristic algorithms are selected to compare with TLNNA, which include the classical algorithm (PSO), the recently proposed algorithms (SCA, WOA, SSA and GWO), and the basic algorithms (TLBO and NNA) of TLNNA. How to make a fair comparison for different meta-heuristic algorithms is a very difficult task due to different features of the applied algorithms. For having fair comparison, like many other related studies [7,11,37,42,43], the parameters of the compared algorithms were taken from the original literature and have been shown in Table 3. In addition, in GWO, SSA, SCA and WOA, there are some parameters relating to the maximum number of function evaluations, which are not shown in Table 3 and can be found in the corresponding references.

4.1.2.2. Dimensions. For each test function, two dimensions were investigated, which include 10-dimensional ($10D$) and 30-dimensional ($30D$).

4.1.2.3. The independent trials. Considering the results of a single run might be unreliable due to the stochastic nature of meta-heuristics, all applied algorithms were executed in 30 independent runs for the same test problem. Then the average values of the obtained results were recorded.

4.1.2.4. Stopping criteria. The maximum number of function evaluations is often considered as the stopping criteria for meta-heuristic algorithms. How to determine the maximum number of

Table 2
CEC2015's benchmark functions.

No.	Name	Type	Range search	Optimal
F_{16}	Rotated high conditioned elliptic function	Unimodal	$[-100,100]^D$	100
F_{17}	Rotated cigar function	Unimodal	$[-100,100]^D$	200
F_{18}	Shifted and rotated Ackley's function	Multi-modal	$[-100,100]^D$	300
F_{19}	Shifted and rotated Rastrigin's function	Multi-modal	$[-100,100]^D$	400
F_{20}	Shifted and rotated Schwefel's function	Multi-modal	$[-100,100]^D$	500
F_{21}	Hybrid function 1 (n = 3)	Hybrid	$[-100,100]^D$	600
F_{22}	Hybrid function 2 (n = 4)	Hybrid	$[-100,100]^D$	700
F_{23}	Hybrid function 3 (n = 5)	Hybrid	$[-100,100]^D$	800
F_{24}	Composition 1 (n = 3)	Composition	$[-100,100]^D$	900
F_{25}	Composition 2 (n = 3)	Composition	$[-100,100]^D$	1000
F_{26}	Composition 3 (n = 5)	Composition	$[-100,100]^D$	1100
F_{27}	Composition 4 (n = 5)	Composition	$[-100,100]^D$	1200
F_{28}	Composition 5 (n = 5)	Composition	$[-100,100]^D$	1300
F_{29}	Composition 6 (n = 7)	Composition	$[-100,100]^D$	1400
F_{30}	Composition 7 (n = 10)	Composition	$[-100,100]^D$	1500

Table 3
Parameter settings of the applied algorithms.

Algorithm	Parameter	Reference
GWO	Population size = 30.	[7]
NNA	Population size = 50.	[28]
WOA	Population size = 30.	[8]
SCA	Population size = 30.	[11]
SSA	Population size = 30.	[9]
TLBO	Population size = 50.	[1]
TLNNA	Population size = 50.	–
PSO	Population size = 40, personal learning coefficient = 1.4986, social learning coefficient = 1.4986, weight = 0.72984.	[44]

function evaluations all depends on the specific algorithms and the problems to be solved, which is not unity. In our experiments, it is 5000 multiples by dimension size for each test function [28, 45], which is sufficient for most test problems with both 10*D* and 30*D*.

4.1.3. Evaluation metrics

In order to analyze the experimental results, three quality indicators are used in the following experiments.

The first one is the value-based method, which is the solution quality in terms of two metrics including mean value and standard deviation. The smaller the mean value is, the stronger the global optimization ability of the algorithm is; the smaller the standard deviation is, the more stability the algorithm is.

The second one is the rank-based method. Tied rank (TR) [37] is used in this section to intuitively compare the optimization performance among the considered algorithms. In this research, TR assigns rank 1 to the algorithm with the best mean value; rank 2 to the second best and rank M (the number of the considered algorithms) to the Mth best. In addition, several algorithms with the same results will share the average of ranks. The smaller the value of TR is, the better the algorithm is.

The third one is the statistical test-based method. Wilcoxon signed-rank test [46] is employed to compare the performance between the proposed TLNNA and the other algorithms. Compared with t-test, Wilcoxon signed-rank test has the following advantages [46]: (1) it does not assume normal distributions for the tested sample; (2) it is less affected than the t-test by the outliers; (3) it is more sensitive than the t-test. Considering these advantages, it has been widely used to compare the optimization performance between two algorithms [47–50].

4.1.4. Results and analysis

4.1.4.1. The comparison of mean value and standard deviation. The statistical results of mean value and standard deviation are presented in Tables 4–5 for 10*D* and 30*D*. In these tables, "Mean", "Std" and "Rank" stand for mean optimal value, standard deviation and ranking, respectively. Moreover, the best results are highlighted in bold.

As given in Table 4 for the dimension 10, the proposed TLNNA outperforms NNA and TLBO on 28 and 25 functions, respectively. However, NNA and TLBO only can surpass TLNNA on 1(F_{18}) and 4 (F_{19}, F_{25}, F_{29} and F_{30}) functions, respectively. In addition, TLNNA is superior to SCA on all functions except F_6. SSA and PSO are inferior to TLNNA on 28 functions. TLNNA can get better results than WOA on 23 functions while WOA only offer better solutions than TLNNA for 5 test functions (F_1, F_2, F_3, F_4 and F_8). Besides, GWO shows a strong competitiveness, which can acquire better results than TLNNA on 12 functions (F_1, F_2, F_3, F_4, F_5, F_7, F_8, F_9, F_{11}, F_{13}, F_{20} and F_{26}). But TLNNA still achieves better results than GWO on more than half of functions. Note that GWO only shows better performance than TLNNA on 2 functions (F_{20} and F_{26}) of CEC2015 test suit, which indicates TLNNA is more suitable to solve relatively complex optimization problems compared with GWO.

When examining the values shown in Table 5 for the dimension 30, the proposed TLNNA provides better solutions than NNA and TLBO on 22 and 23 functions, respectively. NNA and TLBO can get better results than TLNNA on 7 (F_{15}, F_{18}, F_{20}, F_{24}, F_{28}, F_{29} and F_{30}) and 4 functions (F_{19}, F_{24}, F_{27} and F_{29}), respectively. PSO is only superior to TLNNA for 2 functions (F_{26} and F_{29}) among all 30 functions. In addition, SCA cannot compete with TLNNA on all functions except F_6. TLNNA can get better results than SSA on all functions except 5 functions (F_{18}, F_{20}, F_{24}, F_{29} and F_{30}). Besides, both GWO and WOA only can surpass TLNNA on 4 functions.

Based on the above analysis, it is clearly that the proposed method can offer better solutions than the compared algorithms in solving most test functions. Note that TLNNA is superior to both NNA and TLBO on at least 22 functions with 10*D* and 30*D*. This fully demonstrates that the combination of NNA and TLBO is very effective.

In order to compare the convergence performance of the applied algorithms, Figs. 5 and 6 presents some typical averaged convergence curves over 30 independent runs corresponding to 10*D* and 30*D*, respectively. The selected functions with 10*D* are F_6, F_{10}, F_{16}, F_{21}, F_{23} and F_{25}. The selected functions with 30*D* include F_5, F_6, F_9, F_{10}, F_{14}, F_{16}, F_{21}, F_{23} and F_{25}. In general, the complexity of the problem increases exponentially with its dimension [51]. Looking closely at Figs. 5 and 6, in terms of convergence speed, the proposed method has more obvious advantage on 30*D* than 10*D* compared with the other algorithms, which can be clearly seen on F_{21}, F_{23} and F_{25}. Thus the proposed TLNNA is more suitable for solving complicated optimization problems compared with the other algorithms.

Table 4
The statistical results of the applied algorithms on 30 benchmark functions with 10*D* (the best results are highlighted in bold).

No.		GWO	NNA	SCA	TLBO	WOA	SSA	PSO	TLNNA
	Mean	4.38E−198	3.22E−22	1.77E−45	4.32E−108	**9.48E−265**	5.39E−10	7.81E−62	1.82E−135
F_1	Std	**0.00E+00**	1.37E−21	6.38E−45	9.73E−108	**0.00E+00**	1.97E−10	1.36E−61	5.80E−135
	Rank	2	7	6	4	1	8	5	3
	Mean	3.97E−198	3.94E−23	7.74E−47	4.02E−109	**8.62E−264**	3.33E−11	1.14E−62	1.11E−136
F_2	Std	**0.00E+00**	1.12E−22	2.81E−46	8.46E−109	**0.00E+00**	1.13E−11	3.11E−62	4.22E−136
	Rank	2	7	6	4	1	8	5	3
	Mean	1.41E−194	1.16E−15	1.50E−41	8.85E−104	**3.47E−255**	2.54E+05	9.96E−57	1.46E−131
F_3	Std	**0.00E+00**	3.21E−15	6.13E−41	1.86E−103	**0.00E+00**	1.82E+05	4.54E−56	3.53E−131
	Rank	2	7	6	4	1	8	5	3
	Mean	**0.00E+00**	1.22E−46	3.48E−73	4.19E−210	**0.00E+00**	1.02E−26	8.19E−119	1.53E−259
F_4	Std	**0.00E+00**	6.30E−46	1.91E−72	**0.00E+00**	**0.00E+00**	7.02E−27	3.02E−118	**0.00E+00**
	Rank	1.5	7	6	4	1.5	8	5	3
	Mean	**2.68E−61**	6.85E−06	2.90E−14	2.04E−45	6.17E−01	1.36E−05	1.28E−22	6.30E−58
F_5	Std	**1.26E−60**	6.14E−06	9.38E−14	1.91E−45	2.15E+00	3.15E−06	2.86E−22	1.75E−57
	Rank	1	6	5	3	8	7	4	2
	Mean	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	4.67E−01	**0.00E+00**	**0.00E+00**
F_6	Std	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	7.30E−01	**0.00E+00**	**0.00E+00**
	Rank	4	4	4	4	4	8	4	4
	Mean	**1.66E−04**	1.04E−03	7.83E−04	5.67E−04	3.40E−04	4.07E−03	1.34E−03	3.05E−04
F_7	Std	**1.08E−04**	8.75E−04	6.25E−04	2.38E−04	3.44E−04	1.92E−03	8.73E−04	2.18E−04
	Rank	1	6	5	4	3	8	7	2
	Mean	1.95E−112	3.24E−12	7.93E−32	3.56E−54	**1.01E−178**	4.97E−04	8.98E−34	3.31E−68
F_8	Std	3.84E−112	5.31E−12	3.27E−31	6.45E−54	**0.00E+00**	2.55E−03	1.18E−33	6.00E−68
	Rank	2	7	6	4	1	8	5	3
	Mean	**3.04E−89**	4.62E−09	6.47E−17	6.81E−46	1.29E+00	1.10E−09	6.56E−29	6.30E−65
F_9	Std	**1.09E−88**	1.16E−08	3.43E−16	3.40E−45	3.18E+00	6.06E−10	1.85E−28	1.83E−64
	Rank	1	7	5	3	8	6	4	2
	Mean	7.15E−03	8.10E−02	8.15E−02	3.00E−03	3.31E−02	2.68E−01	6.20E−02	**2.54E−03**
F_{10}	Std	1.44E−02	3.31E−02	2.37E−01	8.78E−03	6.58E−02	1.26E−01	2.61E−02	**1.39E−02**
	Rank	3	6	7	2	4	8	5	1
	Mean	**1.63E−197**	5.31E−08	1.27E−40	6.20E−19	3.60E−02	1.47E−01	4.08E−16	1.16E−114
F_{11}	Std	**0.00E+00**	9.38E−08	6.91E−40	3.39E−18	1.97E−01	8.69E−02	2.76E−16	6.36E−114
	Rank	1	6	3	4	7	8	5	2
	Mean	4.68E−15	2.25E−11	3.73E−15	4.44E−15	3.49E−15	6.95E−01	3.85E−02	**2.07E−15**
F_{12}	Std	9.01E−16	4.22E−11	1.45E−15	**0.00E+00**	2.07E−15	8.43E−01	2.11E−01	1.70E−15
	Rank	5	6	3	4	2	8	7	1
	Mean	**4.39E−115**	6.34E−15	7.05E−26	3.68E−50	4.63E−03	1.25E−11	2.68E−37	2.43E−69
F_{13}	Std	**2.41E−114**	1.32E−14	1.86E−25	9.25E−50	5.92E−03	3.75E−12	8.96E−37	9.10E−69
	Rank	1	6	5	3	8	7	4	2
	Mean	3.15E−03	3.13E−03	2.81E−03	3.16E−03	2.50E−03	4.36E−03	3.14E−03	**2.29E−03**
F_{14}	Std	7.26E−11	**2.26E−17**	9.54E−04	6.16E−09	1.27E−03	1.74E−03	2.02E−12	1.41E−03
	Rank	6	4	3	7	2	8	5	1
	Mean	1.33E−01	1.18E−16	1.19E+00	5.17E+00	**0.00E+00**	2.04E+01	7.30E+00	**0.00E+00**
F_{15}	Std	7.30E−01	4.51E−16	4.69E+00	9.47E−01	**0.00E+00**	5.69E+00	3.00E+00	**0.00E+00**
	Rank	4	3	5	6	1.5	8	7	1.5
	Mean	2.87E+06	8.15E+05	8.04E+06	8.54E+04	2.76E+06	1.29E+06	2.94E+06	**2.40E+04**
F_{16}	Std	2.43E+06	3.68E+05	4.04E+06	5.31E+04	2.36E+06	6.80E+05	9.86E+05	**3.05E+04**
	Rank	6	3	8	2	5	4	7	1
	Mean	7.31E+06	8.81E+03	6.26E+08	1.21E+04	1.26E+06	8.94E+03	8.78E+07	**6.60E+03**
F_{17}	Std	6.25E+06	1.08E+04	2.22E+08	8.83E+03	1.51E+06	9.85E+03	2.08E+07	**5.80E+03**
	Rank	6	2	8	4	5	3	7	1
	Mean	320.3671	**320.0064**	320.3872	320.3535	320.1453	320.0256	320.4049	320.0369
F_{18}	Std	9.23E−02	**2.14E−02**	7.93E−02	9.02E−02	1.03E−01	5.23E−02	1.00E−01	6.17E−02
	Rank	6	1	7	5	4	2	8	3
	Mean	4.17E+02	4.20E+02	4.44E+02	**4.10E+02**	4.49E+02	4.20E+02	4.29E+02	4.12E+02
F_{19}	Std	7.17E+00	8.40E+00	4.78E+00	**3.40E+00**	9.24E+00	9.01E+00	5.60E+00	6.36E+00
	Rank	3	4	7	1	8	5	6	2
	Mean	**1.01E+03**	1.08E+03	1.76E+03	1.33E+03	1.84E+03	1.19E+03	1.55E+03	1.04E+03
F_{20}	Std	2.80E+02	2.20E+02	1.97E+02	3.93E+02	2.77E+02	2.99E+02	2.24E+02	**1.76E+02**
	Rank	1	3	7	5	8	4	6	2
	Mean	1.70E+04	3.70E+03	2.21E+04	2.34E+03	4.77E+05	3.33E+03	5.05E+03	**1.66E+03**
F_{21}	Std	2.20E+04	3.17E+03	2.43E+04	**1.06E+03**	2.80E+05	2.05E+03	1.62E+03	1.43E+03
	Rank	6	4	7	2	8	3	5	1
	Mean	702.1283	701.8170	705.9348	701.2947	706.6355	703.0319	703.6828	**701.2045**
F_{22}	Std	7.59E−01	7.07E−01	7.92E−01	**4.94E−01**	1.32E+00	9.27E−01	6.72E−01	5.90E−01
	Rank	4	3	7	2	8	5	6	1
	Mean	3.14E+03	5.44E+03	5.42E+03	1.37E+03	4.81E+03	5.72E+03	2.99E+03	**1.27E+03**
F_{23}	Std	1.84E+03	5.35E+03	3.03E+03	**2.28E+02**	5.20E+03	4.03E+03	1.28E+03	2.86E+02
	Rank	4	7	6	2	5	8	3	1
	Mean	1000.438	1000.446	1002.785	1000.365	1000.642	**1000.348**	1000.416	1000.360
F_{24}	Std	1.15E−01	1.44E−01	1.18E+00	1.21E−01	2.91E−01	1.45E−01	**7.77E−02**	9.17E−02
	Rank	5	6	8	3	7	1	4	2

(*continued on next page*)

Table 4 (*continued*).

No.		GWO	NNA	SCA	TLBO	WOA	SSA	PSO	TLNNA
F_{25}	Mean	4.96E+03	3.51E+03	6.35E+03	1.70E+03	1.44E+04	4.57E+03	7.79E+03	**1.48E+03**
	Std	4.14E+03	2.73E+03	4.70E+03	3.27E+02	9.77E+03	2.18E+03	3.49E+03	**2.13E+02**
	Rank	5	3	6	2	8	4	7	1
F_{26}	Mean	**1234.402**	1371.552	1390.252	1312.558	1394.416	1381.265	1401.397	1322.265
	Std	1.49E+02	8.92E+01	5.77E+01	1.36E+02	5.13E+01	7.44E+01	**3.26E−01**	1.31E+02
	Rank	1	4	6	2	7	5	8	3
F_{27}	Mean	1302.369	1304.425	1307.985	1302.479	1305.057	1303.698	1304.287	**1302.031**
	Std	6.42E−01	1.49E+00	1.24E+00	6.49E−01	1.61E+00	1.20E+00	**5.24E−01**	7.90E−01
	Rank	2	6	8	3	7	4	5	1
F_{28}	Mean	1300.036	1300.032	1300.044	1300.031	1300.034	1300.033	1300.043	**1300.030**
	Std	2.77E−02	1.52E−04	3.08E−03	**1.39E−04**	2.17E−03	2.51E−04	8.38E−03	3.78E−04
	Rank	6	3	8	2	5	4	7	1
F_{29}	Mean	8.31E+03	5.54E+03	7.99E+03	4.36E+03	7.94E+03	7.01E+03	**3.26E+03**	5.05E+03
	Std	2.62E+03	2.56E+03	1.23E+03	2.13E+01	**1.20E+03**	2.67E+03	1.72E+03	1.85E+03
	Rank	8	4	7	2	6	5	1	3
F_{30}	Mean	1608.556	1600.010	1620.381	**1600.000**	1602.730	1600.020	1609.802	1600.007
	Std	4.63E+00	1.13E−07	4.02E+00	**3.78E−13**	2.00E+00	4.06E−06	1.42E+00	1.04E−12
	Rank	6	3	8	1	5	4	7	2
Average ranking		3.52	4.83	6.10	3.27	4.97	5.90	5.47	1.95
Total ranking		3	4	8	2	5	7	6	1

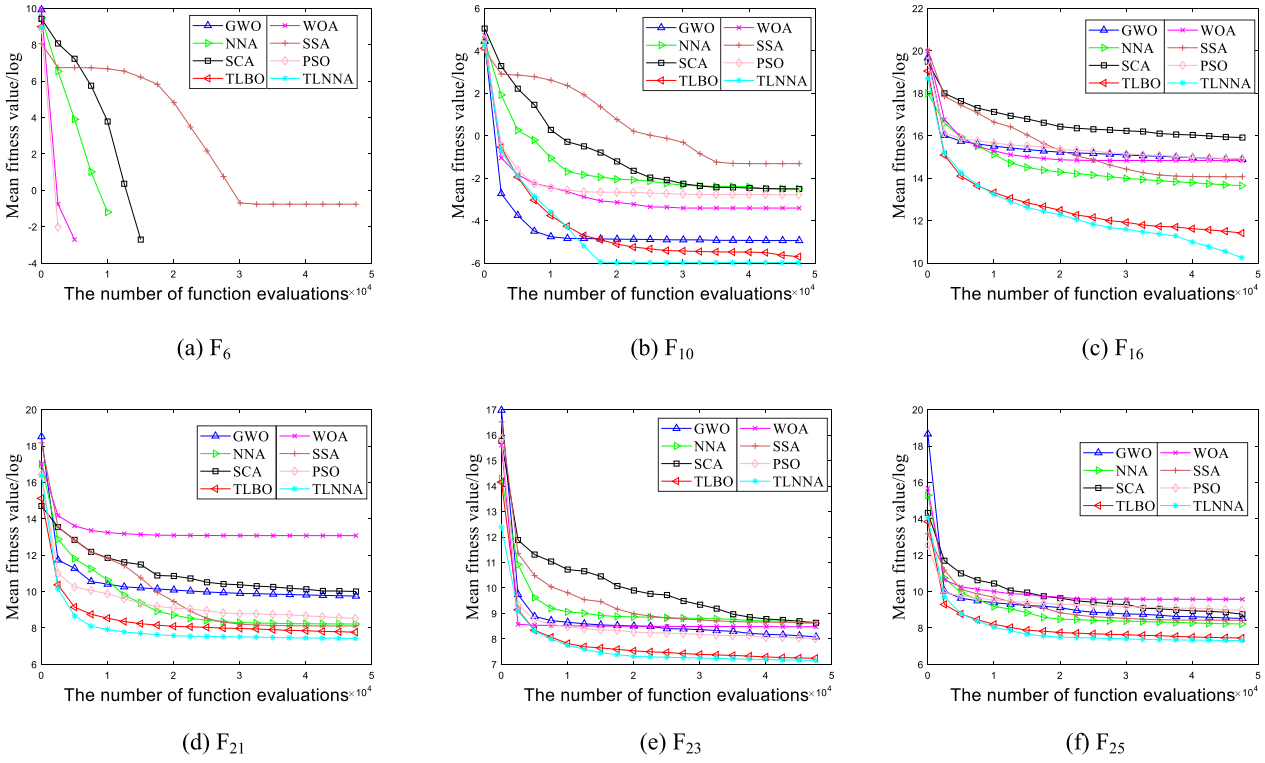

Fig. 5. The convergence curves of average fitness value on selected functions with 10*D*.

4.1.4.2. The statistical results of tied rank. The experimental results of the tied rank on 10*D* and 30*D* functions are listed in Tables 4 and 5, respectively. Moreover, the last two columns of these tables indicate the overall ranking and the average ranking of the proposed TLNNA over the other algorithms.

From Tables 4 and 5, the applied algorithms can be sorted by the overall ranking in the following order:

(1) For 10*D*: TLNNA, TLBO, GWO, NNA, WOA, PSO, SSA and SCA.
(2) For 30*D*: TLNNA, TLBO, GWO, NNA, WOA, SSA, PSO and SCA.

Obviously, the proposed TLNNA is the best algorithm on both 10*D* and 30*D*. TLBO and GWO show strong competitiveness, which are the second and third best on all test dimensions. Moreover, Fig. 7 shows the radar chart for overall ranking on different types of functions. From Fig. 7, TLBO is the second place on CEC 2015 and all functions for both 10*D* and 30*D* while TLBO is inferior to WOA, GWO and TLNNA on basic functions. NNA is the third and fourth places for CEC2015 and all functions on both 10*D* and 30*D*, respectively. However, in terms of basic functions, NNA is only superior to SSA on 10*D* and only surpasses SCA and SSA on 30*D*. In addition, it is clearly that the proposed TLNNA is the best for all types of functions with both 10*D* and 30*D*. This fully indicates that TLNNA based on TLBO and NNA is an effective method to solve different types of optimization problems.

4.1.4.3. The results analysis of Wilcoxon signed-rank test. The results produced by Wilcoxon signed-rank test with a significance level $\alpha = 0.05$ are shown in Tables 6–7 for 10*D* and 30D, respectively. In these tables, "H" is marked as "1", which means there is significant difference between TLNNA and the compared algorithm; "H" is marked as "0", which indicates there is no

Table 5
The statistical results of the applied algorithms on 30 benchmark functions with 30*D* (the best results are highlighted in bold).

No.		GWO	NNA	SCA	TLBO	WOA	SSA	PSO	TLNNA
	Mean	**0.00E+00**	1.62E−21	6.95E−25	1.07E−268	**0.00E+00**	5.13E−09	3.99E−73	**0.00E+00**
F_1	Std	**0.00E+00**	4.72E−21	3.47E−24	**0.00E+00**	**0.00E+00**	8.17E−10	2.00E−72	**0.00E+00**
	Rank	2	7	6	4	2	8	5	2
	Mean	**0.00E+00**	5.47E−22	4.91E−26	1.38E−268	**0.00E+00**	1.04E−08	5.37E−75	**0.00E+00**
F_2	Std	**0.00E+00**	1.67E−21	1.85E−25	**0.00E+00**	**0.00E+00**	2.16E−09	1.50E−74	**0.00E+00**
	Rank	2	7	6	4	2	8	5	2
	Mean	4.38E−308	1.30E−15	1.65E−24	9.49E−264	**0.00E+00**	1.44E+06	2.42E−69	**0.00E+00**
F_3	Std	**0.00E+00**	2.28E−15	5.67E−24	**0.00E+00**	**0.00E+00**	5.14E+05	8.79E−69	**0.00E+00**
	Rank	3	7	6	4	1.5	8	5	1.5
	Mean	**0.00E+00**	3.61E−46	3.88E−27	**0.00E+00**	**0.00E+00**	1.25E−24	3.70E−117	**0.00E+00**
F_4	Std	**0.00E+00**	1.45E−45	1.66E−26	**0.00E+00**	**0.00E+00**	4.70E−25	1.47E−116	**0.00E+00**
	Rank	2.5	6	7	2.5	2.5	8	5	2.5
	Mean	5.68E−76	3.55E−03	1.93E+00	8.63E−108	1.47E+01	1.68E+00	3.19E−05	**3.52E−147**
F_5	Std	1.29E−75	1.96E−03	3.11E+00	1.56E−107	2.11E+01	2.07E+00	3.60E−05	**1.20E−146**
	Rank	3	5	7	2	8	6	4	1
	Mean	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	7.53E+00	5.33E−01	**0.00E+00**
F_6	Std	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	3.81E+00	1.20E+00	**0.00E+00**
	Rank	3.5	3.5	3.5	3.5	3.5	8	7	3.5
	Mean	1.41E−04	2.50E−03	3.48E−03	3.47E−04	3.58E−04	1.91E−02	4.29E−03	**1.07E−04**
F_7	Std	8.68E−05	1.49E−03	2.56E−03	1.22E−04	4.55E−04	7.51E−03	2.00E−03	**5.09E−05**
	Rank	2	5	6	3	4	8	7	1
	Mean	5.69E−179	1.39E−11	4.48E−28	1.12E−133	**0.00E+00**	5.48E−01	2.06E−30	4.73E−177
F_8	Std	**0.00E+00**	3.58E−11	1.63E−27	1.24E−133	**0.00E+00**	7.52E−01	6.43E−30	**0.00E+00**
	Rank	2	7	6	4	1	8	5	3
	Mean	2.40E−88	1.90E−04	3.53E+02	1.04E−56	1.41E+01	3.45E−07	8.04E−08	**2.28E−149**
F_9	Std	1.26E−87	4.52E−04	1.54E+03	5.20E−56	3.41E+01	1.06E−07	1.44E−07	**1.19E−148**
	Rank	2	6	8	3	7	5	4	1
	Mean	9.29E−04	5.75E−03	2.79E−03	**0.00E+00**	8.27E−04	7.55E−03	1.47E−02	**0.00E+00**
F_{10}	Std	2.86E−03	8.49E−03	1.07E−02	**0.00E+00**	3.19E−03	7.53E−03	1.38E−02	**0.00E+00**
	Rank	4	6	5	1.5	3	7	8	1.5
	Mean	**0.00E+00**	1.58E−06	1.32E−16	3.85E−269	**0.00E+00**	3.20E+00	1.40E−15	**0.00E+00**
F_{11}	Std	**0.00E+00**	2.05E−06	7.13E−16	**0.00E+00**	**0.00E+00**	2.36E+00	5.04E−16	**0.00E+00**
	Rank	2	7	5	4	2	8	6	2
	Mean	8.11E−15	2.05E−11	1.32E+01	5.51E−15	**4.09E−15**	1.88E+00	7.71E−01	4.32E−15
F_{12}	Std	1.47E−15	1.75E−11	9.35E+00	1.66E−15	2.53E−15	8.65E−01	8.30E−01	**6.49E−16**
	Rank	4	5	8	3	1	7	6	2
	Mean	1.12E−107	1.06E−09	2.73E−02	1.96E−36	5.49E+02	2.55E−10	1.89E−13	**7.09E−124**
F_{13}	Std	6.00E−107	8.95E−10	7.37E−02	3.09E−36	1.25E+01	5.87E−11	3.24E−13	**3.67E−123**
	Rank	2	6	7	3	8	5	4	1
	Mean	3.14E−03	4.06E−03	3.23E−03	3.15E−03	**2.71E−03**	1.20E−02	7.76E−03	3.13E−03
F_{14}	Std	2.33E−11	1.44E−03	5.66E−04	1.72E−09	1.08E−03	2.30E−03	2.09E−03	**0.00E+00**
	Rank	3	6	5	4	1	8	7	2
	Mean	**0.00E+00**	1.43E−14	9.33E+00	1.76E+01	**0.00E+00**	8.70E+01	3.19E+01	7.37E+00
F_{15}	Std	**0.00E+00**	6.22E−14	1.36E+01	5.50E+00	**0.00E+00**	2.78E+01	1.75E+01	7.08E+00
	Rank	1.5	3	5	6	1.5	8	7	4
	Mean	2.70E+07	1.99E+06	1.76E+08	6.42E+05	1.89E+07	2.01E+06	3.14E+07	**4.80E+05**
F_{16}	Std	1.20E+07	1.01E+06	6.00E+07	8.00E+05	1.21E+07	1.14E+06	7.12E+06	**3.49E+05**
	Rank	6	3	8	2	5	4	7	1
	Mean	2.87E+09	5.18E+03	1.90E+10	2.96E+03	4.37E+07	3.97E+03	2.02E+09	**2.73E+03**
F_{17}	Std	2.00E+09	5.76E+03	3.73E+09	4.14E+03	4.22E+07	4.33E+03	2.60E+08	**2.84E+03**
	Rank	7	4	8	2	5	3	6	1
	Mean	320.9826	**320.0536**	320.9673	320.9409	320.4568	320.0850	320.9674	320.1178
F_{18}	Std	5.57E−02	9.22E−02	4.65E−02	7.69E−02	1.79E−01	1.21E−01	**4.63E−02**	1.42E−01
	Rank	8	1	6	5	4	2	7	3
	Mean	4.96E+02	5.31E+02	6.89E+02	**4.85E+02**	6.89E+02	5.34E+02	6.11E+02	4.94E+02
F_{19}	Std	2.59E+01	2.86E+01	1.86E+01	**1.26E+01**	5.68E+01	4.20E+01	2.06E+01	1.86E+01
	Rank	3	4	8	1	7	5	6	2
	Mean	**3.31E+03**	3.76E+03	7.61E+03	7.19E+03	4.98E+03	4.13E+03	7.05E+03	4.29E+03
F_{20}	Std	5.30E+02	5.70E+02	3.71E+02	**3.63E+02**	8.76E+02	7.35E+02	3.86E+02	5.80E+02
	Rank	1	2	8	7	5	3	6	4
	Mean	1.63E+06	3.08E+05	6.93E+06	1.74E+05	5.19E+06	1.42E+05	1.12E+06	**6.60E+04**
F_{21}	Std	1.24E+06	1.75E+05	2.98E+06	1.60E+05	1.76E+06	1.27E+05	5.18E+05	**3.83E+04**
	Rank	6	4	8	3	7	2	5	1
	Mean	7.20E+02	7.15E+02	7.49E+02	**7.11E+02**	7.27E+02	7.18E+02	7.28E+02	7.13E+02
F_{22}	Std	6.34E+00	**2.30E+00**	2.12E+01	3.32E+00	1.22E+01	5.38E+00	5.76E+00	2.54E+00
	Rank	5	3	8	1	6	4	7	2
	Mean	5.37E+05	1.21E+05	1.40E+06	6.91E+04	6.48E+05	7.56E+04	2.83E+05	**2.54E+04**
F_{23}	Std	8.80E+05	8.32E+04	6.07E+05	3.88E+04	4.28E+05	4.14E+04	8.52E+04	**1.26E+04**
	Rank	6	4	8	2	7	3	5	1
	Mean	1043.728	1004.283	1073.956	**1003.826**	1107.217	1004.267	1010.608	1004.547
F_{24}	Std	7.01E+01	**3.90E−01**	1.17E+01	6.89E−01	1.87E+02	5.64E−01	1.36E+00	6.41E−01
	Rank	6	3	7	1	8	2	5	4

(*continued on next page*)

Table 5 (*continued*).

No.		GWO	NNA	SCA	TLBO	WOA	SSA	PSO	TLNNA
F_{25}	Mean	1.87E+06	1.19E+05	4.98E+06	1.11E+05	4.90E+06	1.59E+05	8.61E+05	**3.41E+04**
	Std	1.23E+06	1.07E+05	2.63E+06	6.14E+04	2.35E+06	1.28E+05	4.21E+05	**2.98E+04**
	Rank	6	3	8	2	7	4	5	1
F_{26}	Mean	1.88E+03	1.99E+03	2.35E+03	1.97E+03	2.48E+03	2.00E+03	**1.77E+03**	1.86E+03
	Std	1.27E+02	2.74E+02	3.23E+02	1.54E+02	2.03E+02	**1.11E+02**	3.33E+02	3.30E+02
	Rank	3	5	7	4	8	6	1	2
F_{27}	Mean	1310.460	1328.060	1379.510	**1308.363**	1315.197	1388.383	1400.538	1360.227
	Std	1.72E+01	3.68E+01	3.16E+01	1.18E+00	1.62E+01	3.12E+01	**3.43E−02**	4.67E+01
	Rank	2	4	6	1	3	7	8	5
F_{28}	Mean	1300.068	**1300.029**	1300.497	1300.031	1300.038	1300.037	1300.072	1300.031
	Std	2.87E−02	**1.94E−03**	1.66E−01	6.08E−03	1.27E−02	1.14E−02	1.21E−02	3.48E−03
	Rank	6	1	8	3	5	4	7	2
F_{29}	Mean	3.73E+04	3.52E+04	4.87E+04	3.45E+04	3.85E+04	3.54E+04	**3.26E+04**	3.61E+04
	Std	1.70E+03	1.43E+03	2.69E+03	1.46E+03	2.20E+03	**1.09E+03**	9.26E+03	1.51E+03
	Rank	6	3	8	2	7	4	1	5
F_{30}	Mean	1670.813	**1600.000**	2227.516	1604.366	1639.728	1600.010	1615.687	1600.771
	Std	1.38E+02	**7.56E−07**	4.71E+02	5.92E+00	2.07E+01	3.17E−06	9.17E−01	2.39E+00
	Rank	7	1	8	4	6	2	5	3
Average ranking		3.88	4.38	6.82	3.05	4.60	5.50	5.53	2.23
Total ranking		3	4	8	2	5	6	7	1

significant difference between TLNNA and the compared algorithm. "S" is marked as "+", which imply the proposed method is superior to the compared algorithm; "S" is marked as "=", which indicates TLNNA has the same performance with the compared algorithm; "S" is marked as "−", which demonstrates TLNNA is inferior to the compared algorithm. Moreover, the last column of the each of these tables under the heading $w/t/l$ represents the win, tie and lose counts of the proposed TLNNA over the compared algorithms. Besides, Fig. 8 shows the statistical results of Wilcoxon signed-rank test.

From Fig. 8(a), the proposed TLNNA is superior to NNA and TLBO on 23 and 19 functions, respectively. Note that NNA and TLBO only surpass TLNNA on 1(F_{18}) and 2(F_{29} and F_{30}) functions, respectively. In addition, GWO, SCA, WOA, SSA and PSO are inferior to TLNNA on 13, 28, 21, 26 and 28 functions, respectively. SCA and PSO only get better results than TLNNA on 1 function (F_{29}). GWO and WOA show strong competitiveness, which can offer better solutions than TLNNA on 12 (F_1, F_2, F_3, F_4, F_5, F_7, F_8, F_9, F_{11}, F_{13}, F_{14} and F_{26}) and 7 (F_1, F_2, F_3, F_4, F_8, F_{11} and F_{13}) functions, respectively. However, in terms of complex CEC2015 test suit, GWO only can obtain better result than TLNNA on F_{26}; WOA cannot excel TLNNA on any function.

From Fig. 8(b), NNA and TLBO are inferior to TLNNA on 19 and 17 functions, respectively. TLNNA has the same performance with TLBO and NNA on 7 functions, which cannot surpass NNA and TLBO on 4(F_{13}, F_{20}, F_{28} and F_{29}) and 6(F_{13}, F_{19}, F_{22}, F_{24}, F_{27} and F_{29}) functions, respectively. Moreover, TLNNA is superior to GWO, SCA, WOA, SSA and PSO on 20, 28, 18, 23 and 28 functions, respectively. GWO and WOA only can offer better solutions than TLNNA on 3 (F_{13}, F_{20} and F_{27}) and 3 (F_8, F_{13} and F_{27}) functions, respectively. SCA only surpasses TLNNA on F_{13}. SSA and PSO cannot excel TLNNA on any function.

Based on No Free Lunch [52], there is no meta-heuristic best suited for solving all optimization problems. More specifically, a meta-heuristic algorithm may present very promising results on a set of optimization problems while it may show poor performance on another set of optimization problems. TLNNA can offer better solutions than the compared algorithms on most functions with all test dimensions, which indicates the proposed TLNNA has better global optimization performance than the compared algorithms.

4.2. Engineering design optimization problems

In this section, the performance of the proposed TLNNA is examined by solving four challenging engineering design optimization problems and the optimization results were compared with other optimizers. In the following experiments, the parameters were set as follows: (1) TLNNA, NNA and TLBO had the same number of function evaluations for every problem; (2) the population sizes of TLNNA, NNA and TLBO were set as 50, 50 and 50, respectively; (3) every engineering problem for TLNNA, NNA and TLBO was executed in 30 independent runs. All results are showed in Tables 8–11. In these tables, 'Worst', 'Mean', 'Best', 'STD' and 'NFEs' stand for the worst solution, the mean solution, the best solution, the standard deviation and the number of function evaluations, respectively. In addition, these engineering design problems are presented in Appendix.

4.2.1. Welded beam design problem

This is a classical engineering design problem (see Appendix A.1), which was proposed by Coello [53]. The goal of this problem is to design a welded beam that achieves the minimum cost subject to five constraints including shear stress, bending stress in the beam, buckling load on the bar, end deflection of the beam and side. In addition, there are four continuous design variables with five nonlinear and two linear inequality constraints.

This problem was optimized previously using GA2 [53], GA3 [54], CAEP [55], CPSO [56], HPSO [57], PSO-DE [58], NM-PSO [59], MGA [60], SC [61], DE [62], QS [3] and WCA. The comparisons of the statistical results are shown in Table 8.

From Table 8, TLNNA can obtain best solution of $f(\boldsymbol{x}) = 1.724852$ after 9000 function evaluations. Although CAEP, HPSO, PSO-DE and QS also can achieve the same best result with TLNNA, they need more function evaluations than TLNNA. In addition, TLNNA outperforms NNA and TLBO on Worst, Mean, Best and STD. Fig. 9 shows the convergence curves of the best optimal solutions obtained by NNA, TLBO and TLNNA for the welded beam design problem.

4.2.2. Speed reducer design problem

The aim of this problem (see Appendix A.2) is to minimize the weight of speed reducer subject to four constraints including bend stress of the gear teeth, surface stress, and transverse deflections of the shafts and stressed in the shafts [9]. In addition, there are seven continuous design variables with seven nonlinear and four linear inequality constraints.

The problem has been solved by SC, PSO-DE, DELC [63], DEDS [64], HEAA [65], FFA [66], MBA [67], CSA [68], PVS, QS and WCA. The comparisons of the statistical results are shown in Table 9.

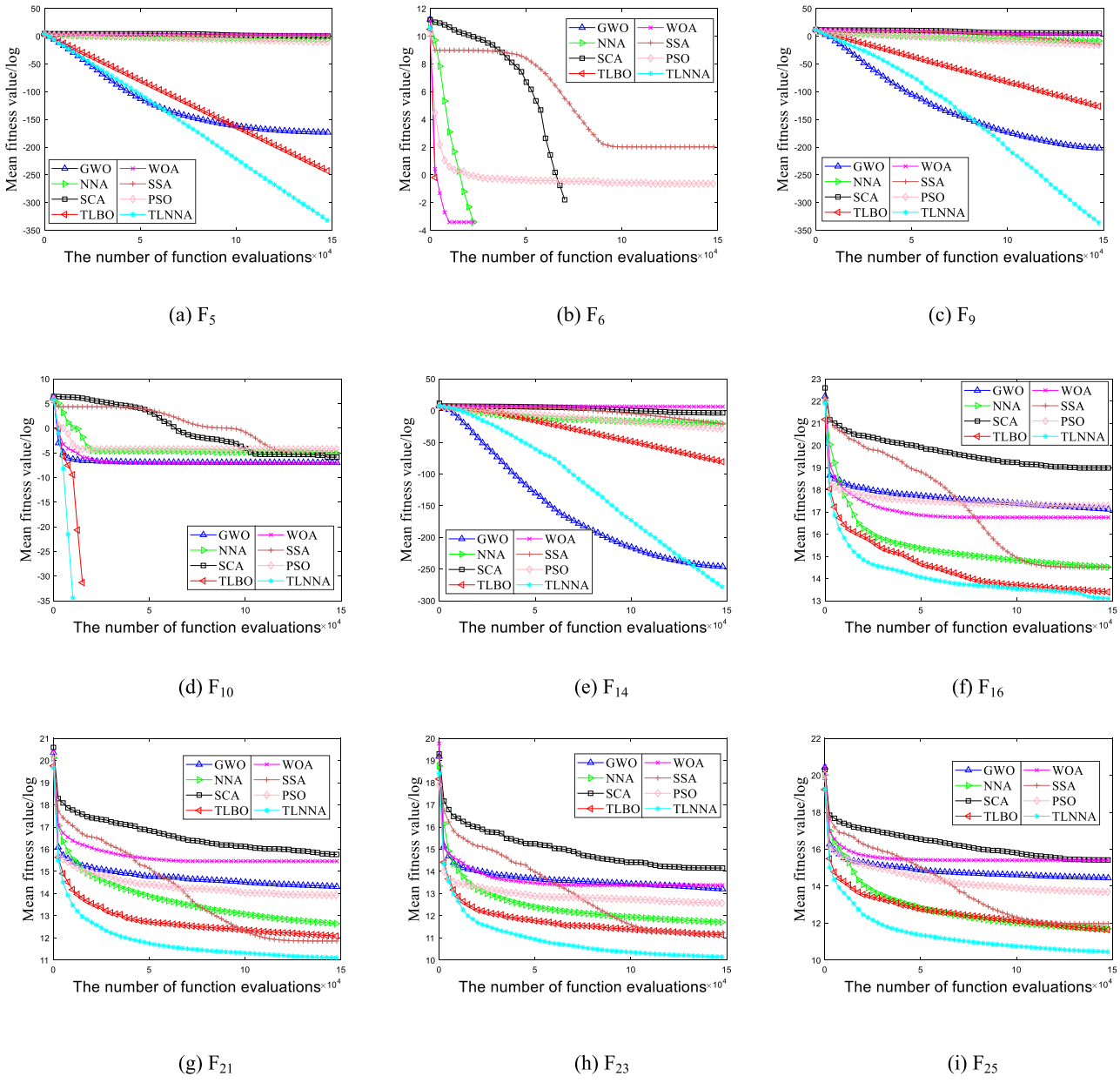

Fig. 6. The convergence curves of average fitness value on selected functions with 30*D*.

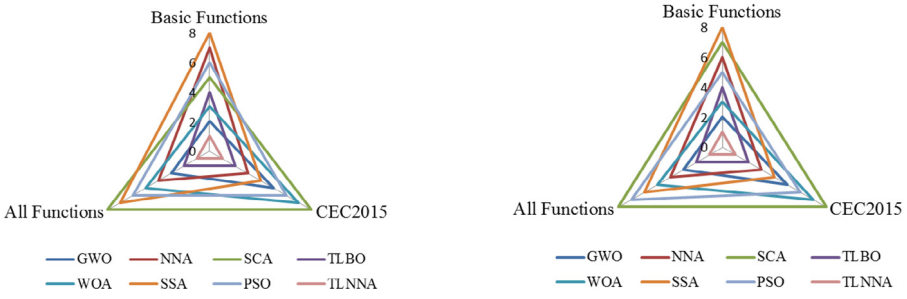

Fig. 7. The radar chart of the overall ranking on different types of functions.

From Table 9, TLNNA can achieve the best solution of $f(\boldsymbol{x}) = 2994.471066$ using 10,500 function evaluations. Compared with TLNNA, DELC and DEDS can obtain the same best solution while they consume nearly three times function evaluations compared with TLNNA. What is more, TLNNA surpasses NNA all four metrics. TLNNA is superior to TLBO on Best. Fig. 10 shows the convergence curves of the best optimal solutions obtained by NNA, TLBO and TLNNA for the speed reducer design problem.

Table 6
The comparison results of several algorithms for test functions with 10*D* by Wilcoxon signed-rank test (a level of significance $\alpha = 0.05$).

No.	TLNNA vs.																				
	GWO			NNA			SCA			TLBO			WOA			SSA			PSO		
	p-value	H	S	p-value	H	S	p-value	H	S	p-value	H	S	p-value	H	S	p-value	H	S	p-value	H	S
F_1	1.73E−6	1	−	1.73E−6	1	+	1.73E−6	1	+	1.73E−6	1	+	1.73E−6	1	−	1.73E−6	1	+	1.73E−6	1	+
F_2	1.73E−6	1	−	1.73E−6	1	+	1.73E−6	1	+	1.73E−6	1	+	1.73E−6	1	−	1.73E−6	1	+	1.73E−6	1	+
F_3	1.73E−6	1	−	1.73E−6	1	+	1.73E−6	1	+	1.73E−6	1	+	1.73E−6	1	−	1.73E−6	1	+	1.73E−6	1	+
F_4	1.73E−6	1	−	1.73E−6	1	+	1.73E−6	1	+	1.73E−6	1	+	1.73E−6	1	−	1.73E−6	1	+	1.73E−6	1	+
F_5	1.73E−6	1	−	1.73E−6	1	+	1.73E−6	1	+	1.73E−6	1	+	1.73E−6	1	+	1.73E−6	1	+	1.73E−6	1	+
F_6	1.00E+0	0	=	1.00E+0	0	=	1.00E+0	0	=	1.00E+0	0	=	1.00E+0	0	=	1.95E−3	1	+	1.00E+0	0	=
F_7	9.27E−3	1	−	2.83E−4	1	+	6.89E−5	1	+	1.71E−3	1	+	7.34E−1	0	=	1.73E−6	1	+	3.18E−6	1	+
F_8	1.73E−6	1	−	1.73E−6	1	+	1.73E−6	1	+	1.73E−6	1	+	1.73E−6	1	−	1.73E−6	1	+	1.73E−6	1	+
F_9	1.73E−6	1	−	1.73E−6	1	+	1.73E−6	1	+	1.73E−6	1	+	1.73E−6	1	+	1.73E−6	1	+	1.73E−6	1	+
F_{10}	1.29E−1	0	=	1.73E−6	1	+	3.91E−2	1	+	1.51E−3	1	+	2.34E−2	1	+	1.73E−6	1	+	2.35E−6	1	+
F_{11}	1.73E−6	1	−	1.73E−6	1	+	1.73E−6	1	+	1.73E−6	1	+	3.11E−5	1	−	1.73E−6	1	+	1.55E−6	1	+
F_{12}	7.10E−6	1	+	1.73E−6	1	+	9.67E−4	1	+	7.74E−6	1	+	1.05E−2	1	+	1.73E−6	1	+	6.12E−6	1	+
F_{13}	1.56E−2	1	−	1.81E−1	0	=	1.56E−2	1	−	1.11E−1	0	=	1.56E−2	1	−	1.73E−6	1	+	4.27E−6	1	+
F_{14}	1.73E−6	1	−	1.73E−6	1	+	1.73E−6	1	+	1.73E−6	1	+	1.73E−6	1	+	1.73E−6	1	+	1.73E−6	1	+
F_{15}	1.73E−6	1	+	4.88E−4	1	+	6.16E−4	1	+	1.73E−6	1	+	3.12E−2	1	+	1.72E−6	1	+	2.43E−4	1	+
F_{16}	1.73E−6	1	+	1.73E−6	1	+	1.73E−6	1	+	5.79E−5	1	+	1.73E−6	1	+	1.73E−6	1	+	1.73E−6	1	+
F_{17}	1.73E−6	1	+	8.45E−1	0	=	1.73E−6	1	+	1.66E−2	1	+	1.73E−6	1	+	5.86E−1	0	=	1.73E−6	1	+
F_{18}	1.73E−6	1	+	3.61E−3	1	−	1.73E−6	1	+	1.73E−6	1	+	4.73E−6	1	+	9.75E−1	0	=	1.73E−6	1	+
F_{19}	2.18E−2	1	+	5.71E−4	1	+	1.73E−6	1	+	2.71E−1	0	=	1.73E−6	1	+	5.71E−4	1	+	1.73E−6	1	+
F_{20}	3.71E−1	0	=	1.78E−1	0	=	1.73E−6	1	+	1.11E−3	1	+	1.73E−6	1	+	2.18E−2	1	+	2.35E−6	1	+
F_{21}	1.73E−6	1	+	3.88E−4	1	+	1.73E−6	1	+	1.83E−3	1	+	1.73E−6	1	+	4.20E−4	1	+	2.60E−5	1	+
F_{22}	8.47E−6	1	+	8.31E−4	1	+	1.73E−6	1	+	4.65E−1	0	=	1.73E−6	1	+	1.73E−6	1	+	1.73E−6	1	+
F_{23}	1.92E−6	1	+	3.11E−5	1	+	1.73E−6	1	+	7.52E−2	0	=	1.92E−6	1	+	2.88E−6	1	+	1.92E−6	1	+
F_{24}	1.96E−2	1	+	1.96E−2	1	+	1.73E−6	1	+	7.97E−1	0	=	4.86E−5	1	+	1.92E−1	0	=	2.30E−2	1	+
F_{25}	2.88E−6	1	+	2.60E−5	1	+	1.73E−6	1	+	9.63E−4	1	+	1.73E−6	1	+	1.73E−6	1	+	1.73E−6	1	+
F_{26}	4.07E−2	1	−	2.83E−4	1	+	1.64E−5	1	+	4.65E−1	0	=	1.49E−5	1	+	1.48E−4	1	+	1.73E−6	1	+
F_{27}	9.26E−1	0	=	7.69E−6	1	+	1.73E−6	1	+	3.93E−1	0	=	2.35E−6	1	+	1.97E−5	1	+	1.92E−6	1	+
F_{28}	1.53E−1	0	=	8.94E−1	0	=	1.73E−6	1	+	7.50E−1	0	=	1.73E−6	1	+	1.16E−1	0	=	1.73E−6	1	+
F_{29}	5.31E−5	1	+	3.49E−1	0	=	4.86E−5	1	+	2.56E−2	1	−	2.88E−6	1	+	1.29E−3	1	+	1.89E−4	1	−
F_{30}	1.73E−6	1	+	1.73E−6	1	+	1.73E−6	1	+	6.10E−5	1	−	1.73E−6	1	+	1.73E−6	1	+	1.73E−6	1	+
$w/t/l$		13/5/12			23/6/1			28/1/1			19/9/2			21/2/7			26/4/0			28/1/1	

Table 7
The comparison results of several algorithms for test functions with 30D by Wilcoxon signed-rank test (a level of significance $\alpha = 0.05$).

No.	TLNNA vs.																				
	GWO			NNA			SCA			TLBO			WOA			SSA			PSO		
	p-value	H	S	p-value	H	S	p-value	H	S	p-value	H	S	p-value	H	S	p-value	H	S	p-value	H	S
F_1	1.73E−6	1	+	1.73E−6	1	+	1.73E−6	1	+	1.73E−6	1	+	1.00E+0	0	=	1.73E−6	1	+	1.73E−6	1	+
F_2	1.73E−6	1	+	1.73E−6	1	+	1.73E−6	1	+	1.73E−6	1	+	1.00E+0	0	=	1.73E−6	1	+	1.73E−6	1	+
F_3	1.73E−6	1	+	1.73E−6	1	+	1.73E−6	1	+	1.73E−6	1	+	5.00E−1	0	=	1.73E−6	1	+	1.73E−6	1	+
F_4	1.00E+0	0	=	1.73E−6	1	+	1.73E−6	1	+	1.00E+0	0	=	1.00E+0	0	=	1.73E−6	1	+	1.73E−6	1	+
F_5	1.73E−6	1	+	1.73E−6	1	+	1.73E−6	1	+	1.73E−6	1	+	1.73E−6	1	+	1.73E−6	1	+	1.73E−6	1	+
F_6	1.00E+0	0	=	1.00E+0	0	=	1.00E+0	0	=	1.00E+0	0	=	1.00E+0	0	=	1.60E−6	1	+	7.81E−3	1	+
F_7	8.97E−2	0	=	1.73E−6	1	+	1.73E−6	1	+	1.73E−6	1	+	3.61E−3	1	+	1.73E−6	1	+	1.73E−6	1	+
F_8	8.61E−1	0	=	1.73E−6	1	+	1.73E−6	1	+	1.73E−6	1	+	1.73E−6	1	−	1.73E−6	1	+	1.73E−6	1	+
F_9	1.73E−6	1	+	1.73E−6	1	+	1.73E−6	1	+	1.73E−6	1	+	1.73E−6	1	+	1.73E−6	1	+	1.73E−6	1	+
F_{10}	2.50E−1	0	=	8.83E−5	1	+	3.91E−3	1	+	1.00E+0	0	=	5.00E−1	0	=	1.73E−6	1	+	2.66E−5	1	+
F_{11}	1.73E−6	1	+	1.73E−6	1	+	1.73E−6	1	+	1.73E−6	1	+	1.00E+0	0	=	1.73E−6	1	+	1.65E−6	1	+
F_{12}	1.69E−7	1	+	1.73E−6	1	+	1.72E−6	1	+	1.95E−3	1	+	6.17E−1	0	=	1.73E−6	1	+	1.67E−6	1	+
F_{13}	1.23E−5	1	−	7.69E−6	1	−	9.34E−6	1	−	1.33E−2	1	−	1.23E−5	1	−	1.73E−6	1	+	1.73E−6	1	+
F_{14}	1.73E−6	1	+	1.73E−6	1	+	1.73E−6	1	+	1.73E−6	1	+	1.73E−6	1	+	1.73E−6	1	+	1.73E−6	1	+
F_{15}	1.73E−6	1	+	3.91E−3	1	+	1.73E−6	1	+	1.73E−6	1	+	5.15E−2	0	=	1.72E−6	1	+	1.37E−6	1	+
F_{16}	1.73E−6	1	+	3.52E−6	1	+	1.73E−6	1	+	3.93E−1	0	=	1.73E−6	1	+	1.73E−6	1	+	1.73E−6	1	+
F_{17}	1.73E−6	1	+	1.25E−1	0	=	1.73E−6	1	+	9.43E−1	0	=	1.73E−6	1	+	3.71E−1	0	=	1.73E−6	1	+
F_{18}	1.73E−6	1	+	1.71E−1	0	=	1.73E−6	1	+	1.73E−6	1	+	1.73E−6	1	+	3.18E−1	0	=	1.73E−6	1	+
F_{19}	7.81E−1	0	=	2.60E−5	1	+	1.73E−6	1	+	4.95E−2	1	−	1.73E−6	1	+	1.64E−5	1	+	1.73E−6	1	+
F_{20}	1.24E−5	1	−	2.41E−3	1	−	1.73E−6	1	+	1.73E−6	1	+	1.83E−3	1	+	2.99E−1	0	=	1.73E−6	1	+
F_{21}	1.73E−6	1	+	2.35E−6	1	+	1.73E−6	1	+	9.71E−5	1	+	1.73E−6	1	+	4.68E−3	1	+	1.73E−6	1	+
F_{22}	2.35E−6	1	+	4.99E−3	1	+	1.73E−6	1	+	3.50E−2	1	−	1.73E−6	1	+	5.79E−5	1	+	1.73E−6	1	+
F_{23}	1.73E−6	1	+	3.18E−6	1	+	1.73E−6	1	+	1.49E−5	1	+	1.73E−6	1	+	3.52E−6	1	+	1.73E−6	1	+
F_{24}	2.37E−5	1	+	8.97E−2	0	=	1.73E−6	1	+	7.51E−5	1	−	1.92E−6	1	+	5.45E−2	0	=	1.73E−6	1	+
F_{25}	1.73E−6	1	+	1.60E−4	1	+	1.73E−6	1	+	1.02E−5	1	+	1.73E−6	1	+	1.73E−6	1	+	1.73E−6	1	+
F_{26}	8.29E−1	0	=	1.85E−1	0	=	8.19E−5	1	+	5.04E−1	0	=	7.69E−6	1	+	3.49E−1	0	=	5.04E−1	0	=
F_{27}	7.51E−5	1	−	1.16E−1	0	=	1.96E−2	1	+	7.71E−4	1	−	7.73E−3	1	−	1.36E−5	1	+	1.73E−6	1	+
F_{28}	2.35E−6	1	+	6.42E−3	1	−	1.73E−6	1	+	6.14E−1	0	=	2.11E−3	1	+	4.53E−4	1	+	1.73E−6	1	+
F_{29}	9.84E−3	1	+	8.22E−3	1	−	1.73E−6	1	+	1.48E−4	1	−	1.60E−4	1	+	7.86E−2	0	=	2.80E−1	0	=
F_{30}	1.73E−6	1	+	1.71E−1	0	=	1.73E−6	1	+	1.48E−2	1	+	1.73E−6	1	+	5.71E−2	0	=	1.73E−6	1	+
$w/t/l$		20/7/3			19/7/4			28/1/1			17/7/6			18/9/3			23/7/0			28/2/0	

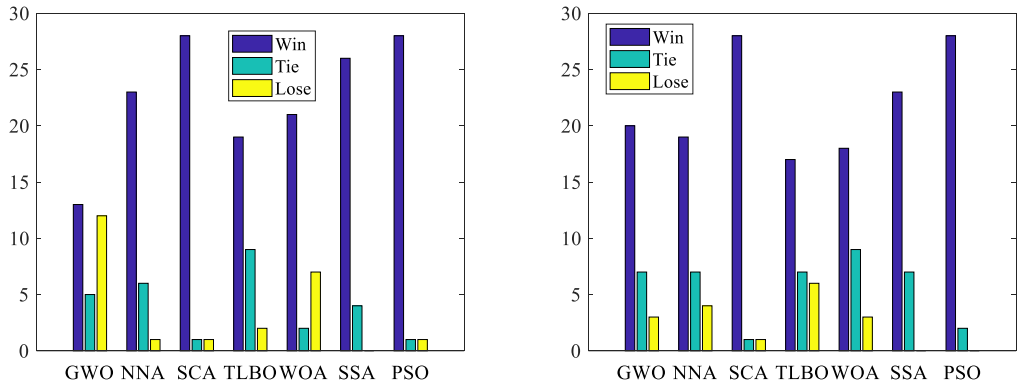

Fig. 8. The statistical results based on Wilcoxon signed-rank test.

Table 8
The statistical results obtained by different optimizers for the welded beam design problem.

Algorithm	Worst	Mean	Best	STD	NFEs
GA2 [53]	1.785835	1.771973	1.748309	1.12E−02	900,000
GA3 [54]	1.993408	1.792654	1.728226	7.47E−02	80,000
CAEP [55]	3.179709	1.971809	1.724852	4.43E−01	50,020
CPSO [56]	1.782143	1.748831	1.728024	1.29E−02	240,000
HPSO [57]	1.814295	1.749040	1.724852	4.01E−02	81,000
PSO-DE [58]	1.724852	1.724852	1.724852	6.70E−16	66,600
NM-PSO [59]	1.733393	1.726373	1.724717	3.50E−03	80,000
MGA [60]	1.995000	1.919000	1.824500	5.37E−02	NA
SC [61]	6.399678	3.002588	2.385434	9.60E−01	33,095
DE [62]	1.824105	1.768158	1.733461	2.21E−02	204,800
WCA [13]	1.744697	1.726427	1.724856	4.29E−03	46,450
QS [3]	1.724852	1.724852	1.724852	NA	20,000
NNA	2.626234	1.839735	1.736676	1.39E−01	9,000
TLBO	1.726209	1.725072	1.724865	2.38E−04	9,000
TLNNA	1.724952	1.724866	1.724852	2.09E−05	9,000

Table 9
The statistical results obtained by different optimizers for the speed reducer design problem.

Algorithm	Worst	Mean	Best	STD	NFEs
SC [61]	3009.964736	3001.758264	2994.744241	4.0E+1	54,456
PSO-DE [58]	2996.348204	2996.348174	2996.348167	6.4E−06	54,350
DELC [63]	2994.471066	2994.471066	2994.471066	1.9E−12	30,000
DEDS [64]	2994.471066	2994.471066	2994.471066	3.6E−12	30,000
HEAA [65]	2994.752311	2994.613368	2994.499107	7.0E−02	40,000
FFA [66]	2996.669	2996.51	2996.37	NA	50,000
MBA [67]	2999.65	2996.769	2994.4824	NA	6,300
CSA [68]	3009	3007.1997	3000.98	NA	5,000
PVS [12]	2996.348165	2996.348165	2996.348165	NA	54,350
WCA [13]	2994.505578	2994.474392	2994.47166	7.4E−03	15,150
QS [3]	2996.348165	2996.348165	2996.348165	NA	25,000
NNA	3026.919910	3010.583596	2997.510246	6.6E+00	10,500
TLBO	2994.471790	2994.471141	2994.471074	1.2E−04	10,500
TLNNA	2994.474519	2994.471175	2994.471066	5.4E−04	10,500

4.2.3. Pressure vessel design problem

This problem (see Appendix A.3) is to minimize the total cost including the cost of material, forming and welding. In addition, there are four continuous design variables with one nonlinear and three linear inequality constraints.

The problem was optimized previously using GA2, GA3, CPSO, HPSO, NM-PSO, PSO, BA [68], CSA, ISA, FFA, PVS, QS and WCA. The comparisons of the statistical results are shown in Table 10.

From Table 10, TLNNA can obtain the best solution of $f(\boldsymbol{x}) = 5885.333$ using 13,500 function evaluations. Although PVS and WCA achieve the same best solution with TLNNA, the number of function evaluations of PVS and WCA are 20,000 and 27,500, respectively. In addition, TLNNA is superior to NNA on Worst, Mean, Best and STD. TLBO is inferior to TLNNA on Best. Fig. 11 shows the convergence curves of the best optimal solutions obtained by NNA, TLBO and TLNNA for the pressure reducer design problem.

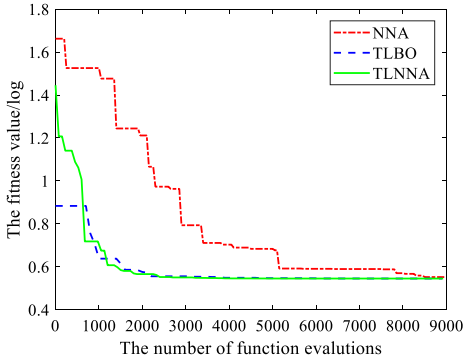

Fig. 9. The convergence curves for the welded beam design problem.

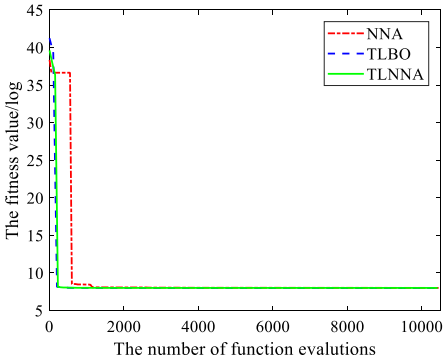

Fig. 10. The convergence curves for the speed reducer design problem.

4.2.4. Tension/comparison spring design problem

This problem (see Appendix A.4) is to minimize the weight of a tension/compression spring subject to the constraints including minimum deflection, shear stress, surge frequency and limits on outside diameter. In addition, there are three continuous design variables with three nonlinear and one linear inequality constraints.

The problem was optimized previously using GA2, GA3, CPSO, HPSO, NM-PSO, PSO, CAEP, DE, DELC, DEDS, HEAA, PSO-DE, PVS, QS and WCA. The comparisons of the statistical results are shown in Table 11.

Table 10
The statistical results obtained by different optimizers for the pressure vessel design problem.

Algorithm	Worst	Mean	Best	STD	NFEs
GA2 [53]	6308.4970	6293.8432	6288.7445	7.41	900,000
GA3 [54]	6469.3220	6177.2533	6059.9463	130.9297	80,000
CPSO [56]	6363.8041	6147.1332	6061.0777	86.4500	240,000
HPSO [57]	6288.6770	6099.9323	6059.7143	86.2000	81,000
NM-PSO [59]	5960.0557	5946.7901	5930.3137	9.1610	80,000
PSO [58]	14076.3240	8756.6803	6693.7212	1492.5670	8,000
BA [68]	6318.95	6179.13	6059.714	NA	20,000
CSA [68]	6495.34	6447.73	6059.714	NA	15,000
ISA [68]	7332.84	6410.087	6059.714	NA	5,000
FFA [66]	6090.52	6064.33	6059.714	NA	50,000
PVS [12]	5886.035	5885.409	5885.333	NA	20,000
WCA [13]	6590.2129	6198.6172	5885.333	213.0490	27,500
QS [3]	6090.526	6060.947	6059.714	NA	20,000
NNA	7375.078	6365.521	5887.252	484.5360	13,500
TLBO	7011.842	5964.996	5885.364	176.3181	13,500
TLNNA	6114.958	5935.423	5885.333	66.28772	13,500

Table 11
The statistical results obtained from different optimizers for the tension/comparison spring design problem.

Algorithm	Worst	Mean	Best	STD	NFEs
GA2 [53]	0.012822	0.012769	0.012704	3.94E−05	900,000
GA3 [54]	0.012973	0.012742	0.012681	5.90E−05	80,000
CPSO [56]	0.012924	0.012730	0.012674	5.20E−04	240,000
HPSO [58]	0.012719	0.012707	0.012665	1.58E−05	81,000
NM-PSO [59]	0.012633	0.012631	0.012630	8.47E−07	80,000
PSO [32]	0.071802	0.019555	0.012857	1.17E−02	2,000
CAEP [55]	0.015116	0.013568	0.012721	8.42E−04	50,020
DE [62]	0.012790	0.012703	0.012670	2.70E−05	204,800
DELC [63]	0.012665	0.012665	0.012665	1.30E−07	20,000
DEDS [64]	0.012738	0.012669	0.012665	1.30E−05	24,000
HEAA [65]	0.012665	0.012665	0.012665	1.40E−09	24,000
PSO-DE [58]	0.012665	0.012665	0.012665	1.20E−08	24,950
WCA [13]	0.012952	0.012746	0.012665	8.06E−05	11,750
PVS [12]	0.012710	0.012670	0.012665	NA	8,000
QS [3]	0.012828	0.012691	0.012665	NA	8,000
NNA	0.017773	0.014490	0.012703	1.83E−03	18,000
TLBO	0.012827	0.012702	0.012666	3.78E−05	18,000
TLNNA	0.012836	0.012689	0.012665	3.24E−05	18,000

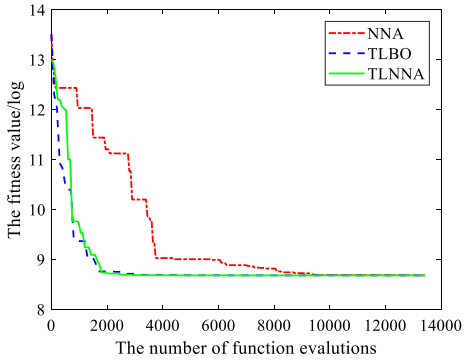

Fig. 11. The convergence curves for the pressure vessel design problem.

From Table 11, TLNNA, HPSO, DELC, DEDS, HEAA, PSO-DE, WCA, PVS and QS can achieve the best solution of $f(\boldsymbol{x}) = 0.012665$. In addition, TLNNA is superior to NNA on Worst, Mean, Best and STD. TLBO is inferior to TLNNA on Mean, Best and STD. Fig. 12 shows the convergence curves of the best optimal solutions obtained by NNA, TLBO and TLNNA for the tension/compression spring design problem.

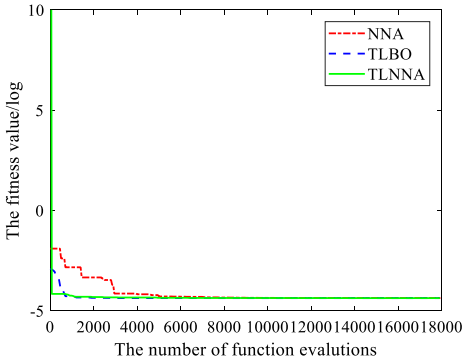

Fig. 12. The convergence curves for the tension/comparison spring design problem.

5. Conclusions

This paper presents an efficient hybrid method based on teaching–learning-based optimization and neural network algorithm, named TLNNA. The core idea of TLNNA is to make full use of the good global search ability of NNA and fast convergence of TLBO. The proposed TLNNA mainly includes three stages: TLBO stage, NNA stage and dynamic grouping stage. More specifically, in one loop, the population is firstly divided into two halves based on the fitness values of individuals in the dynamic grouping. Then the best half of population is optimized by TLBO and the worst half of population is processed by NNA. Finally, the optimal solution is selected.

In order to examine the performance of the proposed algorithm, TLNNA is firstly used to solve 30 well-known unconstrained benchmark functions including 15 basic functions and CEC2015 test suite. The results are compared with 7 competitive algorithms including TLBO, NNA, SSA, SCA, PSO, WOA and GWO. Such comparisons suggest that TLNNA is more successful on most test functions in terms of solution quality and computational efficiency. Then 4 challenging engineering design optimization problems are solved by the proposed TLNNA. The obtained results are compared with other reported optimizers. The experimental results show that the proposed TLNNA can effectively achieve the best solutions for all problems.

TLNNA is a simple and effective algorithm without any effort for fine tuning initial parameters. Thus TLNNA has the potential to be used to solve different types of optimization problems. Further research will intend to apply TLNNA to multilevel image segmentation and economic load dispatch.

Appendix

A.1. Welded beam design problem

$$f(x) = 1.10471x_1^2x_2 + 0.04811x_3x_4(14 + x_2)$$

subject to:
$g_1(x) = \tau(x) - \tau_{\max} \le 0$
$g_2(x) = \sigma(x) - \sigma_{\max} \le 0$
$g_3(x) = x_1 - x_4 \le 0$
$g_4(x) = 0.10471x_1^2 + 0.04811x_3x_4(14 + x_2) - 5 \le 0$
$g_5(x) = 0.125 - x_1 \le 0$
$g_6(x) = \delta(x) - \delta_{\max} \le 0$
$g_7(x) = P - P_c(x) \le 0$
$0.1 \le x_i \le 2 \quad i = 1, 4$
$0.1 \le x_i \le 10 \quad i = 2, 3$

where,

$$\tau(x) = \sqrt{(\tau')^2 + 2\tau'\tau''\frac{x_2}{2R} + (\tau'')^2}, \tau' = \frac{P}{\sqrt{2}x_1x_2}, \tau'' = \frac{MR}{J},$$

$$M = P\left(L + \frac{x_2}{2}\right), R = \sqrt{\left(\frac{x_2}{2}\right)^2 + \left(\frac{x_1 + x_3}{2}\right)^2},$$

$$J = 2\left(\sqrt{2}x_1x_2\left(\frac{x_2^2}{12} + \left(\frac{x_1 + x_3}{2}\right)^2\right)\right), \sigma(x) = \frac{6PL}{x_4x_3^2}, \delta(x) = \frac{4PL^3}{Ex_3^3x_4},$$

$$P_c(x) = \frac{4.013E\sqrt{\frac{x_3^2x_4^6}{36}}}{L^2}\left(1 - \frac{x_3}{2L}\sqrt{\frac{E}{4G}}\right)$$

$P = 6000$ lb, $L = 14$ in, $E = 30 \times 10^6$ psi, $G = 12 \times 10^6$ psi,

$\tau_{max} = 13{,}600$ psi,

$\sigma_{max} = 30{,}000$ psi, $\delta_{max} = 0.25$ in.

A.2. Speed reducer design problem

$$f(x) = 0.7854x_1x_2^2\left(3.3333x_3^2 + 14.9334x_3 - 43.0934\right) - 1.508x_1\left(x_6^2 + x_7^2\right) + 7.4777\left(x_6^3 + x_7^3\right) + 0.7854\left(x_4x_6^2 + x_5x_7^2\right)$$

subject to:

$$g_1(x) = \frac{27}{x_1x_2^2x_3} - 1 \le 0$$

$$g_2(x) = \frac{397.5}{x_1x_2^2x_3} - 1 \le 0$$

$$g_3(x) = \frac{1.93x_4^3}{x_2x_6^4x_3} - 1 \le 0$$

$$g_4(x) = \frac{1.93x_5^3}{x_2x_7^4x_3} - 1 \le 0$$

$$g_5(x) = \frac{\left(\left(\frac{745x_4}{x_2x_3}\right)^2 + 16.9 \times 10^6\right)^{1/2}}{110x_6^3} - 1 \le 0$$

$$g_6(x) = \frac{\left(\left(\frac{745x_5}{x_2x_3}\right)^2 + 157.5 \times 10^6\right)^{1/2}}{85x_7^3} - 1 \le 0$$

$$g_7(x) = \frac{x_2x_3}{40} - 1 \le 0$$

$$g_8(x) = \frac{5x_2}{x_1} - 1 \le 0$$

$$g_9(x) = \frac{x_1}{12x_2} - 1 \le 0$$

$$g_{10}(x) = \frac{1.5x_6 + 1.9}{x_4} - 1 \le 0$$

$$g_{11}(x) = \frac{1.1x_7 + 1.9}{x_5} - 1 \le 0$$

where,

$2.6 \le x_1 \le 3.6, 0.7 \le x_2 \le 0.8, 17 \le x_3 \le 28, 7.3 \le x_4 \le 8.3, 7.3 \le x_5 \le 8.3, 2.9 \le x_6 \le 3.9, 5.0 \le x_7 \le 5.5$

A.3. Pressure vessel design problem

$$f(x) = 0.6224x_1x_3x_4 + 1.7781x_2x_3^2 + 3.1661x_1^2x_4 + 19.84x_1^2x_3$$

subject to:

$$g_1(x) = -x_1 + 0.0193x_3 \le 0$$

$$g_2(x) = -x_2 + 0.00954x_3 \le 0$$

$$g_3(x) = -\pi x_3^2x_4 - \frac{4}{3}\pi x_3^3 + 1296{,}000 \le 0$$

$$g_4(x) = x_4 - 240 \le 0$$

where,

$0 \le x_i \le 100, i = 1, 2$

$10 \le x_i \le 200, i = 3, 4$

A.4. Tension/comparison spring design problem

$$f(x) = (x_3 + 2)x_2x_1^2$$

subject to:

$$g_1(x) = 1 - \frac{x_2^3x_3}{71{,}785x_1^4} \le 0$$

$$g_2(x) = 4x_2^2 - \frac{x_1x_2}{12.566\left(x_2x_1^3 - x_1^4\right)} + \frac{1}{5108x_1^2} - 1 \le 0$$

$$g_3(x) = 1 - \frac{140.45x_1}{x_2^2x_3} \le 0$$

$$g_4(x) = x_2 + \frac{x_1}{1.5} - 1 \le 0$$

where,

$0.05 \le x_1 \le 2, 0.25 \le x_2 \le 1.30, 2.00 \le x_3 \le 15.00$

References

[1] R.V. Rao, V.J. Savsani, D.P. Vakharia, Teaching–learning-based optimization: A novel method for constrained mechanical design optimization problems, Comput.-Aided Des. 43 (2011) 303–315, http://dx.doi.org/10.1016/j.cad.2010.12.015.

[2] R.V. Rao, V.J. Savsani, D.P. Vakharia, Teaching–learning-based optimization: An optimization method for continuous non-linear large scale problems, Inform. Sci. 183 (2012) 1–15, http://dx.doi.org/10.1016/j.ins.2011.08.006.

[3] J. Zhang, M. Xiao, L. Gao, Q. Pan, Queuing search algorithm: A novel metaheuristic algorithm for solving engineering optimization problems, Appl. Math. Model. 63 (2018) 464–490, http://dx.doi.org/10.1016/j.apm.2018.06.036.

[4] R. Storn, K. Price, Differential evolution – A simple and efficient heuristic for global optimization over continuous spaces, J. Global Optim. 11 (1997) 341–359, http://dx.doi.org/10.1023/A:1008202821328.

[5] D. Simon, Biogeography-based optimization, IEEE Trans. Evol. Comput. 12 (2008) 702–713, http://dx.doi.org/10.1109/TEVC.2008.919004.

[6] X. Yang, S. Deb, Cuckoo search via Lévy flights, in: 2009 World Congr. Nat. Biol. Inspired Comput. NaBIC, 2009, pp. 210–214, http://dx.doi.org/10.1109/NABIC.2009.5393690.

[7] S. Mirjalili, S.M. Mirjalili, A. Lewis, Grey wolf optimizer, Adv. Eng. Softw. 69 (2014) 46–61, http://dx.doi.org/10.1016/j.advengsoft.2013.12.007.

[8] S. Mirjalili, A. Lewis, The whale optimization algorithm, Adv. Eng. Softw. 95 (2016) 51–67, http://dx.doi.org/10.1016/j.advengsoft.2016.01.008.

[9] S. Mirjalili, A.H. Gandomi, S.Z. Mirjalili, S. Saremi, H. Faris, S.M. Mirjalili, Salp swarm algorithm: A bio-inspired optimizer for engineering design problems, Adv. Eng. Softw. 114 (2017) 163–191, http://dx.doi.org/10.1016/j.advengsoft.2017.07.002.

[10] J. Kennedy, R. Eberhart, Particle swarm optimization, in: Proc. ICNN95 - Int. Conf. Neural Netw., Vol. 4, 1995, pp. 1942–1948, http://dx.doi.org/10.1109/ICNN.1995.488968.

[11] S. Mirjalili, SCA: A Sine cosine algorithm for solving optimization problems, Knowl.-Based Syst. 96 (2016) 120–133, http://dx.doi.org/10.1016/j.knosys.2015.12.022.

[12] P. Savsani, V. Savsani, Passing vehicle search (PVS): A novel metaheuristic algorithm, Appl. Math. Model. 40 (2016) 3951–3978, http://dx.doi.org/10.1016/j.apm.2015.10.040.

[13] H. Eskandar, A. Sadollah, A. Bahreininejad, M. Hamdi, Water cycle algorithm – A novel metaheuristic optimization method for solving constrained engineering optimization problems, Comput. Struct. 110–111 (2012) 151–166, http://dx.doi.org/10.1016/j.compstruc.2012.07.010.

[14] A. Kaveh, T. Bakhshpoori, Water evaporation optimization: A novel physically inspired optimization algorithm, Comput. Struct. 167 (2016) 69–85, http://dx.doi.org/10.1016/j.compstruc.2016.01.008.

[15] X. Yao, Z. Wang, H. Zhang, A novel photovoltaic power forecasting model based on echo state network, Neurocomputing. 325 (2019) 182–189, http://dx.doi.org/10.1016/j.neucom.2018.10.022.

[16] L. Liu, Z. Wang, H. Zhang, Adaptive fault-tolerant tracking control for MIMO discrete-time systems via reinforcement learning algorithm with less learning parameters, IEEE Trans. Autom. Sci. Eng. 14 (2017) 299–313, http://dx.doi.org/10.1109/TASE.2016.2517155.

[17] D. Varshney, S. Kumar, V. Gupta, Predicting information diffusion probabilities in social networks: A Bayesian networks based approach, Knowl.-Based Syst. 133 (2017) 66–76, http://dx.doi.org/10.1016/j.knosys.2017.07.003.

[18] H. Zhang, Z. Liu, G. Huang, Z. Wang, Novel weighting-delay-based stability criteria for recurrent neural networks with time-varying delay, IEEE Trans. Neural Netw. 21 (2010) 91–106, http://dx.doi.org/10.1109/TNN.2009.2034742.

[19] A.L. Madsen, F. Jensen, A. Salmerón, H. Langseth, T.D. Nielsen, A parallel algorithm for Bayesian network structure learning from large data sets, Var. Veloc. Data Sci. 117 (2017) 46–55, http://dx.doi.org/10.1016/j.knosys.2016.07.031.

[20] H. Zhang, Y. Wang, Stability analysis of Markovian jumping stochastic Cohen–Grossberg neural networks with mixed time delays, IEEE Trans. Neural Netw. 19 (2008) 366–370, http://dx.doi.org/10.1109/TNN.2007.910738.

[21] M.H. Abolbashari, E. Chang, O.K. Hussain, M. Saberi, Smart buyer: A Bayesian network modelling approach for measuring and improving procurement performance in organisations, Knowl.-Based Syst. 142 (2018) 127–148, http://dx.doi.org/10.1016/j.knosys.2017.11.032.

[22] H. Zhang, T. Ma, G. Huang, Z. Wang, Robust global exponential synchronization of uncertain chaotic delayed neural networks via dual-stage impulsive control, IEEE Trans. Syst. Man Cybern. B 40 (2010) 831–844, http://dx.doi.org/10.1109/TSMCB.2009.2030506.

[23] H. Fujita, D. Cimr, Computer aided detection for fibrillations and flutters using deep convolutional neural network, Inform. Sci. 486 (2019) 231–239, http://dx.doi.org/10.1016/j.ins.2019.02.065.

[24] C. Zhang, J. Bi, S. Xu, E. Ramentol, G. Fan, B. Qiao, H. Fujita, Multi-imbalance: An open-source software for multi-class imbalance learning, Knowl.-Based Syst. (2019) http://dx.doi.org/10.1016/j.knosys.2019.03.001.

[25] X. Yang, T. Li, D. Liu, H. Fujita, A temporal-spatial composite sequential approach of three-way granular computing, Inform. Sci. 486 (2019) 171–189, http://dx.doi.org/10.1016/j.ins.2019.02.048.

[26] X. Yao, Z. Wang, H. Zhang, Identification method for a class of periodic discrete-time dynamic nonlinear systems based on Sinusoidal ESN, Neurocomputing. 275 (2018) 1511–1521, http://dx.doi.org/10.1016/j.neucom.2017.09.092.

[27] L. Zhao, Y. Zhou, H. Lu, H. Fujita, Parallel computing method of deep belief networks and its application to traffic flow prediction, Knowl.-Based Syst. 163 (2019) 972–987, http://dx.doi.org/10.1016/j.knosys.2018.10.025.

[28] A. Sadollah, H. Sayyaadi, A. Yadav, A dynamic metaheuristic optimization model inspired by biological nervous systems: Neural network algorithm, Appl. Soft Comput. 71 (2018) 747–782, http://dx.doi.org/10.1016/j.asoc.2018.07.039.

[29] X. Chen, C. Mei, B. Xu, K. Yu, X. Huang, Quadratic interpolation based teaching-learning-based optimization for chemical dynamic system optimization, Knowl.-Based Syst. 145 (2018) 250–263, http://dx.doi.org/10.1016/j.knosys.2018.01.021.

[30] J. Huang, L. Gao, X. Li, An effective teaching-learning-based cuckoo search algorithm for parameter optimization problems in structure designing and machining processes, Appl. Soft Comput. 36 (2015) 349–356, http://dx.doi.org/10.1016/j.asoc.2015.07.031.

[31] A. Birashk, J. Kazemi Kordestani, M.R. Meybodi, Cellular teaching-learning-based optimization approach for dynamic multi-objective problems, Knowl.-Based Syst. 141 (2018) 148–177, http://dx.doi.org/10.1016/j.knosys.2017.11.016.

[32] M. Abirami, S. Ganesan, S. Subramanian, R. Anandhakumar, Source and transmission line maintenance outage scheduling in a power system using teaching learning based optimization algorithm, Appl. Soft Comput. 21 (2014) 72–83, http://dx.doi.org/10.1016/j.asoc.2014.03.015.

[33] B.-C. Wang, H.-X. Li, Y. Feng, An improved teaching-learning-based optimization for constrained evolutionary optimization, Inform. Sci. 456 (2018) 131–144, http://dx.doi.org/10.1016/j.ins.2018.04.083.

[34] Z. Yang, K. Li, Y. Guo, H. Ma, M. Zheng, Compact real-valued teaching-learning based optimization with the applications to neural network training, Knowl.-Based Syst. 159 (2018) 51–62, http://dx.doi.org/10.1016/j.knosys.2018.06.004.

[35] H. Ouyang, L. Gao, X. Kong, D. Zou, S. Li, Teaching-learning based optimization with global crossover for global optimization problems, Appl. Math. Comput. 265 (2015) 533–556, http://dx.doi.org/10.1016/j.amc.2015.05.012.

[36] D. Chen, F. Zou, Z. Li, J. Wang, S. Li, An improved teaching–learning-based optimization algorithm for solving global optimization problem, Inform. Sci. 297 (2015) 171–190, http://dx.doi.org/10.1016/j.ins.2014.11.001.

[37] H. Rakhshani, A. Rahati, Snap-drift cuckoo search: A novel cuckoo search optimization algorithm, Appl. Soft Comput. 52 (2017) 771–794, http://dx.doi.org/10.1016/j.asoc.2016.09.048.

[38] N.J. Cheung, H.-B. Shen, Hierarchical particle swarm optimizer for minimizing the non-convex potential energy of molecular structure, J. Mol. Graph. Model. 54 (2014) 114–122, http://dx.doi.org/10.1016/j.jmgm.2014.10.002.

[39] Y. Sun, X. Wang, Y. Chen, Z. Liu, A modified whale optimization algorithm for large-scale global optimization problems, Expert Syst. Appl. 114 (2018) 563–577, http://dx.doi.org/10.1016/j.eswa.2018.08.027.

[40] C. Lu, L. Gao, J. Yi, Grey wolf optimizer with cellular topological structure, Expert Syst. Appl. 107 (2018) 89–114, http://dx.doi.org/10.1016/j.eswa.2018.04.012.

[41] J. Liang, B. Qu, P. Suganthan, Q. Chen, Problem definitions and evaluation criteria for the CEC 2015 competition on learning-based real-parameter single objective optimization, Tech. Report201411A Comput. Intell. Lab. Zhengzhou Univ. Zhengzhou China Tech. Rep. Nanyang Technol. Univ. Singap. (2014).

[42] R.A. Ibrahim, M.A. Elaziz, S. Lu, Chaotic opposition-based grey-wolf optimization algorithm based on differential evolution and disruption operator for global optimization, Expert Syst. Appl. 108 (2018) 1–27, http://dx.doi.org/10.1016/j.eswa.2018.04.028.

[43] X. Zhang, Q. Kang, J. Cheng, X. Wang, A novel hybrid algorithm based on Biogeography-Based Optimization and Grey Wolf Optimizer, Appl. Soft Comput. 67 (2018) 197–214, http://dx.doi.org/10.1016/j.asoc.2018.02.049.

[44] H. Wang, Z. Wu, S. Rahnamayan, Y. Liu, M. Ventresca, Enhancing particle swarm optimization using generalized opposition-based learning, Inform. Sci. 181 (2011) 4699–4714, http://dx.doi.org/10.1016/j.ins.2011.03.016.

[45] K. Zhang, Q. Huang, Y. Zhang, Enhancing comprehensive learning particle swarm optimization with local optima topology, Inform. Sci. 471 (2019) 1–18, http://dx.doi.org/10.1016/j.ins.2018.08.049.

[46] J. Derrac, S. García, D. Molina, F. Herrera, A practical tutorial on the use of nonparametric statistical tests as a methodology for comparing evolutionary and swarm intelligence algorithms, Swarm Evol. Comput. 1 (2011) 3–18, http://dx.doi.org/10.1016/j.swevo.2011.02.002.

[47] M. Mafarja, I. Aljarah, A.A. Heidari, H. Faris, P. Fournier-Viger, X. Li, S. Mirjalili, Binary dragonfly optimization for feature selection using time-varying transfer functions, Knowl.-Based Syst. 161 (2018) 185–204, http://dx.doi.org/10.1016/j.knosys.2018.08.003.

[48] G. Sun, P. Ma, J. Ren, A. Zhang, X. Jia, A stability constrained adaptive alpha for gravitational search algorithm, Knowl.-Based Syst. 139 (2018) 200–213, http://dx.doi.org/10.1016/j.knosys.2017.10.018.

[49] M.-G. Martínez-Peñaloza, E. Mezura-Montes, Immune generalized differential evolution for dynamic multi-objective environments: An empirical study, Knowl.-Based Syst. 142 (2018) 192–219, http://dx.doi.org/10.1016/j.knosys.2017.11.037.

[50] J. Yi, L. Gao, X. Li, C.A. Shoemaker, C. Lu, An on-line variable-fidelity surrogate-assisted harmony search algorithm with multi-level screening strategy for expensive engineering design optimization, Knowl.-Based Syst. 170 (2019) 1–19, http://dx.doi.org/10.1016/j.knosys.2019.01.004.

[51] H. Wang, Z. Wu, S. Rahnamayan, Enhanced opposition-based differential evolution for solving high-dimensional continuous optimization problems, Soft Comput. 15 (2011) 2127–2140, http://dx.doi.org/10.1007/s00500-010-0642-7.

[52] D.H. Wolpert, W.G. Macready, No free lunch theorems for optimization, IEEE Trans. Evol. Comput. 1 (1997) 67–82, http://dx.doi.org/10.1109/4235.585893.

[53] C.A. Coello Coello, Use of a self-adaptive penalty approach for engineering optimization problems, Comput. Ind. 41 (2000) 113–127, http://dx.doi.org/10.1016/S0166-3615(99)00046-9.

[54] C.A. Coello Coello, E. Mezura Montes, Constraint-handling in genetic algorithms through the use of dominance-based tournament selection, Adv. Eng. Inf. 16 (2002) 193–203, http://dx.doi.org/10.1016/S1474-0346(02)00011-3.

[55] C.A.C. Coello, R.L. Becerra, Efficient evolutionary optimization through the use of a cultural algorithm, Eng. Optim. 36 (2004) 219–236.

[56] R.A. Krohling, L. dos S. Coelho, Coevolutionary particle swarm optimization using Gaussian distribution for solving constrained optimization problems, IEEE Trans. Syst. Man Cybern. B 36 (2006) 1407–1416, http://dx.doi.org/10.1109/TSMCB.2006.873185.

[57] A. Amirjanov, The development of a changing range genetic algorithm, Comput. Methods Appl. Mech. Engrg. 195 (2006) 2495–2508, http://dx.doi.org/10.1016/j.cma.2005.05.014.
[58] H. Liu, Z. Cai, Y. Wang, Hybridizing particle swarm optimization with differential evolution for constrained numerical and engineering optimization, Appl. Soft Comput. 10 (2010) 629–640, http://dx.doi.org/10.1016/j.asoc.2009.08.031.
[59] E. Zahara, Y.-T. Kao, Hybrid nelder–mead simplex search and particle swarm optimization for constrained engineering design problems, Expert Syst. Appl. 36 (2009) 3880–3886, http://dx.doi.org/10.1016/j.eswa.2008.02.039.
[60] C.A.C. Coello, Constraint-handling USing an evolutionary multiobjective optimization technique, Civ. Eng. Syst. 17 (2000) 319–346.
[61] T. Ray, K.M. Liew, Society and civilization: An optimization algorithm based on the simulation of social behavior, IEEE Trans. Evol. Comput. 7 (2003) 386–396, http://dx.doi.org/10.1109/TEVC.2003.814902.
[62] J. Lampinen, A constraint handling approach for the differential evolution algorithm, in: Proc. 2002 Congr. Evol. Comput. CEC02 Cat No02TH8600, Vol. 2, 2002, pp. 1468–1473, http://dx.doi.org/10.1109/CEC.2002.1004459.
[63] L. Wang, L. Li, An effective differential evolution with level comparison for constrained engineering design, Struct. Multidiscip. Optim. 41 (2010) 947–963, http://dx.doi.org/10.1007/s00158-009-0454-5.
[64] M. Zhang, W. Luo, X. Wang, Differential evolution with dynamic stochastic selection for constrained optimization, Nat. Inspired Probl.-Solv. 178 (2008) 3043–3074, http://dx.doi.org/10.1016/j.ins.2008.02.014.
[65] Y. Wang, Z. Cai, Y. Zhou, Z. Fan, Constrained optimization based on hybrid evolutionary algorithm and adaptive constraint-handling technique, Struct. Multidiscip. Optim. 37 (2009) 395–413, http://dx.doi.org/10.1007/s00158-008-0238-3.
[66] A. Baykasoğlu, F.B. Ozsoydan, Adaptive firefly algorithm with chaos for mechanical design optimization problems, Appl. Soft Comput. 36 (2015) 152–164, http://dx.doi.org/10.1016/j.asoc.2015.06.056.
[67] A. Sadollah, A. Bahreininejad, H. Eskandar, M. Hamdi, Mine blast algorithm: A new population based algorithm for solving constrained engineering optimization problems, Appl. Soft Comput. 13 (2013) 2592–2612, http://dx.doi.org/10.1016/j.asoc.2012.11.026.
[68] A.H. Gandomi, X.-S. Yang, A.H. Alavi, S. Talatahari, Bat algorithm for constrained optimization tasks, Neural Comput. Appl. 22 (2013) 1239–1255, http://dx.doi.org/10.1007/s00521-012-1028-9.