Contents lists available at ScienceDirect

Knowledge-Based Systems

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

Chaotic neural network algorithm with competitive learning for global optimization

Yiying Zhang

School of Electrical and Information Engineering, Jiangsu University, Zhenjiang 212013, Jiangsu, China

A R T I C L E   I N F O

Article history:
Received 9 April 2021
Received in revised form 30 June 2021
Accepted 14 August 2021
Available online 17 August 2021

Keywords:
Artificial neural networks
Neural network algorithm
Global optimization
Chaos theory

A B S T R A C T

Neural network algorithm (NNA) is one of the newest proposed metaheuristic algorithms. NNA has strong global search ability due to the unique structure of artificial neural networks. Further, NNA is an algorithm without any effort for fine tuning initial parameters. Thus, it is very easy for NNA to solve different types of optimization problems. However, when used for solving complex optimization problems, slow convergence and premature convergence are its drawbacks. To overcome the two drawbacks, this work presents an improved NNA, namely chaotic neural network algorithm with competitive learning (CCLNNA), for global optimization. In CCLNNA, population is first divided into excellent subpopulation and common subpopulation according to the built competitive mechanism. Then, to balance exploration and exploitation of CCLNNA, excellent subpopulation is optimized by the designed transfer operator while common subpopulation is updated by the combination of the designed bias operator and transfer operator. Besides, chaos theory is introduced to increase the chance of CCLNNA to escape from the local optimum. To investigate the effectiveness of the improved strategies, CCLNNA is first used to solve the well-known CEC 2014 test suite with 30 benchmark functions. Then it is employed for solving three constrained real-world engineering design problems. Experimental results reveal that the improved strategies introduced to NNA can significantly improve the optimization performance of NNA and CCLNNA is a very powerful algorithm in solving complex optimization problems with multimodal properties by comparing with the other competitive algorithms.

1. Introduction

At present, a lot of engineering fields refer to optimization problems, such as small scale load predication problem [1], parameter identification of photovoltaic models [2], job-shop scheduling problem [3], robot path planning [4], wind power predication [5], energy management problem [6], color image segmentation [7], the joint energy-reserve market cleaning problem [8], complex network clustering [9], engineering structure design problem [10], short term power forecast of wind and solar power [11], intelligent skin cancer detection problem [12], energy efficient routing problems for wireless sensor networks [13] and the extraction of appropriate nodal marginal prices for committed reserve [14]. Thus it is always a hot research topic that developing optimization methods solves different types of practical optimization problems. Through unremitting efforts, a lot of optimization methods have been developed by researchers, which can be broadly divided into deterministic methods and metaheuristic methods.

Deterministic methods usually are based on numerical linear and nonlinear programming methods that require substantial gradient information and seek to improve the solution in the neighborhood of a starting point [15,16]. Note that the performance of deterministic algorithms has become highly dependent on the selected starting points. For deterministic algorithms, once the starting points are fixed, the obtained solutions have been determined. In fact, most real-world optimization problems are very complex, whose objective functions usually refer to multiple local optimal solutions and have nonlinear characteristics [17]. Given the sensitivity for the starting points, deterministic algorithms tend to trap into local optimum for solving complex real-world problems [16,18,19]. Compared with deterministic algorithms, metaheuristic algorithms do not rely on gradient information and complex math principles, which are based on the defined simple rules to imitate natural phenomena, such as differential evolution [20] inspired by biological evolution, cuckoo search [21] inspired by the obligate brood parasitic behavior of some cuckoo species, queuing search [22] inspired by human activities in queuing process, and gravitational search algorithm [23] inspired by the law of gravity. Note that, randomness can be regarded as the most remarkable feature for metaheuristic methods, which is mainly reflected in the following aspects:

E-mail addresses: zhangyiying@tju.edu.cn, zhangyiying@ujs.edu.cn.

Nomenclature

$\boldsymbol{X}^t$	Population at time t
$\boldsymbol{W}^t$	Weight matrix at time t
$\boldsymbol{x}^t_{\text{new},j}$	The jth weighted solution at time t
$\boldsymbol{w}^t_{\text{obj}}$	Objective weight at time t
$\boldsymbol{x}^t_{\text{obj}}$	The best solution at time t
β^t	Modification factor at time t
$\boldsymbol{X}^t_{\text{C}}$	Common population at time t
$\boldsymbol{X}^t_{\text{E}}$	Excellent population at time t
$\boldsymbol{M}^t$	The mean solution at time t
η^t	Penalty factor at time t
$n_{\max}$	The length of logistic map
α	Impact factor of logistic map
x_n	The ith element of logistic map
$t_{\max}$	The maximum number of iterations
n_{run}	The number of independent runs
N_{w}	An integer between 0 and N
N_{p}	An integer between 0 and D
$\boldsymbol{l}$	The lower limit of variables
$\boldsymbol{u}$	The upper limit of variables
N	Population size
$\lambda_1, \lambda_2, \ldots, \lambda_6$	Random number between 0 and 1 subject to uniform distribution
MEAN	Mean value
STD	Standard deviance
NNA	Neural network algorithm
MVO	Multi-verse optimizer
SOA	Seagull optimization algorithm
STOA	Sooty tern optimization algorithm
COA	Chimp optimization algorithm
MFO	Moth-flame optimization
SSA	Salp swarm algorithm
TSA	Tunicate swarm algorithm
CCLNNA	Chaotic neural network algorithm with competitive learning

- Initialization population. Metaheuristic methods are population-based optimization techniques. Each individual in the population denotes a potential solution. When one metaheuristic method is employed to perform the optimization task, its population needs to be initialized. Generally, each individual in the population is initialized by randomly selecting a solution from the given search space.
- Random numbers. Random numbers are often introduced to the defined rules of updating the population. The introduced random numbers often obey some typical distributions, such as uniform distribution, normal distribution and Lévy distribution.

Given the advantages of randomness, the sensitivity of metaheuristic methods for the initial solutions is significantly lower than that of deterministic methods, which means metaheuristic methods have stronger stable than deterministic methods. In addition, metaheuristic methods have achieved promising solutions on many real-world engineering optimization problems, which prove the excellent performance of metaheuristic methods for solving complex optimization problems. However, there remains a need for developing efficient metaheuristic methods with the essential parameters due to the following two reasons:

- There is a very important theory motivating the development of metaheuristic methods, which is called No Free Lunch (NFL) theorem [24]. According to NFL theorem, one metaheuristic method may obtain very promising results on a set of optimization problems while it may show poor performance on another set of optimization problems [25–27]. In other words, no single metaheuristic method is suitable for solving all types of optimization problems. Thus, it is essential for researchers to develop more metaheuristic methods to offer better solutions for different types of real-world optimization problems.
- Most reported metaheuristic methods need more parameters. The parameters of metaheuristic methods can be divided into two groups, i.e. common parameters and special parameters [28]. Common parameters are required for all metaheuristic methods, which are population size and terminal criterion (e.g. the defined threshold value, the maximum number of function evaluations, or the maximum number of iterations). Special parameters reflect the features of the methods, such as mutation rate and crossover rate in differential evolution to simulate the two key processes of biological evolution [29] and discovery rate in cuckoo search to imitate the phenomenon of an alien egg discovered by a host [21]. Here, it should be pointed out that the major drawback of metaheuristic methods with special parameters is that it is a very hard task to set the values of special parameters for the unknown optimization problems to get the optimal solutions [28]. Given this drawback, the applications of metaheuristic methods with special parameters will be restricted. Thus, in order to address this drawback, developing metaheuristic methods without special parameters is an efficient way.

Motivated by the above two reasons, this paper presents a novel optimization method, called chaotic neural network algorithm with competitive learning (CCLNNA), for global optimization problems. CCLNNA is a new variant of neural network algorithm (NNA) [30]. NNA is a very interesting metaheuristic algorithm, which is inspired by artificial neural networks and biological nervous systems. Artificial neural networks are mostly used for prediction purposes. However, NNA is designed by smartly combining artificial neural networks and randomness for solving optimization problems. Given the unique structure of artificial neural networks, NNA shows strong global search ability. In addition, NNA does not need other parameters except for population size and stopping criterion, which can distinguish the NNA over most existing metaheuristic methods [30]. However, as the experimental results shown in the original reference of NNA [30], NNA has slow convergence rate and may be trapped into local optimum for complex multimodal optimization problems, which will restrict its further applications in some real-world optimization problems with limitations like computation resource constraints. CCLNNA is proposed to enhance global search ability and improve convergence performance of NNA. The main contributions of this work can be stated as follows:

- A novel optimization method, namely CCLNNA, is reported for global optimization.
- In order to keep population diversity of CCLNNA, the population is first divided into two subpopulations, i.e. excellent subpopulation and common subpopulation, based on the built competitive mechanism. Then, the two subpopulations are optimized by different learning strategies. Transfer operator is performed for excellent subpopulation to enhance the local search ability of CCLNNA. The hybrid operator based on transfer operator and bias operator is performed for common subpopulation to enhance the global search ability of CCLNNA.

- In order to increase the chance of CCLNNA to escape from the local optimal solutions, a classical chaotic map, namely logistic map, is introduced to adjust the running probability of the different learning strategies.
- The optimization performance of CCLNNA is investigated by 30 numerical optimization problems from CEC 2014 test suit and three real-world constrained engineering problems.

The rest of this paper is organized as follows: In Section 2, NNA is briefly introduced. The proposed CCLNNA is described in Section 3. CCLNNA is used for solving CEC 2014 test suite in Section 4. The applications of CCLNNA in practical engineering design problems are presented in Section 5. Finally, conclusion and further work are drawn in Section 6.

2. Neural network algorithm

Artificial neural networks (ANNs) are computational models inspired by the structure and/or functional aspects of biological neural networks [31]. ANNs are used for prediction purposes in most cases, which try to close the gap among the predicted solution and the given target solution by frequently changing the weight values. ANNs can generally be divided into two categories: feed forward neural networks and recurrent neural networks. As can be seen from Fig. 1(a), the architecture of feed forward neural networks has no loops. Thus such neural networks produce only one set of output values from the given data set. Unlike feed forward neural networks, recurrent neural networks have loop architecture, which can produce a sequence of values from the given data set as shown in Fig. 1(b). ANNs usually need a given target solution. However, the goal of NNA is to find the target solution from the given search space. That is, the target solution in ANNs is known while the target solution is unknown in NNA. In order to address this issue, the authors of NNA first see the current optimal solution as the target solution. Then the better target solution can be found by adjusting the weight values of each neural cell as shown in Fig. 2. NNA is also a population-based optimization algorithm, which has the following four key components:

- Update population

According to the authors of NNA, population $\boldsymbol{X}^t = \left\{\boldsymbol{x}_1^t, \boldsymbol{x}_2^t, \ldots, \boldsymbol{x}_N^t\right\}$ is updated by weight matrix $\boldsymbol{W}^t = \left\{\boldsymbol{w}_1^t, \boldsymbol{w}_2^t, \ldots, \boldsymbol{w}_N^t\right\}$, where $\boldsymbol{w}_i^t = \left[w_{i,1}^t, w_{i,2}^t, \ldots, w_{i,N}^t\right]$ is the weight vector of the ith individual and $\boldsymbol{x}_i^t = \left[x_{i,1}^t, x_{i,2}^t, \ldots, x_{i,D}^t\right]$ is the position of the ith individual. In addition, D is the number of variables. Further, the formula of generating new population can be expressed by

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

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

where N is population size, t is the current number of iterations, $\boldsymbol{x}_i^t$ is solution of the ith individual at time t, and $\boldsymbol{x}_{\text{new},j}^t$ is the weighted solution of the jth individual at time t. In addition, the weight vector $\boldsymbol{w}_i^t$ should meet the following formulation:

$$\sum_{j=1}^{N} w_{i,j}^t = 1,\ 0 < w_{i,j}^t < 1, i = 1, 2, \ldots, N, j = 1, 2, \ldots, N \tag{3}$$

- Update weight matrix

As shown in Eq. (1), weight matrix $\boldsymbol{W}^t$ plays a key role in NNA in generating new population. The weight matrix $\boldsymbol{W}^t$ can be updated by

$$\boldsymbol{w}_i^t = \left|\boldsymbol{w}_i^t + 2 \times \lambda_1 \times \left(\boldsymbol{w}_{\text{obj}}^t - \boldsymbol{w}_i^t\right)\right|, i = 1, 2, \ldots, N \tag{4}$$

where λ_1 is a random number between 0 and 1 subject to uniform distribution and $\boldsymbol{w}_{\text{obj}}^t$ is the target weight vector. Note that, the target weight vector $\boldsymbol{w}_{\text{obj}}^t$ and the target solution $\boldsymbol{x}_{\text{obj}}^t$ share the same index. More specifically, if $\boldsymbol{x}_{\text{obj}}^t$ is equal to $\boldsymbol{x}_v^t$ $(v \in [1, N])$at time t, $\boldsymbol{w}_{\text{obj}}^t$ is equal to $\boldsymbol{w}_v^t$.

- Bias operator

Bias operator is to enhance the global search ability of NNA. Modification factor β is employed to determine the bias proportion, which can be updated by

$$\beta^{t+1} = 0.99\beta^t \tag{5}$$

Bias operator consists of bias population and bias weight matrix, which can be described as follows: bias population operator includes two variables: a random number N_{p} and a set P. Let $\boldsymbol{l} = (l_1, l_2, \ldots, l_D)$ and $\boldsymbol{u} = (u_1, u_2, \ldots, u_D)$ be the lower and the upper limits of variables.N_{p} equals $\lceil \beta^t \times D \rceil$. P is the set of N_{p} integers between 0 and D selected randomly. Therefore, the bias population can be defined as

$$x_{i,P(s)}^t = l_{P(s)} + \left(u_{P(s)} - l_{P(s)}\right) \times \lambda_2, s = 1, 2, \ldots, N_{\text{p}} \tag{6}$$

where λ_2 is a random number between 0 and 1 subject to uniform distribution. Bias matrix also refers to two variables: a random number N_{w} and a set Q.N_{w} is equal to $\lceil \beta^t \times N \rceil$. Q is the set of N_{w} integers between 0 and N selected randomly. Thus the bias weight matrix can be defined as

$$w_{i,Q(r)}^t = \lambda_3, r = 1, 2, \ldots, N_{\text{w}} \tag{7}$$

where λ_3 is a random number between 0 and 1 subject to uniform distribution.

- Transfer operator

Transfer operator is to generate a better solution toward the current best solution, which focuses on the local search ability of NNA. This operator can be represented as

$$\boldsymbol{x}_i^{t+1} = \boldsymbol{x}_i^t + 2 \times \lambda_4 \times \left(\boldsymbol{x}_{\text{obj}}^t - \boldsymbol{x}_i^t\right), i = 1, 2, \ldots, N \tag{8}$$

where λ_4 is a random number between 0 and 1 subject to uniform distribution.

In addition, like other population-based metaheuristic methods, NNA is initialized by

$$x_{i,j}^t = l_j + \left(u_j - l_j\right) \times \lambda_5, i = 1, 2, \ldots, N, j = 1, 2, \ldots, D \tag{9}$$

where λ_5 is a random number between 0 and 1 subject to uniform distribution.

Based on the mentioned four key steps, the implementation of NNA can be found in Fig. 3. From Fig. 3, NNA has a very simple structure and is easy to work.

3. Chaotic neural network algorithm with competitive learning

In this section, the proposed CCLNNA is introduced, which includes two subsections. Firstly, the motivation of CCLNNA is presented. Then, the framework and implementation of CCLNNA are described.

3.1. Motivation of CCLNNA

The motivation of CCLNNA is from two drawbacks of NNA for solving complex optimization problems, i.e. slow convergence rate and premature convergence, which can be introduced as follows.

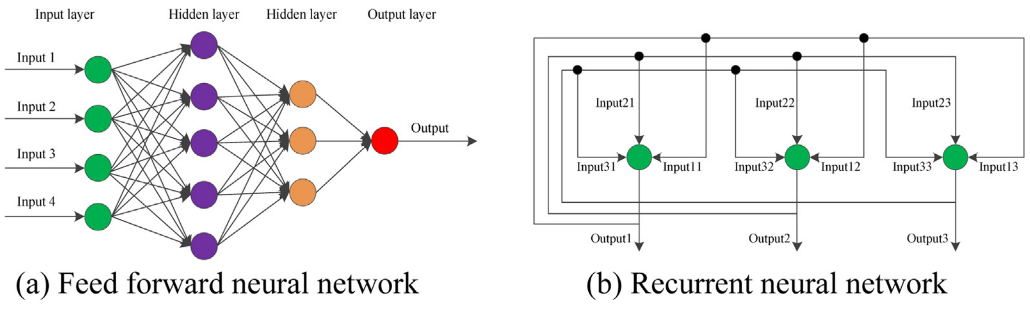

Fig. 1. Schematic view of neural networks.

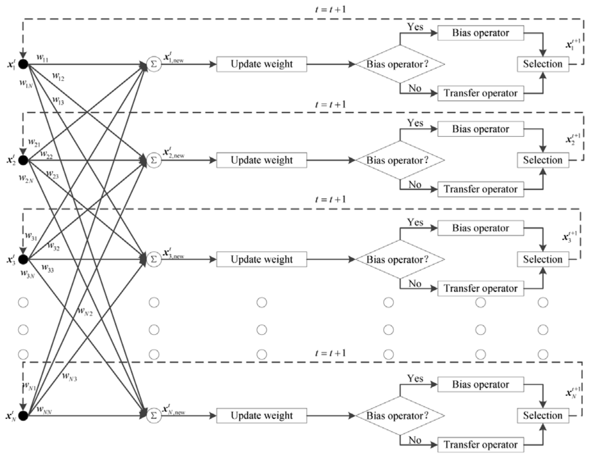

Fig. 2. Schematic view of NNA.

- Slow convergence rate. An individual has a better fitness value, which means there is more helpful information around this individual to find better solutions. Thus, performing local search for this individual can increase the chance of finding global optimal solution. Similarly, an individual has a worse fitness value, which indicates there is less helpful information around this individual to obtain better solutions. Thus, performing global search for this individual is a good choice to achieve better solutions. However, in NNA, local search or global search performed for an individual is based on modification factor, which does not fully consider the differences among individuals. That is, this designed mechanism for updating individuals will lead to slow convergence rate of NNA to some extent.
- Premature convergence. Premature convergence is used for describing a phenomenon that one algorithm converges to a local optimum in shorter periods of time. In NNA, the current optimal solution plays a very important role, which is used to guide the optimization process as shown in Eq. (8). Note that Eq. (8) has more chance to be performed with the increasing of the iteration times. Obviously, if the current optimal solution is a local minimum, the whole population may be trapped into local optimum, which will lead to premature convergence.

3.2. The framework and implementation of CCLNNA

To overcome the slow convergence rate of NNA, in CCLNNA, population is divided into two subpopulations, i.e. excellent subpopulation and common subpopulation, based on the built competitive mechanism. Two different learning strategies are designed for the two subpopulations. In addition, to avoid premature convergence of NNA, in CCLNNA, mean position of the current population and chaotic map are introduced to the designed learning strategies. The framework consists of the following four phases.

(1) Competitive mechanism. As done in [32], N individuals in the population $\boldsymbol{X}^t$ are first randomly allocated into $N/2$ couples. Then, for the ith couple, one individual with better fitness value is called excellent individual $\boldsymbol{x}^t_{\mathrm{E},i}$ and another individual is called common individual $\boldsymbol{x}^t_{\mathrm{C},i}$. Lastly, all common individuals form common subpopulation $\boldsymbol{X}^t_{\mathrm{C}} = \left\{\boldsymbol{x}^t_{\mathrm{C},1}, \boldsymbol{x}^t_{\mathrm{C},2}, \ldots, \boldsymbol{x}^t_{\mathrm{C},N/2}\right\}$ and all excellent individuals make excellent subpopulation $\boldsymbol{X}^t_{\mathrm{E}} = \left\{\boldsymbol{x}^t_{\mathrm{E},1}, \boldsymbol{x}^t_{\mathrm{E},2}, \ldots, \boldsymbol{x}^t_{\mathrm{E},N/2}\right\}$.

(2) Learning strategy for excellent subpopulation. Based on the built competitive mechanism, excellent subpopulation has more potential to find better solutions compared with common subpopulation. Therefore, local search is performed for excellent subpopulation in the designed learning strategy, which can be expressed as

$$\boldsymbol{x}^{t+1}_{\mathrm{E},i} = \boldsymbol{x}^t_{\mathrm{E},i} + \underbrace{2 \times \kappa_1 \times \left(\boldsymbol{x}^t_{\mathrm{obj}} - \boldsymbol{x}^t_{\mathrm{E},i}\right)}_{\text{Guidance term}} + \underbrace{2 \times \eta^t \times \kappa_2 \times \left(\boldsymbol{M}^t - \boldsymbol{x}^t_{\mathrm{E},i}\right)}_{\text{Penalty term}} \tag{10}$$

where κ_1 and κ_2 are two random numbers between 0 and 1 subject to uniform distribution, $\boldsymbol{M}^t$ is the mean position of the population $\boldsymbol{X}^t$ and η^t is a penalty factor. In addition, $\boldsymbol{M}^t$ can be computed by

$$\boldsymbol{M}^t = \frac{1}{N}\sum_{i=1}^{N}\boldsymbol{x}^t_i \tag{11}$$

Penalty term is the only difference between Eqs. (8) and (10). In this work, penalty term is introduced to prevent NNA from premature convergence, whose motivations can be stated as follows: (1)$\boldsymbol{M}^t$ is an efficient indicator measuring the distribution of the population, which has been used to update population in some reported studies [28,32,33]. Unlike $\boldsymbol{x}^t_{\mathrm{obj}}$, $\boldsymbol{M}^t$ is changing in the whole search process. Thus, as shown in Eq. (10), one individual is guided by $\boldsymbol{x}^t_{\mathrm{obj}}$ and $\boldsymbol{M}^t$, which can increase the chance of the whole population to escape from the local optimum; (2) one promising algorithm needs to consider both quality and efficiency. That is, this algorithm should converge to best solution with the fastest speed as soon as possible. Given this, the penalty

```
NNA Algorithm
01: Begin
02:   Initialize the weight matrix $\boldsymbol{W}^t$ and the population $\boldsymbol{X}^t$ by Eq. (3) and Eq. (9), respectively
03:   Calculate the fitness value of each solution and then set the optimal solution and the optimal weight
04:   Repeat
05:     For each individual $i \in N$ do
06:       Generate the new solution $\boldsymbol{x}_i^t$ by Eq. (1) and Eq. (2) and the new weight matrix $\boldsymbol{w}_i^t$ by Eq.(4)
07:       If rand $\le \beta^t$
08:         Perform the bias operator for $\boldsymbol{x}_i^{t+1}$ by Eq. (6) and the weight $\boldsymbol{w}_i^t$ by Eq. (7)
09:       Else
10:         Perform the transfer function operator for $\boldsymbol{x}_i^t$ by Eq.(8)
11:       End if
12:     End for
13:   Generate the new modification factor $\beta^{t+1}$ via Eq. (5)
14:   Calculate the fitness value of each solution and find the optimal solution and the optimal weight
15:   Until (stop condition=false)
16: Post process results and visualization
17: End
```

Fig. 3. The implementation of NNA.

factor η^t is introduced, which is equal to β^t. According to Eq. (5), the value of η^t becomes small gradually, which reduces gradually the contribution of penalty term. This is helpful for accelerating the convergence speed of CCLNNA.

As can be seen from Eq. (10), the learning strategy for excellent subpopulation consists of guidance term and penalty term. Guidance term means an individual is guided by another individual with better fitness value. Generally speaking, an individual with better fitness value has a greater probability to find the global optimal solution compared with an individual with worse fitness value. Thus, guidance term is a very effective learning strategy to improve the quality of the whole population as shown in Eq. (8). It should be pointed out that when the leader of the guidance term is trapped into a local optimal solution, the whole population may find the local optimal solution not the global optimal solution. To avoid this drawback, penalty term is introduced to Eq. (10). The mean position of the whole population will change almost every loop. Thus, the penalty term in Eq. (10) can be regarded as dynamic. Obviously, even though the leader in the guidance term gets trapped into a local optimal solution, it still has a high probability to escape from the local optimal solution due to the impact of penalty term. In other words, Eq. (10) in CCLNNA shows the stronger ability of preventing premature convergence compared with Eq. (8) in NNA.

(3) Learning strategy for common subpopulation. Considering the built competitive mechanism, common subpopulation has more chance to find better solutions by performing global search. In addition, in order to improve the convergence rate of CCLNNA, two candidate search operators are performed for the individuals of common population. The first candidate search operator is bias population in NNA. Note that, unlike Eq. (6) in NNA, in order to accelerate convergence of CCLNNA, a new bias operator is designed, which can be written as

$$x_{\mathrm{C},i,P(s)}^t = \begin{cases} \gamma \times \underbrace{\left(l_{P(s)} + \left(u_{P(s)} - l_{P(s)}\right) \times \lambda_2\right)}_{\text{Basic term}} + (1-\gamma) \times \underbrace{x_{\mathrm{E},i,P(s)}^t}_{\text{Adjustment term}}, \text{if } \lambda_5 \le \lambda_6 \\ \gamma \times \underbrace{\left(l_{P(s)} + \left(u_{P(s)} - l_{P(s)}\right) \times \lambda_2\right)}_{\text{Basic term}} + (1-\gamma) \times \underbrace{x_{\mathrm{obj},P(s)}^t}_{\text{Adjustment term}}, \text{otherwise} \end{cases},$$

$$i = 1, 2, \ldots, N/2, s = 1, 2, \ldots, N_{\mathrm{p}} \tag{12}$$

$$\boldsymbol{x}_{\mathrm{C},i}^{t+1} = \boldsymbol{x}_{\mathrm{C},i}^t \tag{13}$$

where λ_5, λ_6 and γ are three random numbers between 0 and 1 subject to uniform distribution. From the right-hand side of Eq. (12), the designed bias operator consists of basic term and adjustment term. Basic term is from Eq. (6). The adjustment term is introduced based on the following consideration. Current optimal solution $\boldsymbol{x}_{\mathrm{obj}}^t$ and excellent subpopulation $\boldsymbol{X}_{\mathrm{E}}^t$ include more helpful information to find global optimal solution compared with common subpopulation. So, adjustment term with current optimal solution $\boldsymbol{x}_{\mathrm{obj}}^t$ and excellent subpopulation $\boldsymbol{X}_{\mathrm{E}}^t$ is very helpful for common subpopulation to find better solutions. In addition, γ is called adjustment factor, which used to balance basic term and adjustment term. As done in the designed learning strategy for excellent subpopulation, the second candidate search operator for common population can be represented by

$$\boldsymbol{x}_{\mathrm{C},i}^{t+1} = \boldsymbol{x}_{\mathrm{C},i}^t + \underbrace{2 \times \kappa_1 \times \left(\boldsymbol{x}_{\mathrm{obj}}^t - \boldsymbol{x}_{\mathrm{C},i}^t\right)}_{\text{Guidance term}} + \underbrace{2 \times \eta^t \times \kappa_2 \times \left(\boldsymbol{M}^t - \boldsymbol{x}_{\mathrm{C},i}^t\right)}_{\text{Penalty term}} \tag{14}$$

In addition, a selection mechanism based on chaotic map is built. Chaotic map has been widely used for improving the performance of metaheuristic methods, such as chaotic dynamic weight particle swarm optimization [34], chaotic whale optimization algorithm [35], enhance chaotic gravitational search algorithm [36], chaos-enhanced cuckoo search optimization [37], chaos-based gray wolf optimizer [38] and chaos-based firefly algorithm [39]. In this work, the selected chaotic map is the well-known logistic map [37], which can be defined by

$$x_{n+1} = \alpha x_n(1 - x_n), n = 1, 2, 3, \ldots, n_{\max} \tag{15}$$

where α is called impact factor and $n_{\max}$ is the length of logistic map. In addition, α is set to 4 in this work as done in the other reported references [40–42]. In general, x_1 is initialized by a random number between 0 and 1. Fig. 4 shows the schematic view of logistic map, where $n_{\max}$ is equal to 10,000. From Fig. 4, the values generated by logistic map range are from 0 and 1. In this work, $n_{\max}$ is equal to the maximum number of iterations $t_{\max}$. The built selection mechanism based on logistic map can be described as follows. Firstly, a random number ρ between 0 and 1 subject to uniform distribution is generated at time t. Then if ρ is more than x_t, Eq. (14) is performed; otherwise, Eqs. (12)–(13) are performed.

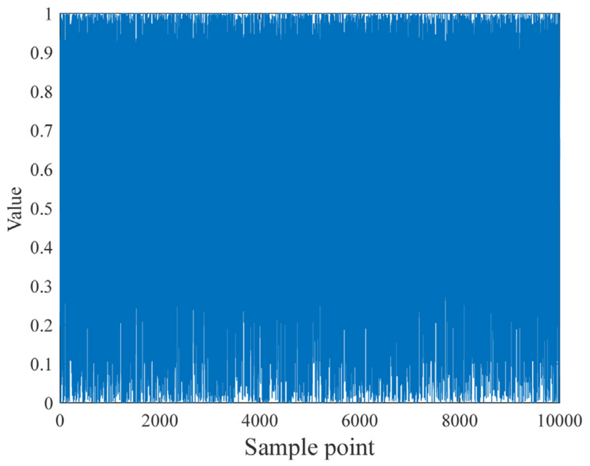

Fig. 4. Schematic view of logistic map.

(4) Bias weight matrix based on chaotic map. In CCLNNA, chaotic map is employed to determine that the bias weight matrix is performed or not. The way of using chaotic map is very similar with that of learning strategy for common subpopulation, which can be stated as follows. Firstly, a random number ρ between 0 and 1 subject to uniform distribution is generated at time t. Then if ρ is less than x_t, Eq. (7) is performed; otherwise, the bias weight matrix is not performed.

Based on the presented framework of CCLNNA, the detailed implementation of CCLNNA can be found in Fig. 5.

4. CCLNNA for numerical optimization

This section is to check the performance of the proposed CCLNNA for numerical optimization problems. This section is divided into the following three subsections. Benchmark functions are presented in Section 4.1. Section 4.2 is to check the effectiveness of the improved strategies by comparing NNA and CCLNNA. The performance comparisons between CCLNNA and several recently proposed metaheuristic methods are made in Section 4.3.

4.1. Benchmark functions

CEC 2014 test suite [43] has been widely employed by many reported metaheuristic methods for checking their performance for complex optimization problems [44–46]. CEC 2014 test suite includes 30 test functions as listed in Table 1, which consists of four types of test functions and can be stated as follows:

- Three unimodal functions, i.e. F1–F3. Unimodal functions are used to evaluate the capability of the proposed CCLNNA in exploiting the search regions of promising search areas.
- 13 simple multimodal functions, i.e. F4–F16. Multimodal functions are good option to check the ability of CCLNNA in exploring the search regions of promising search areas.
- Six hybrid functions, i.e. F17–F22. Hybrid functions are employed to check the ability of the proposed CCLNNA in exploring along with the strength of maintain a suitable balance between exploitation and exploration [46].
- Eight composition functions, i.e. F23–F30. Composition functions have the same function with hybrid functions.

In addition, in order to verify the competitiveness of the proposed CCLNNA, the solutions obtained by CCLNNA are compared with those of eight recently reported methods including NNA, multi-verse optimizer (MVO) [47], seagull optimization algorithm (SOA) [48], chimp optimization algorithm (COA) [49], sooty tern optimization algorithm (STOA) [50], tunicate swarm algorithm (TSA) [51], moth-flame optimization(MFO) [52] and salp swarm algorithm (SSA) [26]. For a fair comparison, all the considered algorithms have the same population size and the number of function evaluations, which was set to 50 and 5,000 multiples by dimension size, respectively. In addition, the other parameters required by the considered algorithms were extracted directly from the original references. The parameters of the applied algorithms can be found in Table 2. Besides, each algorithm was executed 50 independent runs for each test function and mean value (MEAN) and standard deviance (STD) of the obtained results were recorded. MEAN and STD can be defined by

$$\mathrm{MEAN} = \frac{1}{n_{\mathrm{run}}}\sum_{i=1}^{n_{\mathrm{run}}} f(\boldsymbol{x}_i^*) \tag{16}$$

$$\mathrm{STD} = \sqrt{\frac{1}{n_{\mathrm{run}}-1}\sum_{i=1}^{n_{\mathrm{run}}}\left(f(\boldsymbol{x}_i^*)-\mathrm{MEAN}\right)^2} \tag{17}$$

where $\boldsymbol{x}_i^*$ is the best solution achieved by one algorithm on one test function at the ith run and n_{run} is the number of independent runs. Experimental results obtained by the applied algorithms on CEC 2014 test suite have been shown in Tables 3 and 4. In Tables 3–4, the best results were in highlighted in bold type.

4.2. Case 1: The impact of the improved strategies on NNA

In this work, CCLNNA is proposed to enhance the global search ability of NNA by introducing some improved strategies. Thus, in order to check the effectiveness of the improved strategies, this section focuses on comparing optimization performance between NNA and CCLNNA.

Experimental results achieved by NNA and CCLNNA on CEC 2014 test suite have been presented in Table 3. From Table 3, in terms of MEAN, CCLNNA outperforms NNA on 25 test functions including F1, F2, F3, F4, F5, F6, F8, F9, F10, F11, F13, F15, F16, F17, F18, F19, F20, F21, F22, F23, F24, F25, F28, F29 and F30; CCLNNA can find the same solutions with NNA on F7, F12 and F14; CLNNA is only inferior to NNA on F26 and F27. In terms of STD, NNA beats CCLNNA on F16, F23, F26 and F28 while CCLNNA is superior to NNA on the rest 26 functions. Obviously, CCLNNA shows remarkable advantages over NNA on both MEAN and STD, which means the proposed CCLNNA has more stronger global search ability and stable than NNA. That is, the introduced improved strategies are very effective.

In addition, in order to compare the convergence performance between NNA and CCLNNA, Fig. 6 presents eleven typical convergence curves on CEC 2014 test suite. From Fig. 6, CCLNNA can find better solutions with faster convergence speed than NNA on these curves. Note that, for convergence curves of F5, F6, F8, F9, F10, F11, F12, F13 and F16, there are the following two common characteristic: (1) in the initial stage, NNA has better convergence performance than CCLNNA on these curves. However, NNA tend to converge to local optimum with the increasing of function evaluations; (2) although CCLNNA does not show convergence advantages over NNA on these curves in the initial stage, the convergence advantages of CCLNNA are gradually revealed with the increasing of function evaluations. The two characteristics mean that the proposed CCLNNA has stronger ability of escaping from local optimum compared with NNA.

```
CCLNNA Algorithm
01: Begin
02:   Initialize the current number of iterations $t(t = 0)$, the maximum number of function evaluations $t_{\max}$ and
      modification factor $\beta^t$ ($\beta^t = 1$)  // Initialization parameters
03:   Initialize the weight matrix $W^t$ and the population $X^t$ by Eq. (3) and Eq. (9), respectively
04:   Calculate fitness values of individuals and then find the optimal solution $x^t_{\mathrm{obj}}$ and the optimal weight $w^t_{\mathrm{obj}}$
05:   Update the current number of iterations by $t$=$t$+1
06:   Generate chaotic sequence $\varphi$=[$\varphi_1,\varphi_2,\ldots,\varphi_{t_{\max}}$] by Eq. (15)   // Generate chaotic sequence
07:   While $t < t_{\max}$ // Main loop
08:     Calculate $M^t$ by Eq. (11) and $\eta^t$ by $\eta^t=\beta^t$  // Update mean position and penalty factor
09:     Generate $X^t_{\mathrm{E}}$ and $X^t_{\mathrm{C}}$ by the built competitive mechanism    // Competitive mechanism
10:     Update $X^t_{\mathrm{E}}$ and $X^t_{\mathrm{C}}$ by Eqs. (1-2) and generate $W^{t+1}$ by Eq. (4)
11:     For $i$=1: $0.5*N$ do     // Update common subpopulation
12:       Generate random number $\rho$ and $\gamma$
13:       If $\rho \le \varphi_t$
14:         Obtain $x^{t+1}_{\mathrm{C},i}$ by Eqs. (12-13)
15:       Else
16:         Obtain $x^{t+1}_{\mathrm{C},i}$ by Eq. (14)
17:       End if
18:     End for
19:     For $i$=1: $0.5*N$ do     // Update excellent subpopulation
20:       Obtain $x^{t+1}_{\mathrm{E},i}$ by Eq. (10)
21:     End for
22:     For $i$=1: $0.5*N$ do     // Update weight
23:       Generate a random number $\rho$
24:       If $\rho \le \varphi_t$
25:         Update $w^{t+1}_i$ by Eq. (7)
26:       End if
27:     End for
28:     Calculate fitness values of individuals and find the optimal solution $x^{t+1}_{\mathrm{obj}}$ and the optimal weight $w^{t+1}_{\mathrm{obj}}$
29:     Update modification factor $\beta^t$ by Eq. (5)
30:     Update the current number of iterations by $t$=$t$+1  // Update the number of iterations
31:   End while
32: End
```

Fig. 5. The implementation of CCLNNA.

4.3. Case 2: Comparisons between CCLNNA and recently proposed algorithms

This section is to compare the optimization performance between CCLNNA and seven recent proposed methods including SSA, MFO, MVO, SOA, CHOP, TSA, and STOA.

Table 4 displays the experimental results obtained by CCLNNA and seven compared algorithms on CEC 2014 test suite. As can be seen from Table 4, CCLNNA can offer better solutions than TSA on all test functions in terms of MEAN. SOA, CHOP and STOA only can achieve better MEAN than CCLNNA on two (i.e. F24 and F26), one (i.e. F24) and two (i.e. F24 and F26) test functions, respectively. However, CCLNNA can give better MEAN than SOA, CHOP and STOA on 28, 29 and 28 test functions, respectively. In addition, MFO only outperforms CCLNNA on F26 and F28 while MFO is inferior to CCLNNA on 26 test functions. Besides, MVO and SSA show strong competitiveness, which are superior to CCLNNA on eight (i.e. F3, F6, F15, F19, F20, F21, F22, F27 and F28) and six test functions (i.e. F15, F19, F20, F21, F22 and F26), respectively. Moreover, MVO and SSA can find the same MEAN with CCLNNA on two (i.e. F12 and F13) and two (i.e. F7 and F14) test functions, respectively. Note that, MVO and SSA cannot compete with CCLNNA on 20 and 22 test functions in terms MEAN, respectively.

From Table 4, in terms of STD, TSA only beats CCLNNA on F16 while TSA is inferior to CCLNNA on the rest 29 test functions. SOA and MFO can offer better STD than CCLNNA on F16, F26, F27 and F28 while CCLNNA outperforms SOA and MFO on the rest 26 test functions. In addition, CHOP, STOA, MVO and SSA have advantages over CCLNNA on seven (i.e. F6, F11, F16, F22, F24, F27 and F28), six (i.e. F16, F23, F24, F26, F27 and F28), ten (i.e. F3, F6, F15, F16, F19, F20, F21, F23, F27 and F28) and nine (i.e. F7, F15, F16, F19, F20, F22, F23, F26 and F27) test functions in terms of STD, respectively. Note that CCLNNA is superior to CHOP, STOA, MVO and SSA on the rest 23, 24, 20 and 21 test functions, respectively. Obviously, CCLNNA shows stronger stable than the compared algorithms.

In order to compare the overall differences among the applied algorithms, tied rank [53] is used for ranking for the applied algorithms on basis of MEAN and STD. Table 5 and Fig. 6 present the ranking results. From Table 5 and Fig. 7(a), in terms of MEAN, according to the mean value of tied ranks obtained by the applied algorithm on all test functions, from best to worst, the applied algorithms can be sorted in the following order: CCLNNA, MVO, SSA, STOA, SOA, MFO, TSA and CHOP. Obviously, the proposed CCLNNA is best of all applied algorithms. In addition, from Table 5 and Fig. 7(b), in terms of STD, according to the mean value of

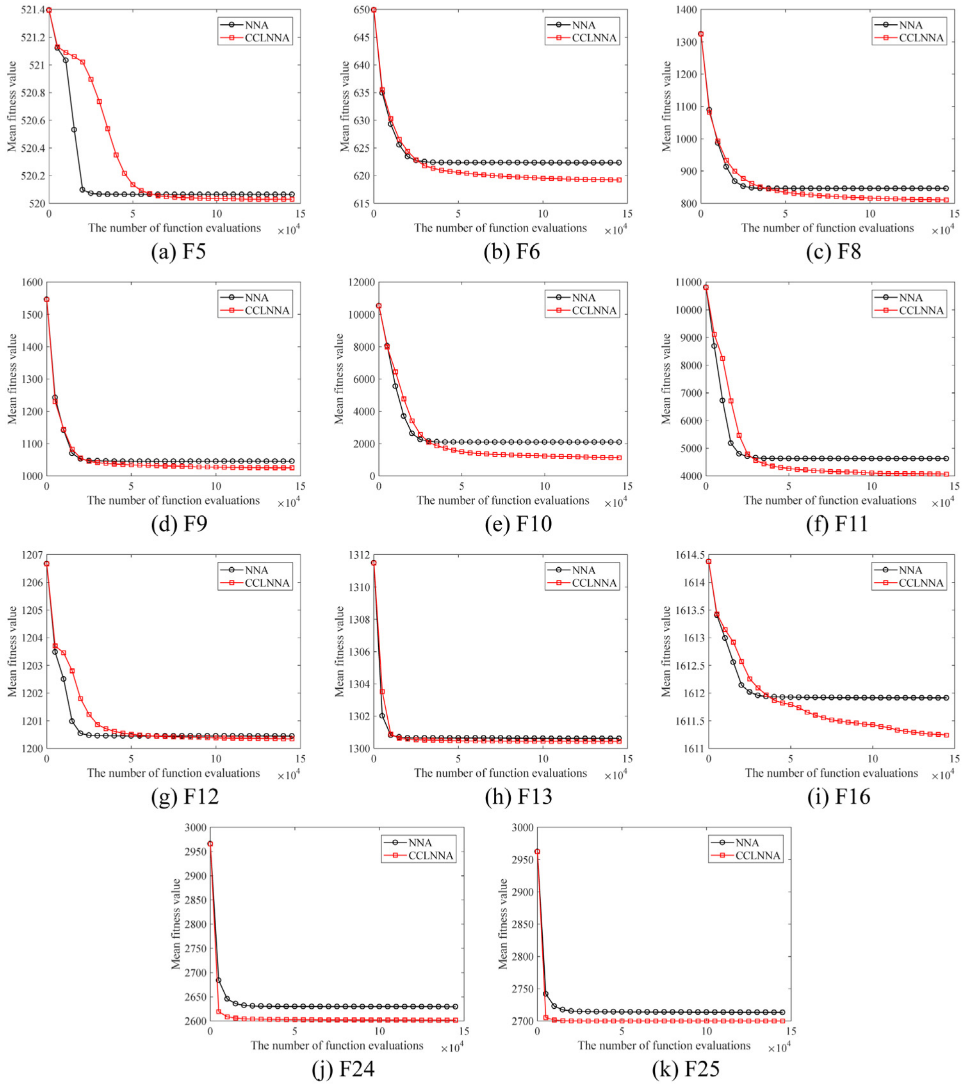

Fig. 6. Several typical convergence curves obtained by NNA and CCLNNA.

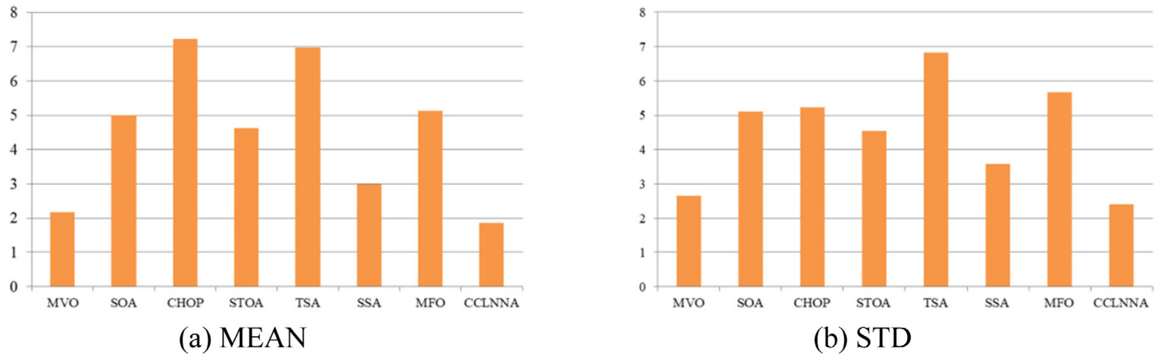

Fig. 7. Mean value of tied ranks obtained by the applied algorithm on all test functions.

tied ranks obtained by the applied algorithm on all test functions, from best to worst, the applied algorithms can be sorted in the following order: CCLNNA, MVO, SSA, STOA, SOA, CHOP, MFO and TSA. It is clear that the proposed CCLNNA is the best of all applied algorithms.

In order to further compare the convergence performance among the applied algorithms, Fig. 8 gives the convergence curves obtained by CCLNNA and the compared algorithms on F2, F5, F7, F8, F10, F13, F14, F16, F19, F23 and F25. Note that, for the selected test functions, F2 is the only unimodal function and the rest ten test functions are multimodal functions. From Fig. 8, MFO and TSA are easy to converge to local optimum with fast speed. STOA, CHOP and SOA have slow convergence. In addition, in terms of convergence performance, SSA and MVO show strong competitiveness while they still cannot compete with CCLNNA. Obviously, a conclusion can be drawn that CCLNNA has better

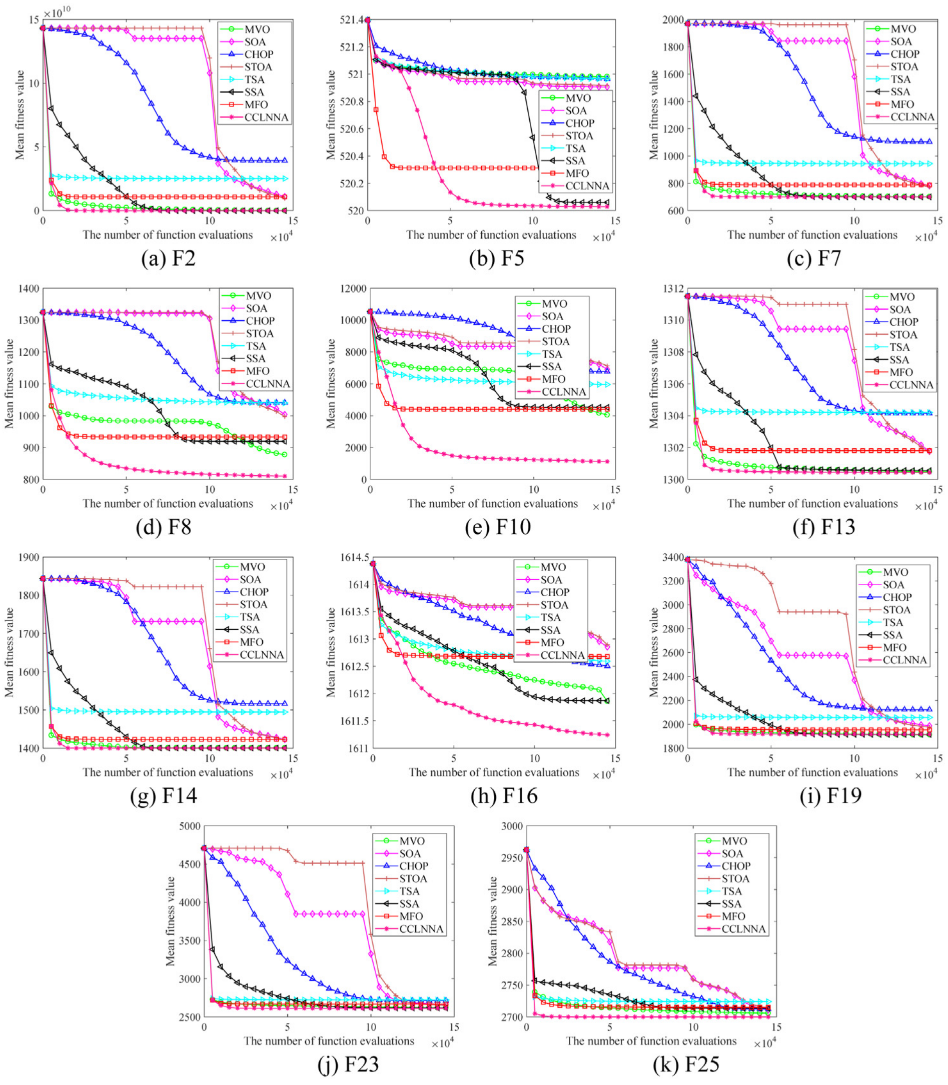

Fig. 8. Several typical convergence curves obtained by NNA and CCLNNA.

convergence performance than the compared algorithms on these curves.

Based on the above discussion, CCLNNA is obviously superior to the compared algorithms on most of test functions in terms of MEAN and STD. However, note that, SOA can offer better MEAN and STD than CCLNNA on F26; CHOP outperforms CCLNNA on F24 in terms of MEAN and STD; CCLNNA is inferior to STOA on F24 and F26 in terms of MEAN and STD; MEAN and STD of CCLNNA cannot compete with those of MVO on F3, F6, F15, F19, F20, F21, F27 and F28; SSA can get better MEAN and STD compared with CCLNNA on F15, F19, F20, F22 and F26. It is clearly that the experimental results are completely in accord with the NFL theorem. Further, although CCLNNA cannot beat the compared algorithms on each test function in terms of the considered indicators, CCLNNA can achieve better solutions than the compared algorithms on most of test functions according to the considered indicators. This fully demonstrates that CCLNNA has better overall performance than the compared SSA, MFO, MVO, SOA, CHOP, TSA, and STOA.

5. CCLNNA for real-world engineering design problems

In this section, CCLNNA is employed to solve three constrained real-world engineering design problems, i.e. rolling element bearing design problem, speed reducer design problem and pressure vessel design problem. When solving the constrained optimization problems, how to deal with the constraints is a big challenge. As done in [54], the penalty function is introduced to deal with the constraints in this work, whose core idea is that nonlinear constraints can be collapsed with the cost function into a response functional. By doing so, the constrained optimization problem can be transformed in an unconstrained optimization problem. For the three problems, population size and the maximum number of iterations of CCLNNA were set to 50 and 2000, respectively. In addition, in order to show the alternative of CCLNNA, the solutions of CCLNNA are compared with those of recently reported methods.

5.1. *Rolling element bearing design problem*

This problem is to maximize the dynamic load carting capacity of a rolling element bearing and the detailed description for this problem can be found in [55], whose schematic view is shown in Fig. 9 [56]. This problem includes 10 design variables, i.e. $\boldsymbol{x} = [D_m, D_b, Z, f_i, f_o, K_{Dmin}, K_{Dmax}, \varepsilon, e, \zeta]$. The meanings of these variables are as follows:D_m is the pitch diameter,D_b is the ball diameter,Z is the number of balls,f_i is the inner raceway curvature coefficients, f_o is the outer raceway curvature coefficients, K_{Dmin} is the minimum ball diameter limiter, K_{Dmax}

Table 1
The definition of CEC2014 test suite.

No.	Types	Name	Dimension	Limits	Optimum
F1	Unimodal functions	Rotated high conditioned elliptic function	30	[−100,100]	100
F2		Rotated bent cigar function	30	[−100,100]	200
F3		Rotated discus function	30	[−100,100]	300
F4	Multimodal functions	Shifted and rotated Rosenbrocks function	30	[−100,100]	400
F5		Shifted and rotated Ackleys function	30	[−100,100]	500
F6		Shifted and rotated Weierstrass function	30	[−100,100]	600
F7		Shifted and rotated Griewanks function	30	[−100,100]	700
F8		Shifted Rastrigins function	30	[−100,100]	800
F9		Six Hump Camel Back	30	[−100,100]	900
F10		Shifted and rotated Rastrigins function	30	[−100,100]	1000
F11		Shifted and rotated Schwefels function	30	[−100,100]	1100
F12		Shifted and rotated Katsuura function	30	[−100,100]	1200
F13		Shifted and rotated HappyCat function	30	[−100,100]	1300
F14		Shifted and rotated HGBat function	30	[−100,100]	1400
F15		Shifted and rotated Expanded Griewanks plus Rosenbrocks function	30	[−100,100]	1500
F16		Shifted and rotated Expanded Scaffers F6 function	30	[−100,100]	1600
F17	Hybrid functions	Hybrid function 1 (m = 3)	30	[−100,100]	1700
F18		Hybrid function 2 (m = 3)	30	[−100,100]	1800
F19		Hybrid function 3 (m = 4)	30	[−100,100]	1900
F20		Hybrid function 4 (m = 4)	30	[−100,100]	2000
F21		Hybrid function 5 (m = 5)	30	[−100,100]	2100
F22		Hybrid function 6 (m = 5)	30	[−100,100]	2200
F23	Composition functions	Composition function 1 (m = 5)	30	[−100,100]	2300
F24		Composition function 2 (m = 3)	30	[−100,100]	2400
F25		Composition function 3 (m = 3)	30	[−100,100]	2500
F26		Composition function 4 (m = 5)	30	[−100,100]	2600
F27		Composition function 5 (m = 5)	30	[−100,100]	2700
F28		Composition function 6 (m = 5)	30	[−100,100]	2800
F29		Composition function 7 (m = 3)	30	[−100,100]	2900
F30		Composition function 8 (m = 3)	30	[−100,100]	3000

Table 2
The parameters of the applied algorithms. "NFEs" means the number of function evaluations.

Year	Algorithm	Population size	Parameters	NFEs
2015	MFO	50	a linearly decreases from −1 to −2	$5000 \times D$
2016	MVO	50	$WEP_{\min} = 0.2$, $WEP_{\max} = 1.0$, $TDR_{\min} = 0.0$, $TDR_{\max} = 0.6$	$5000 \times D$
2017	SSA	50	$c_1 = 2\exp(-(\frac{4l}{L})^2)$	$5000 \times D$
2018	NNA	50	–	$5000 \times D$
2019	SOA	50	$A \in [0, 2]$, $f_c = 2$	$5000 \times D$
2019	STOA	50	$f_c = 2$	$5000 \times D$
2020	TSA	50	$p_{\min} = 1$, $p_{\min} = 4$	$5000 \times D$
2020	CHOA	50	$r_1, r_2 = rand$, $m = chaotic$	$5000 \times D$
Current work	CCLNNA	50	–	$5000 \times D$

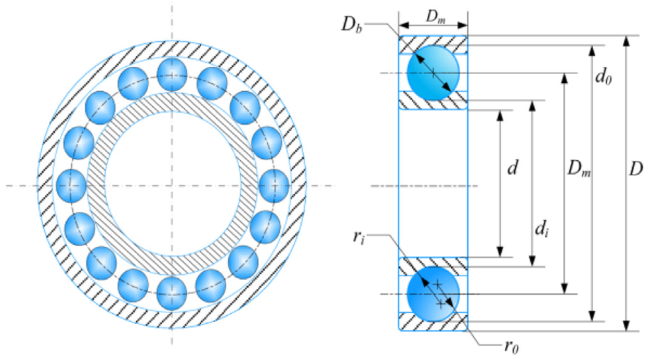

Fig. 9. Schematic view of the rolling element bearing design problem.

is the maximum ball diameter limiter, ε is the parameter for outer ring strength consideration,e is the parameter for mobility condition, and ζ is the bearing width limiter. In addition, nine constraints need to be considered. Mathematically, this problem can be expressed as

$$\text{Maximum } C_{\mathrm{d}} = \begin{cases} f_{\mathrm{c}} Z^{2/3} D_{\mathrm{b}}^{1.8}, \text{ if } D \le 25.4 \text{ mm} \\ 3.647 f_{\mathrm{c}} Z^{2/3} D_{\mathrm{b}}^{1.4}, \text{ if } D > 25.4 \text{ mm} \end{cases} \tag{18}$$

Subject to:

$$g_1(\boldsymbol{x}) = \frac{\phi_0}{2\sin^{-1}(D_{\mathrm{b}}/\mathrm{D_m})} - Z + 1 \ge 0 \tag{19}$$

$$g_2(\boldsymbol{x}) = 2D_{\mathrm{b}} - K_{\mathrm{D\,min}}(D - d) \ge 0 \tag{20}$$

$$g_3(\boldsymbol{x}) = K_{\mathrm{D\,max}}(D - d) - 2D_{\mathrm{b}} \ge 0 \tag{21}$$

$$g_4(\boldsymbol{x}) = \zeta B_{\mathrm{w}} - D_{\mathrm{b}} \le 0 \tag{22}$$

$$g_5(\boldsymbol{x}) = D_{\mathrm{m}} - 0.5(D + d) \ge 0 \tag{23}$$

$$g_6(\boldsymbol{x}) = (0.5 + e)(D + d) - D_{\mathrm{m}} \ge 0 \tag{24}$$

$$g_7(\boldsymbol{x}) = 0.5(D - D_{\mathrm{m}} - D_{\mathrm{b}}) - \varepsilon D_{\mathrm{b}} \ge 0 \tag{25}$$

$$g_8(\boldsymbol{x}) = f_{\mathrm{i}} - 0.515 \ge 0 \tag{26}$$

$$g_9(\boldsymbol{x}) = f_{\mathrm{o}} - 0.515 \ge 0 \tag{27}$$

Table 3
Experimental results obtained by NNA and CCLNNA on CEC 2014 test suite.

No.	Indicator	NNA	CCLNNA	No.	Indicator	NNA	CCLNNA
F1	MEAN	6.268E+06	**2.518E+06**	F16	MEAN	1.612E+03	**1.611E+03**
	STD	4.382E+06	**1.130E+06**		STD	**5.362E−01**	5.734E−01
F2	MEAN	1.212E+04	**1.032E+04**	F17	MEAN	6.410E+05	**2.563E+05**
	STD	1.265E+04	**5.330E+03**		STD	4.503E+05	**1.111E+05**
F3	MEAN	3.878E+03	**3.094E+03**	F18	MEAN	1.037E+04	**2.851E+03**
	STD	3.016E+03	**2.138E+03**		STD	8.655E+03	**1.091E+03**
F4	MEAN	5.131E+02	**4.871E+02**	F19	MEAN	1.924E+03	**1.916E+03**
	STD	3.193E+01	**2.585E+01**		STD	2.637E+01	**1.457E+01**
F5	MEAN	5.201E+02	**5.200E+02**	F20	MEAN	1.393E+04	**4.407E+03**
	STD	7.926E−02	**4.792E−02**		STD	4.381E+03	**1.850E+03**
F6	MEAN	6.223E+02	**6.191E+02**	F21	MEAN	2.220E+05	**1.366E+05**
	STD	2.988E+00	**2.918E+00**		STD	2.304E+05	**8.054E+04**
F7	MEAN	**7.000E+02**	**7.000E+02**	F22	MEAN	2.757E+03	**2.664E+03**
	STD	3.032E−02	**1.769E−02**		STD	2.088E+02	**1.768E+02**
F8	MEAN	8.461E+02	**8.097E+02**	F23	MEAN	2.615E+03	**2.613E+03**
	STD	1.469E+01	**2.809E+00**		STD	**2.912E−04**	1.630E+01
F9	MEAN	1.045E+03	**1.025E+03**	F24	MEAN	2.629E+03	**2.602E+03**
	STD	3.436E+01	**2.427E+01**		STD	1.309E+01	**3.031E−01**
F10	MEAN	2.089E+03	**1.127E+03**	F25	MEAN	2.713E+03	**2.700E+03**
	STD	3.526E+02	**1.208E+02**		STD	6.272E+00	**2.040E−05**
F11	MEAN	4.624E+03	**4.051E+03**	F26	MEAN	**2.701E+03**	2.703E+03
	STD	6.914E+02	**5.529E+02**		STD	**1.534E−01**	1.406E+01
F12	MEAN	**1.200E+03**	**1.200E+03**	F27	MEAN	**3.272E+03**	3.297E+03
	STD	1.435E−01	**1.230E−01**		STD	2.550E+02	**2.284E+02**
F13	MEAN	1.301E+03	**1.300E+03**	F28	MEAN	3.989E+03	**3.940E+03**
	STD	1.448E−01	**7.873E−02**		STD	**2.299E+02**	2.958E+02
F14	MEAN	**1.400E+03**	**1.400E+03**	F29	MEAN	1.879E+05	**4.342E+03**
	STD	2.689E−01	**3.391E−02**		STD	1.296E+06	**4.678E+02**
F15	MEAN	1.519E+03	**1.516E+03**	F30	MEAN	1.111E+04	**5.852E+03**
	STD	6.129E+00	**5.811E+00**		STD	6.961E+03	**6.925E+02**

where $\gamma = \frac{D_b}{D_m}, f_i = \frac{r_i}{D_b}, f_o = \frac{r_o}{D_b}, T = D - d - 2D_b, D = 160,$ $d = 90, B_w = 30, D = 160, r_i = r_o = 11.033,$ $0.5\,(D+d) \le D_m \le 0.6\,(D+d)\,, 0.15\,(D-d) \le D_b \le 0.45\,(D-d)\,,\ 4 \le Z \le 50,\ 0.515 \le f_i \le 0.6,\ 0.515 \le f_o \le 0.6,$ $0.4 \le K_{D\min} \le 0.5, 0.6 \le K_{D\max} \le 0.7, 0.3 \le e \le 0.4, 0.02 \le \varepsilon \le 0.1, 0.6 \le \zeta \le 0.85$

$$f_c = 37.91\left[1+\left\{1.04\left(\frac{1-\gamma}{1+\gamma}\right)^{1.72}\left(\frac{f_i\,(2f_o-1)}{f_o\,(2f_i-1)}\right)^{0.41}\right\}^{10/3}\right]^{-0.3}$$
$$\times\left(\frac{\gamma^{0.3}\,(1-\gamma)^{1.39}}{(1+\gamma)^{1./3}}\right)\left[\frac{2f_i}{2f_i-1}\right]^{0.41},$$

$$\phi_0 = 2\pi - 2\cos^{-1}$$
$$\frac{\left[\{(D-d)/2-3(T/4)\}^2+\{D/2-(T/4)-D_b\}^2-\{d/2+(T/4)\}^2\right]}{2\,\{(D-d)/2-3(T/4)\}\,\{D/2-(T/4)-D_b\}}.$$

Some researchers have used different metaheuristic methods to solve this problems, such as TLBO [17], EPO [57], SOA [48], STOA [18], CMVHHO [58], SHO [59] and HHO [60]. Table 6 shows the best solutions obtained by CCLNNA and the compared algorithms. From Table 6, the proposed CCLNNA can get the optimal cost, i.e. 85548.2300. In addition, Table 7 presents the constraints with the optimal cost achieved by CCLNNA. As can be seen from Table 7, the obtained constraints meet Eqs. (17)–(25), which means the optimal cost offered by CCLNNA is effective. In addition, the solutions of EPSO, SOA, STOA and SHO are very competitiveness.

5.2. Speed reducer design problem

This problem is to minimize the weight of a speed reducer, whose schematic view can be found in Fig. 10 [61]. This problem has seven variables, i.e. $\boldsymbol{x} = [b, m, z, l_1, l_2, d_1, d_2]$. The meanings of these variables are as follows: $b(x_1)$ is face width, $m(x_2)$ is module of teeth, $z(x_3)$ is the number of teeth in the pinion, $l_1(x_4)$ is the length of the first shaft between bearings, $l_2(x_5)$ is the length of the second shaft between bearings, $d_1(x_6)$ is diameter of first shafts and $d_2(x_7)$ is diameter of second shafts. In addition, 11 constraints need to be considered. The mathematical formulation can be expressed as

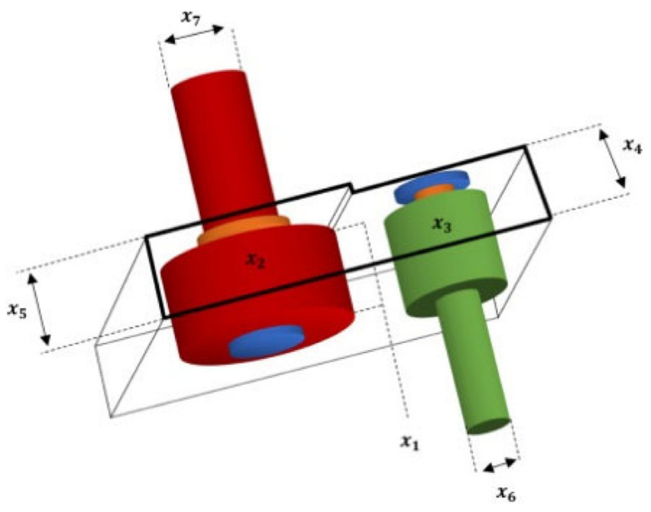

Fig. 10. Schematic view of the speed reducer design problem.

$$\text{Minimize}\, f(\boldsymbol{x}) = 0.7854x_1x_2^2\left(3.3333x_3^2+14.9334x_3-43.0934\right)$$
$$-1.508x_1\left(x_6^2+x_7^2\right)+7.4777\left(x_6^3+x_7^3\right)+0.7854\left(x_4x_6^2+x_5x_7^2\right) \quad (28)$$

Table 4
Experimental results obtained by CCLNNA and seven compared methods on CEC 2014 test suite.

No.	Indicator	MVO	SOA	CHOP	STOA	TSA	SSA	MFO	CCLNNA
F1	MEAN	4.751E+06	5.192E+07	4.738E+08	4.948E+07	3.404E+08	4.650E+06	6.046E+07	**2.518E+06**
	STD	1.853E+06	3.765E+07	1.061E+08	2.426E+07	1.855E+08	2.082E+06	6.565E+07	**1.130E+06**
F2	MEAN	4.810E+04	9.918E+09	3.922E+10	8.505E+09	2.512E+10	1.308E+04	1.065E+10	**1.032E+04**
	STD	1.462E+04	4.543E+09	7.184E+09	3.251E+09	7.539E+09	1.176E+04	6.878E+09	**5.330E+03**
F3	MEAN	**7.407E+02**	3.817E+04	6.275E+04	3.370E+04	4.424E+04	1.324E+04	8.720E+04	3.094E+03
	STD	**1.310E+02**	9.137E+03	7.356E+03	7.943E+03	9.907E+03	5.211E+03	3.794E+04	2.138E+03
F4	MEAN	5.058E+02	9.303E+02	2.580E+03	8.424E+02	2.383E+03	5.060E+02	1.213E+03	**4.871E+02**
	STD	2.791E+01	1.919E+02	1.244E+03	1.265E+02	1.218E+03	3.531E+01	8.619E+02	**2.585E+01**
F5	MEAN	5.201E+02	5.209E+02	5.210E+02	5.209E+02	5.210E+02	5.201E+02	5.203E+02	**5.200E+02**
	STD	7.114E−02	8.469E−02	5.071E−02	7.603E−02	5.557E−02	7.545E−02	1.699E−01	**4.792E−02**
F6	MEAN	**6.105E+02**	6.270E+02	6.351E+02	6.274E+02	6.305E+02	6.203E+02	6.229E+02	6.191E+02
	STD	**2.797E+00**	3.483E+00	2.485E+00	3.315E+00	3.516E+00	4.771E+00	3.295E+00	2.918E+00
F7	MEAN	7.002E+02	7.751E+02	1.104E+03	7.623E+02	9.435E+02	**7.000E+02**	7.888E+02	**7.000E+02**
	STD	5.945E−02	3.837E+01	8.199E+01	2.812E+01	8.465E+01	**9.923E−03**	5.535E+01	1.769E−02
F8	MEAN	8.759E+02	9.617E+02	1.041E+03	9.532E+02	1.039E+03	9.192E+02	9.335E+02	**8.097E+02**
	STD	1.639E+01	2.656E+01	2.282E+01	3.011E+01	3.403E+01	3.448E+01	2.961E+01	**2.809E+00**
F9	MEAN	9.954E+02	1.087E+03	1.167E+03	1.068E+03	1.195E+03	1.038E+03	1.093E+03	**1.025E+03**
	STD	2.553E+01	3.146E+01	2.462E+01	3.425E+01	5.992E+01	3.689E+01	3.872E+01	**2.427E+01**
F10	MEAN	4.002E+03	5.006E+03	6.769E+03	5.149E+03	5.949E+03	4.531E+03	4.411E+03	**1.127E+03**
	STD	5.587E+02	8.123E+02	9.191E+02	7.832E+02	7.304E+02	7.264E+02	8.346E+02	**1.208E+02**
F11	MEAN	4.117E+03	5.788E+03	8.436E+03	5.559E+03	6.625E+03	4.892E+03	5.234E+03	**4.051E+03**
	STD	6.220E+02	7.537E+02	**3.730E+02**	7.585E+02	6.379E+02	6.090E+02	7.130E+02	5.529E+02
F12	MEAN	**1.200E+03**	1.202E+03	1.203E+03	1.202E+03	1.202E+03	1.201E+03	**1.200E+03**	**1.200E+03**
	STD	2.266E−01	5.500E−01	2.847E−01	4.696E−01	3.115E−01	2.637E−01	2.504E−01	**1.230E−01**
F13	MEAN	**1.300E+03**	1.301E+03	1.304E+03	1.301E+03	1.304E+03	1.301E+03	1.302E+03	**1.300E+03**
	STD	1.156E−01	9.085E−01	7.815E−01	8.003E−01	7.718E−01	1.223E−01	1.244E+00	**7.873E−02**
F14	MEAN	1.401E+03	1.419E+03	1.516E+03	1.419E+03	1.494E+03	**1.400E+03**	1.423E+03	**1.400E+03**
	STD	3.366E−01	9.505E+00	2.360E+01	8.806E+00	2.951E+01	2.214E−01	1.798E+01	**3.391E−02**
F15	MEAN	**1.508E+03**	4.065E+03	7.533E+04	2.921E+03	1.744E+04	1.509E+03	1.025E+05	1.516E+03
	STD	**2.426E+00**	4.562E+03	6.987E+04	2.225E+03	1.809E+04	3.470E+00	3.141E+05	5.811E+00
F16	MEAN	1.612E+03	1.613E+03	1.612E+03	1.613E+03	1.613E+03	1.612E+03	1.613E+03	**1.611E+03**
	STD	4.677E−01	4.786E−01	**2.929E−01**	4.875E−01	3.811E−01	5.416E−01	4.651E−01	5.734E−01
F17	MEAN	2.642E+05	1.399E+06	1.688E+07	1.331E+06	1.657E+07	2.907E+05	2.788E+06	**2.563E+05**
	STD	1.838E+05	1.036E+06	1.345E+07	9.369E+05	2.252E+07	1.587E+05	3.540E+06	**1.111E+05**
F18	MEAN	9.744E+03	3.783E+07	4.035E+08	3.260E+07	5.249E+08	7.818E+03	2.061E+07	**2.851E+03**
	STD	8.053E+03	2.443E+07	5.840E+08	1.914E+07	9.516E+08	6.412E+03	8.634E+07	**1.091E+03**
F19	MEAN	**1.911E+03**	1.981E+03	2.122E+03	1.969E+03	2.057E+03	1.914E+03	1.953E+03	1.916E+03
	STD	**2.310E+00**	3.164E+01	1.102E+02	3.086E+01	8.047E+01	2.311E+00	4.941E+01	1.457E+01
F20	MEAN	**2.380E+03**	2.044E+04	6.062E+04	1.937E+04	4.260E+04	3.643E+03	5.939E+04	4.407E+03
	STD	**9.320E+01**	7.257E+03	2.712E+04	6.688E+03	4.016E+04	1.195E+03	3.348E+04	1.850E+03
F21	MEAN	**8.915E+04**	5.257E+05	7.746E+06	6.390E+05	4.462E+06	1.242E+05	8.184E+05	1.366E+05
	STD	**6.550E+04**	3.466E+05	4.191E+06	6.568E+05	4.498E+06	9.242E+04	1.297E+06	8.054E+04
F22	MEAN	**2.610E+03**	2.716E+03	3.029E+03	2.683E+03	3.860E+03	2.639E+03	2.962E+03	2.664E+03
	STD	1.940E+02	1.899E+02	**1.252E+02**	1.940E+02	4.964E+03	1.658E+02	2.334E+02	1.768E+02
F23	MEAN	2.617E+03	2.659E+03	2.713E+03	2.649E+03	2.722E+03	2.618E+03	2.668E+03	**2.613E+03**
	STD	1.798E+00	1.833E+01	4.657E+01	1.267E+01	7.925E+01	**2.892E+00**	3.505E+01	1.630E+01
F24	MEAN	2.629E+03	**2.600E+03**	**2.600E+03**	**2.600E+03**	2.603E+03	2.636E+03	2.672E+03	2.602E+03
	STD	1.104E+01	**1.268E−03**	5.725E−03	4.755E−03	1.283E+01	7.892E+00	2.783E+01	3.031E−01
F25	MEAN	2.706E+03	2.710E+03	2.713E+03	2.712E+03	2.724E+03	2.714E+03	2.715E+03	2.700E+03
	STD	1.597E+00	8.213E+00	1.161E+01	7.709E+00	7.993E+00	4.129E+00	7.192E+00	2.040E−05
F26	MEAN	2.730E+03	**2.701E+03**	2.801E+03	**2.701E+03**	2.780E+03	**2.701E+03**	**2.701E+03**	2.703E+03
	STD	6.228E+01	3.934E−01	9.033E+01	2.817E−01	6.834E+01	**1.111E−01**	8.638E−01	1.406E+01
F27	MEAN	**3.228E+03**	3.659E+03	3.894E+03	3.705E+03	3.638E+03	3.397E+03	3.565E+03	3.297E+03
	STD	1.254E+02	2.145E+02	**1.175E+02**	1.525E+02	3.595E+02	1.922E+02	2.122E+02	2.284E+02
F28	MEAN	3.880E+03	4.054E+03	5.370E+03	4.071E+03	6.877E+03	3.960E+03	**3.838E+03**	3.940E+03
	STD	2.464E+02	1.902E+02	2.817E+02	2.346E+02	6.982E+02	2.962E+02	**1.142E+02**	2.958E+02
F29	MEAN	2.340E+05	4.352E+06	1.980E+07	3.676E+06	4.910E+07	3.321E+06	2.118E+06	**4.342E+03**
	STD	1.501E+06	4.381E+06	1.443E+07	4.099E+06	4.396E+07	6.200E+06	3.470E+06	**4.678E+02**
F30	MEAN	1.074E+04	8.816E+04	4.652E+05	8.425E+04	2.931E+05	1.402E+04	3.428E+04	**5.852E+03**
	STD	3.056E+03	5.195E+04	1.660E+05	4.086E+04	2.737E+05	4.802E+03	2.690E+04	**6.925E+02**

Subject to:

$$g_1(\boldsymbol{x}) = \frac{27}{x_1 x_2^2 x_3} - 1 \leq 0 \tag{29}$$

$$g_2(\boldsymbol{x}) = \frac{397.5}{x_1 x_2^2 x_3} - 1 \leq 0 \tag{30}$$

$$g_3(\boldsymbol{x}) = \frac{1.93 x_4^3}{x_2 x_6^4 x_3} - 1 \leq 0 \tag{31}$$

$$g_4(\boldsymbol{x}) = \frac{1.93 x_5^3}{x_2 x_7^4 x_3} - 1 \leq 0 \tag{32}$$

$$g_5(\boldsymbol{x}) = \frac{\left(\left(\frac{745 x_5}{x_2 x_3}\right)^2 + 157.5 \times 10^6\right)^{1/2}}{85 x_7^3} - 1 \leq 0 \tag{33}$$

$$g_6(\boldsymbol{x}) = \frac{x_2 x_3}{40} - 1 \leq 0 \tag{34}$$

$$g_7(\boldsymbol{x}) = \frac{5 x_2}{x_1} - 1 \leq 0 \tag{35}$$

$$g_8(\boldsymbol{x}) = \frac{x_1}{12 x_2} - 1 \leq 0 \tag{36}$$

$$g_9(\boldsymbol{x}) = \frac{1.5 x_6 + 1.9}{x_4} - 1 \leq 0 \tag{37}$$

Table 5
Ranking results of the applied algorithms on the basis of mean values.

No.	MEAN								STD							
	MVO	SOA	CHOP	STOA	TSA	SSA	MFO	CCLNNA	MVO	SOA	CHOP	STOA	TSA	SSA	MFO	CCLNNA
F1	3	5	8	4	7	2	6	1	2	5	7	4	8	3	6	1
F2	3	5	8	4	7	2	6	1	3	5	7	4	8	2	6	1
F3	1	5	7	4	6	3	8	2	1	6	4	5	7	3	8	2
F4	2	5	8	4	7	3	6	1	2	5	8	4	7	3	6	1
F5	2.5	5.5	7.5	5.5	7.5	2.5	4	1	4	7	2	6	3	5	8	1
F6	1	5	8	6	7	3	4	2	2	6	1	5	7	8	4	3
F7	3	5	8	4	7	1.5	6	1.5	3	5	7	4	8	1	6	2
F8	2	6	8	5	7	3	4	1	2	4	3	6	7	8	5	1
F9	1	5	7	4	8	3	6	2	3	4	2	5	8	6	7	1
F10	2	5	8	6	7	4	3	1	2	6	8	5	4	3	7	1
F11	2	6	8	5	7	3	4	1	4	7	1	8	5	3	6	2
F12	2	6	8	6	6	4	2	2	2	8	5	7	6	4	3	1
F13	1.5	4	7.5	4	7.5	4	6	1.5	2	7	5	6	4	3	8	1
F14	3	4.5	8	4.5	7	1.5	6	1.5	3	5	7	4	8	2	6	1
F15	1	5	7	4	6	2	8	3	1	5	7	4	6	2	8	3
F16	3	6.5	3	6.5	6.5	3	6.5	1	4	5	1	6	2	7	3	8
F17	2	5	8	4	7	3	6	1	3	5	7	4	8	2	6	1
F18	3	6	7	5	8	2	4	1	3	5	7	4	8	2	6	1
F19	1	6	8	5	7	2	4	3	1	5	8	4	7	2	6	3
F20	1	5	8	4	6	2	7	3	1	5	6	4	8	2	7	3
F21	1	4	8	5	7	2	6	3	1	4	7	5	8	3	6	2
F22	1	5	7	4	8	2	6	3	5.5	4	1	5.5	8	2	7	3
F23	2	5	7	4	8	3	6	1	1	5	7	3	8	2	6	4
F24	6	2	2	2	5	7	8	4	6	1	3	2	7	5	8	4
F25	2	3	5	4	8	6	7	1	2	7	8	5	6	3	4	1
F26	6	2.5	8	2.5	7	2.5	2.5	5	6	3	8	2	7	1	4	5
F27	1	6	8	7	5	3	4	2	2	6	1	3	8	4	5	7
F28	2	5	7	6	8	4	1	3	4	2	5	3	8	7	1	6
F29	2	6	7	5	8	4	3	1	2	5	7	4	8	6	3	1
F30	2	6	8	5	7	3	4	1	2	6	7	5	8	3	4	1
Avg.	2.167	5.000	7.233	4.633	6.983	3.000	5.133	1.850	2.650	5.100	5.233	4.550	6.833	3.567	5.667	2.400

Table 6
The optimal solutions obtained by CCLNNA and the reported methods for rolling element design bearing problem.

Parameter	TLBO	EPO	SOA	STOA	CMVHHO	SHO	HHO	CCLNNA
D_m	125.7191	125	125	125	125.01812	125	125.0000	125.71924
D_b	21.42559	21.41890	21.41892	21.41890	21.27836	21.04732	21.00000	21.425447
Z	11	10.94113	10.94123	10.94113	10.86415	10.93268	11.092073	11
f_i	0.515	0.515	0.515	0.515	0.515	0.515	0.51500	0.515
f_o	0.515	0.515	0.515	0.515	0.518045	0.515	0.51500	0.515
K_{Dmin}	0.424266	0.4	0.4	0.4	0.456659	0.4	0.40000	0.45755764
K_{Dmax}	0.633948	0.7	0.7	0.7	0.639656	0.7	0.60000	0.69999641
ε	0.3	0.3	0.3	0.3	0.301777	0.3	0.30000	0.30000069
e	0.068858	0.02	0.02	0.02	0.021893	0.02	0.050474	0.07802888
ζ	0.799498	0.6	0.6	0.6	0.646298	0.6	0.600000	0.60000355
Opt. cost	81859.74	85067.983.	85068.052	85067.983	83800.078	85054.032	83011.883	**85548.2300**

Table 7
The optimal constraints with the optimal cost obtained by CCLNNA for rolling element design bearing problem.

$g(x)$	$g_1(x)$	$g_2(x)$	$g_3(x)$	$g_4(x)$	$g_5(x)$	$g_6(x)$	$g_7(x)$	$g_8(x)$	$g_9(x)$
Value	1.84E−5	1.08E+1	6.15E+0	−3.43E+0	0.72E+0	1.88E+1	1.01E−5	0	8.34E−6

$$g_{10}(\boldsymbol{x}) = \frac{1.1x_7 + 1.9}{x_5} - 1 \le 0 \tag{38}$$

$$g_{11}(\boldsymbol{x}) = \frac{\left(\left(\frac{745x_4}{x_2x_3}\right)^2 + 16.9 \times 10^6\right)^{1/2}}{110x_6^3} - 1 \le 0 \tag{39}$$

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

Speed reducer design problem is a very classical engineering design problem, which has been solved by many metaheuristic methods, such as SHO [59], GWO [62], SCA [27], MBA [15], PVS [16] and HGSO [63]. Table 8 shows the optimal solutions obtained by CCLNNA and the compared algorithms. From Table 8, the proposed CCLNNA can find the optimal weight, i.e. 2994.4716. In addition, Table 9 gives the optimal constraints with optimal weight obtained by CCLNNA. According to Table 9, the optimal constraints meet Eqs. (27)–(37), which proves the effectiveness of the optimal weight obtained by CCLNNA. In addition, the optimal weight of MBA is 2994.482453, which is very similar with that of CCLNNA. The optimal weights of GWO, MVO and SCA are significantly inferior to the other algorithms, which means the three algorithms are not suitable for solving this problem.

5.3. Pressure vessel design problem

As an often used benchmark engineering design problem, pressure vessel design problem is to minimize the total cost of a pressure vessel considering the cost of material, forming and welding. This problem considers four design variables $\boldsymbol{x} =$

Table 8
The optimal solutions obtained by CCLNNA and the reported methods for speed reducer design problem.

Parameter	SHO	GWO	MVO	SCA	MBA	PVS	HGSO	CCLNNA
x_1	3.50159	3.50669	3.508502	3.508755	3.5	3.49999	3.498	3.5
x_2	0.7	0.7	0.7	0.7	0.7	0.6999	0.71	0.7
x_3	17	17	17	17	17	17	17.02	17
x_4	7.3	7.380933	7.392843	7.3	7.300033	7.3	7.67	7.3
x_5	7.8	7.815726	7.816034	7.8	7.715772	7.8	7.81	7.715
x_6	3.35127	3.357847	3.358073	3.461020	3.350218	3.3502	3.36	3.35
x_7	5.28874	5.286768	5.286777	5.289213	5.286654	5.2866	5.289	5.2867
Opt. weight	2998.5507	3001.288	3002.928	3030.563	2994.482453	2996.3481	2997.10	**2994.4716**

Table 9
The optimal constraints with the optimal cost obtained by CCLNNA for speed reducer problem.

$g(x)$	$g_1(x)$	$g_2(x)$	$g_3(x)$	$g_4(x)$	$g_5(x)$	$g_6(x)$	$g_7(x)$	$g_8(x)$	$g_9(x)$	$g_{10}(x)$	$g_{11}(x)$
Value	−0.074	−0.1980	−0.4992	−0.9046	−4.78E−7	−1.32E−7	−7.025	−5.71E−8	−0.5833	−0.0513	−8.33E−7

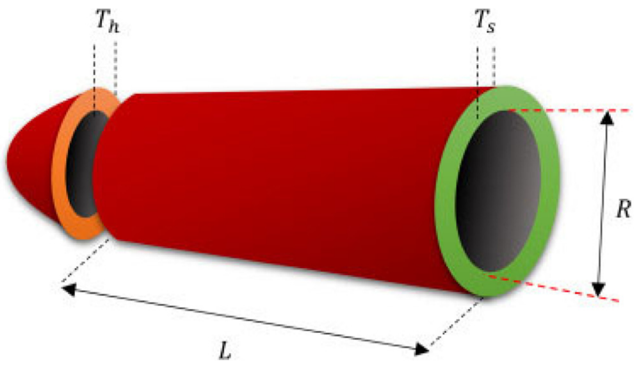

Fig. 11. Schematic view of the pressure vessel design problem.

$[T_s, T_h, R, L]$ as shown in Fig. 11. The meanings of these variables are as follows: $T_s\,(x_1)$ is the thickness of the shell, $T_h\,(x_2)$ is the thickness of the head, $R\,(x_3)$ is the inner radius, and $L\,(x_4)$ is the length of the cylindrical section of the vessel. The mathematical formulation of this problem can be defined as

$$\text{Minimize} f(\boldsymbol{x}) = 0.6224x_1x_3x_4 + 1.7781x_2x_3^2 + 3.1661x_1^2x_4 + 19.84x_1^2x_3 \tag{40}$$

$$g_1(\boldsymbol{x}) = -x_1 + 0.0193x_3 \le 0 \tag{41}$$

$$g_2(\boldsymbol{x}) = -x_2 + 0.00954x_3 \le 0 \tag{42}$$

$$g_3(\boldsymbol{x}) = -\pi x_3^2x_4 - \frac{4}{3}\pi x_3^3 + 1296,000 \le 0 \tag{43}$$

$$g_4(\boldsymbol{x}) = x_4 - 240 \le 0 \tag{44}$$

where $0 \le x_i \le 100, i = 1, 2; 10 \le x_i \le 200, i = 3, 4$

Recently, many solutions obtained by metaheuristic methods to this problem have been reported. These method include HHO [60], GWO [62], MFO [52], WEO [64], WOA [65], PRO [66] and SC-GWO [67]. Table 10 presents the solutions obtained by CCLNNA and the compared methods. From Table 10, CCLNNA can achieve the best cost, i.e. 5996.291. In addition, Table 11 displays the optimal constraints with the optimal cost obtained by CCLNNA. From Table 11, the obtained optimal constraints meet Eqs. (39)–(42), which indicates the obtained solution by CCLNNA is effective. Besides, by observing Table 10 carefully, the optimal costs of HHO, GWO, MFO, WEO, WOA, PRO and SC-GWO are obviously more than that of CCLNNA, which demonstrates that CCLNNA has a significant advantage over the eight algorithms for solving this problem.

6. Conclusion

This work reports a novel metaheuristic method called chaotic neural network algorithm with competitive learning (CCLNNA) for global optimization. CCLNNA is a new variant of neural network algorithm (NNA), which is to improve convergence rate and global search ability of NNA. In CCLNNA, population is first divided into two subpopulations. Then the two subpopulations are optimized by the designed different learning strategies to balance exploration and exploitation of CCLNNA. In addition, chaos theory is introduced, which is used to improve the ability of CCLNNA to escape from the local optimum. In order to verify the effectiveness of the improved strategies, the performance of CCLNNA for numerical optimization problems is investigated by solving challenging CEC 2014 test suite. Experimental results show CCLNNA can beat NNA on 25 of 30 test functions and outperforms the other seven compared algorithms on nearly half of test functions. Further, CCLNNA is employed for solving three challenging real-world engineering design problems. Experimental results prove the superiority of the solutions obtained by CCLNNA by comparing with the recently reported solutions.

In further research, we focus on the following two aspects. Firstly, we intend to use CCLNNA to solve more real-world engineering optimization problems, such as the opinion leader detection in online social network, multi-controller placement in software-defined networks, multi-robot path planning and the deployment optimization of multi-unmanned aerial vehicle. Secondly, as the basis of deep learning, neural network technique has been growing very fast in recent years. Although CCLNNA can significantly improve the global search ability of NNA, further work is needed to further improve performance of NNA. Thus we will try to propose some other variants of NNA in further study, such as introducing reinforcement learning mechanism to control its modification factor and using opposition learning to perform its initialization.

CRediT authorship contribution statement

Yiying Zhang: Conceptualization, Methodology, Writing – original draft, Software.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgment

This research did not receive any specific grant from funding agencies in the public, commercial, or not-for-profit sectors.

Table 10
The optimal solutions obtained by CCLNNA and the reported methods for pressure vessel design problem.

Parameter	HHO	GWO	MFO	WEO	WOA	PRO	SC-GWO	CCLNNA
x_1	0.81758383	0.8125	0.8125	0.8125	0.8125	0.7445	0.8125	0.838164
x_2	0.4072927	0.4345	0.4375	0.4375	0.4375	0.4424	0.4375	0.414392
x_3	42.09174576	42.089181	42.098445	42.098444	42.0982699	38.489983	42.0984	43.42817
x_4	176.7196352	176.758731	176.636596	176.636622	176.638998	200.0000	176.6370	160.8277
Opt. cost	6000.46259	6051.5639	6059.7143	6059.71	6059.741	6050.7134	6059.7179	**5996.291**

Table 11
The optimal constraints with the optimal cost obtained by CCLNNA for pressure vessel problem.

$g(x)$	$g_1(x)$	$g_2(x)$	$g_3(x)$	$g_4(x)$
Value	−3.94E−07	−8.73E−05	−3.75E−01	−7.92E+01

References

[1] Y. Liu, W. Wang, N. Ghadimi, Electricity load forecasting by an improved forecast engine for building level consumers, Energy 139 (2017) 18–30, http://dx.doi.org/10.1016/j.energy.2017.07.150.

[2] R. Abbassi, A. Abbassi, A.A. Heidari, S. Mirjalili, An efficient salp swarm-inspired algorithm for parameters identification of photovoltaic cell models, Energy Convers. Manag. 179 (2019) 362–372, http://dx.doi.org/10.1016/j.enconman.2018.10.069.

[3] D. Gao, G. Wang, W. Pedrycz, Solving fuzzy job-shop scheduling problem using DE algorithm improved by a selection mechanism, IEEE Trans. Fuzzy Syst. (2020) 1, http://dx.doi.org/10.1109/TFUZZ.2020.3003506.

[4] P.K. Das, H.S. Behera, B.K. Panigrahi, A hybridization of an improved particle swarm optimization and gravitational search algorithm for multi-robot path planning, Swarm Evol. Comput. 28 (2016) 14–28, http://dx.doi.org/10.1016/j.swevo.2015.10.011.

[5] H. Leng, X. Li, J. Zhu, H. Tang, Z. Zhang, N. Ghadimi, A new wind power prediction method based on ridgelet transforms, hybrid feature selection and closed-loop forecasting, Adv. Eng. Inf. 36 (2018) 20–30, http://dx.doi.org/10.1016/j.aei.2018.02.006.

[6] G. Aghajani, N. Ghadimi, Multi-objective energy management in a micro-grid, Energy Rep. 4 (2018) 218–225, http://dx.doi.org/10.1016/j.egyr.2017.10.002.

[7] Z. Xing, An improved emperor penguin optimization based multilevel thresholding for color image segmentation, Knowl.-Based Syst. 194 (2020) 105570, http://dx.doi.org/10.1016/j.knosys.2020.105570.

[8] M. Hamian, A. Darvishan, M. Hosseinzadeh, M.J. Lariche, N. Ghadimi, A. Nouri, A framework to expedite joint energy-reserve payment cost minimization using a custom-designed method based on mixed integer genetic algorithm, Eng. Appl. Artif. Intell. 72 (2018) 203–212, http://dx.doi.org/10.1016/j.engappai.2018.03.022.

[9] L. Li, L. Jiao, J. Zhao, R. Shang, M. Gong, Quantum-behaved discrete multi-objective particle swarm optimization for complex network clustering, Pattern Recognit. 63 (2017) 1–14, http://dx.doi.org/10.1016/j.patcog.2016.09.013.

[10] L. Wu, Q. Liu, X. Tian, J. Zhang, W. Xiao, A new improved fruit fly optimization algorithm IAFOA and its application to solve engineering optimization problems, Knowl.-Based Syst. 144 (2018) 153–173, http://dx.doi.org/10.1016/j.knosys.2017.12.031.

[11] F. Mirzapour, M. Lakzaei, G. Varamini, M. Teimourian, N. Ghadimi, A new prediction model of battery and wind-solar output in hybrid power system, J. Ambient Intell. Humaniz. Comput. 10 (2019) 77–87, http://dx.doi.org/10.1007/s12652-017-0600-7.

[12] T.Y. Tan, L. Zhang, S.C. Neoh, C.P. Lim, Intelligent skin cancer detection using enhanced particle swarm optimization, Knowl.-Based Syst. 158 (2018) 118–135, http://dx.doi.org/10.1016/j.knosys.2018.05.042.

[13] P. Kuila, P.K. Jana, Energy efficient clustering and routing algorithms for wireless sensor networks: Particle swarm optimization approach, Eng. Appl. Artif. Intell. 33 (2014) 127–140, http://dx.doi.org/10.1016/j.engappai.2014.04.009.

[14] P. Akbary, M. Ghiasi, M.R.R. Pourkheranjani, H. Alipour, N. Ghadimi, Extracting appropriate nodal marginal prices for all types of committed reserve, Comput. Econ. 53 (2019) 1–26, http://dx.doi.org/10.1007/s10614-017-9716-2.

[15] 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.

[16] 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.

[17] 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.

[18] 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.

[19] A. Askarzadeh, A novel metaheuristic method for solving constrained engineering optimization problems: Crow search algorithm, Comput. Struct. 169 (2016) 1–12, http://dx.doi.org/10.1016/j.compstruc.2016.03.001.

[20] S. Rahnamayan, H.R. Tizhoosh, M.M.A. Salama, Opposition-based differential evolution, IEEE Trans. Evol. Comput. 12 (2008) 64–79, http://dx.doi.org/10.1109/TEVC.2007.894200.

[21] X.-S. Yang, S. Deb, Cuckoo search: recent advances and applications, Neural Comput. Appl. 24 (2014) 169–174, http://dx.doi.org/10.1007/s00521-013-1367-1.

[22] 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.

[23] H. Garg, A hybrid GSA-GA algorithm for constrained optimization problems, Inform. Sci. 478 (2019) 499–523, http://dx.doi.org/10.1016/j.ins.2018.11.041.

[24] 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.

[25] N. Lynn, P.N. Suganthan, Ensemble particle swarm optimizer, Appl. Soft Comput. 55 (2017) 533–548, http://dx.doi.org/10.1016/j.asoc.2017.02.007.

[26] 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.

[27] 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.

[28] Y. Zhang, Z. Jin, Group teaching optimization algorithm: A novel metaheuristic method for solving global optimization problems, Expert Syst. Appl. 148 (2020) 113246, http://dx.doi.org/10.1016/j.eswa.2020.113246.

[29] M. Tian, X. Gao, An improved differential evolution with information intercrossing and sharing mechanism for numerical optimization, Swarm Evol. Comput. 50 (2019) 100341, http://dx.doi.org/10.1016/j.swevo.2017.12.010.

[30] 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.

[31] A. Sadollah, H. Sayyaadi, A. Yadav, Appl. Soft Comput. 71 (2018) 747–782, http://dx.doi.org/10.1016/j.asoc.2018.07.039.

[32] Ran Cheng, Yaochu Jin, A competitive swarm optimizer for large scale optimization, IEEE Trans. Cybern. 45 (2015) 191–204, http://dx.doi.org/10.1109/TCYB.2014.2322602.

[33] 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.

[34] K. Chen, F. Zhou, A. Liu, Chaotic dynamic weight particle swarm optimization for numerical function optimization, Knowl.-Based Syst. 139 (2018) 23–40, http://dx.doi.org/10.1016/j.knosys.2017.10.011.

[35] D. Oliva, M. Abd El Aziz, A. Ella Hassanien, Parameter estimation of photovoltaic cells using an improved chaotic whale optimization algorithm, Appl. Energy. 200 (2017) 141–154, http://dx.doi.org/10.1016/j.apenergy.2017.05.029.

[36] J. Jiang, X. Yang, X. Meng, K. Li, Enhance chaotic gravitational search algorithm (CGSA) by balance adjustment mechanism and sine randomness function for continuous optimization problems, Phys. Stat. Mech. Appl. 537 (2020) 122621, http://dx.doi.org/10.1016/j.physa.2019.122621.

[37] L. Huang, S. Ding, S. Yu, J. Wang, K. Lu, Chaos-enhanced Cuckoo search optimization algorithms for global optimization, Appl. Math. Model. 40 (2016) 3860–3875, http://dx.doi.org/10.1016/j.apm.2015.10.052.

[38] J. Oliveira, P.M. Oliveira, J. Boaventura-Cunha, T. Pinho, Chaos-based grey wolf optimizer for higher order sliding mode position control of a robotic manipulator, Nonlinear Dynam. 90 (2017) 1353–1362, http://dx.doi.org/10.1007/s11071-017-3731-7.

[39] A. Kaveh, S.M. Javadi, Chaos-based firefly algorithms for optimization of cyclically large-size braced steel domes with multiple frequency constraints, Comput. Struct. 214 (2019) 28–39, http://dx.doi.org/10.1016/j.compstruc.2019.01.006.

[40] T.-S. Du, X.-T. Ke, J.-G. Liao, Y.-J. Shen, DSLC-FOA : Improved fruit fly optimization algorithm for application to structural engineering design optimization problems, Appl. Math. Model. 55 (2018) 314–339, http://dx.doi.org/10.1016/j.apm.2017.08.013.

[41] S. Shao, Y. Peng, C. He, Y. Du, Efficient path planning for UAV formation via comprehensively improved particle swarm optimization, ISA Trans. 97 (2020) 415–430, http://dx.doi.org/10.1016/j.isatra.2019.08.018.

[42] M.W. Ouertani, G. Manita, O. Korbaa, Chaotic lightning search algorithm, Soft Comput. 25 (2021) 2039–2055, http://dx.doi.org/10.1007/s00500-020-05273-0.

[43] J. Liang, B. Qu, P. Suganthan, Problem Definitions and Evaluation Criteria for the CEC 2014 Special Session and Competition on Single Objective Real-Parameter Numerical Optimization, Tech. Rep., Comput. Intell. Lab. Zhengzhou Univ. Zhengzhou China Nanyang Technol. Univ. Singap., 2013, 635.

[44] T. Le-Duc, Q.-H. Nguyen, H. Nguyen-Xuan, Balancing composite motion optimization, Inform. Sci. 520 (2020) 250–270, http://dx.doi.org/10.1016/j.ins.2020.02.013.

[45] M. Jain, V. Singh, A. Rani, A novel nature-inspired algorithm for optimization: Squirrel search algorithm, Swarm Evol. Comput. 44 (2019) 148–175, http://dx.doi.org/10.1016/j.swevo.2018.02.013.

[46] S. Gupta, K. Deep, Improved sine cosine algorithm with crossover scheme for global optimization, Knowl.-Based Syst. (2018) http://dx.doi.org/10.1016/j.knosys.2018.12.008.

[47] S. Mirjalili, S.M. Mirjalili, A. Hatamlou, Multi-verse optimizer: a nature-inspired algorithm for global optimization, Neural Comput. Appl. 27 (2016) 495–513, http://dx.doi.org/10.1007/s00521-015-1870-7.

[48] G. Dhiman, V. Kumar, Seagull optimization algorithm: Theory and its applications for large-scale industrial engineering problems, Knowl.-Based Syst. 165 (2019) 169–196, http://dx.doi.org/10.1016/j.knosys.2018.11.024.

[49] M. Khishe, M.R. Mosavi, Chimp optimization algorithm, Expert Syst. Appl. 149 (2020) 113338, http://dx.doi.org/10.1016/j.eswa.2020.113338.

[50] G. Dhiman, A. Kaur, STOA: A bio-inspired based optimization algorithm for industrial engineering problems, Eng. Appl. Artif. Intell. 82 (2019) 148–174, http://dx.doi.org/10.1016/j.engappai.2019.03.021.

[51] S. Kaur, L.K. Awasthi, A.L. Sangal, G. Dhiman, Tunicate Swarm Algorithm: A new bio-inspired based metaheuristic paradigm for global optimization, Eng. Appl. Artif. Intell. 90 (2020) 103541, http://dx.doi.org/10.1016/j.engappai.2020.103541.

[52] S. Mirjalili, Moth-flame optimization algorithm: A novel nature-inspired heuristic paradigm, Knowl.-Based Syst. 89 (2015) 228–249, http://dx.doi.org/10.1016/j.knosys.2015.07.006.

[53] 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.

[54] A.H. Gandomi, X.-S. Yang, A.H. Alavi, Mixed variable structural optimization using Firefly Algorithm, Comput. Struct. 89 (2011) 2325–2336, http://dx.doi.org/10.1016/j.compstruc.2011.08.002.

[55] S. Gupta, R. Tiwari, S.B. Nair, Multi-objective design optimisation of rolling bearings using genetic algorithms, Mech. Mach. Theory 42 (2007) 1418–1443, http://dx.doi.org/10.1016/j.mechmachtheory.2006.10.002.

[56] A.R. Yildiz, H. Abderazek, S. Mirjalili, A comparative study of recent non-traditional methods for mechanical design optimization, Arch. Comput. Methods Eng. 27 (2020) 1031–1048, http://dx.doi.org/10.1007/s11831-019-09343-x.

[57] G. Dhiman, V. Kumar, Emperor penguin optimizer: A bio-inspired algorithm for engineering problems, Knowl.-Based Syst. 159 (2018) 20–50, http://dx.doi.org/10.1016/j.knosys.2018.06.001.

[58] A.A. Ewees, M.A. Elaziz, Performance analysis of Chaotic Multi-Verse Harris Hawks Optimization: A case study on solving engineering problems, Eng. Appl. Artif. Intell. 88 (2020) 103370, http://dx.doi.org/10.1016/j.engappai.2019.103370.

[59] G. Dhiman, V. Kumar, Spotted hyena optimizer: A novel bio-inspired based metaheuristic technique for engineering applications, Adv. Eng. Softw. 114 (2017) 48–70, http://dx.doi.org/10.1016/j.advengsoft.2017.05.014.

[60] A.A. Heidari, S. Mirjalili, H. Faris, I. Aljarah, M. Mafarja, H. Chen, Harris hawks optimization: Algorithm and applications, Future Gener. Comput. Syst. 97 (2019) 849–872, http://dx.doi.org/10.1016/j.future.2019.02.028.

[61] L. Abualigah, A. Diabat, S. Mirjalili, M. Abd Elaziz, A.H. Gandomi, The arithmetic optimization algorithm, Comput. Methods Appl. Mech. Engrg. 376 (2021) 113609, http://dx.doi.org/10.1016/j.cma.2020.113609.

[62] 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.

[63] F.A. Hashim, E.H. Houssein, M.S. Mabrouk, W. Al-Atabany, S. Mirjalili, Henry gas solubility optimization: A novel physics-based algorithm, Future Gener. Comput. Syst. 101 (2019) 646–667, http://dx.doi.org/10.1016/j.future.2019.07.015.

[64] 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.

[65] 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.

[66] S.H. Samareh Moosavi, V.K. Bardsiri, Poor and rich optimization algorithm: A new human-based and multi populations algorithm, Eng. Appl. Artif. Intell. 86 (2019) 165–181, http://dx.doi.org/10.1016/j.engappai.2019.08.025.

[67] S. Gupta, K. Deep, H. Moayedi, L.K. Foong, A. Assad, Sine Cosine grey wolf optimizer to solve engineering design problems, Eng. Comput. (2020) http://dx.doi.org/10.1007/s00366-020-00996-y.