Complex & Intelligent Systems (2023) 9:6593–6609
https://doi.org/10.1007/s40747-023-01074-8

ORIGINAL ARTICLE

Multi-population cooperative teaching–learning-based optimization for nonlinear equation systems

Liao Zuowen[1,4] · Li Shuijia[2] · Gong Wenyin[2] · Gu Qiong[3]

Received: 12 October 2022 / Accepted: 10 April 2023 / Published online: 25 May 2023

Abstract
Solving nonlinear equation systems (NESs) requires locating different roots in one run. To effectively deal with NESs, a multi-population cooperative teaching–learning-based optimization, named MCTLBO, is presented. The innovations of MCTLBO are as follows: (i) two niching technique (crowding and improved speciation) are integrated into the algorithm to enhance population diversity; (ii) an adaptive selection scheme is proposed to select the learning rules in the teaching phase; (iii) the new learning rules based on experience learning are developed to promote the search efficiency in the teaching and learning phases. MCTLBO was tested on 30 classical problems and the experimental results show that MCTLBO has better root finding performance than other algorithms. In addition, MCTLBO achieves competitive results in eighteen new test sets.

Keywords Nonlinear equation systems · multi-population cooperation · teaching–learning-based optimization · niching technique · adaptive selection scheme

Introduction

Many real-world problems, such as those found in robotics [1], automatic control [2, 3] and fuzzy systems [4], can be reduced to nonlinear equation systems (NESs). In practical application scenarios, NESs contain several different optimal solutions. Solving NESs requires finding different roots in one run, which can offer decision makers with various in different environments [5]. However, it is challenging work.

The conventional approach for NESs is to use numerical methods, such as the Newton method [6] and homotopy continuation [7]. Although numerical methods have several advantages, they also have several disadvantages. For instance, they are sensitive to the objective function and can only find one root at a time. Therefore, researchers have sought new ideas to replace numerical methods to deal with NESs.

Evolutionary algorithms (EAs) [8] which use natural evolution principles, are attract lots of researchers attention. Since it is insensitive to the objective function and the initial guess. Therefore, competitive results have been obtained in many optimization problems, such as in energy system design [9, 10], image registration [11], many-objective optimization [12], multi-modal optimization [13], power systems [14, 15], etc. Several methods for NESs have been proposed to locate the roots. Gong et al. proposed RADE, which combined repulsive technique, crowding method, and parameter adaptation to solve NESs [16]. However, RADE uses a fixed repulsion radius and limits its search efficiency. Hence, Liao et al. designed a dynamic repulsion-based EA (DREA) [17]. The authors designed four different dynamic variation of the repulsive radius [17]. He et al. employed fuzzy neighborhood to improve population diversity, and an orientation-based

✉ Gu Qiong
qionggu@hbuas.edu.cn

Liao Zuowen
liaozuowen@bbgu.edu.cn

Li Shuijia
shuijiali@cug.edu.cn

Gong Wenyin
wygong@cug.edu.cn

1 Beibu Gulf Ocean Development Research Center, Beibu Gulf University, Qinzhou 535000, China

2 School of Computer Science, China University of Geosciences, Wuhan 430074, China

3 School of Computer Engineering, Hubei University of Arts and Science, Xiangyang 441053, China

4 Key Laboratory of Beibu Gulf Offshore Engineering Equipment And Technology (Beibu Gulf University), Education Department of Guangxi Zhuang Autonomous Region, Qinzhou 535011, China

mutation was proposed to generate trial vectors [18]. A memetic niching-based EAs was designed in [19] to find different roots. Liao et al. [20] applied the decomposition technique to effectively divide the population and proposed subpopulation mutation strategies to deal with NESs. Wu et al. [21] present a k-means clustering-based (KSDE) for multiple roots location. Wang et al. [22], two-archive techniques was designed to exploit inferior and elite individual to guide the evolution. They also combined niching methods and differential evolution to solve NESs. The author proposed AGSDE to determine the roots of NESs [23]. In AGSDE, archives save useful historical individuals and use that information to guide the evolution of algorithms. Other researcher have also adopted multi-objective techniques to deal with NESs. Song et al. [24], MONES transformed NESs into a bi-objective optimization problem and applied NSGA-II to solve it, which obtained promising results. To compensate the curse of dimensionality, Gong et al. [25] designed a A-WeB. Gao et al. developed a new multi-objective EA (MOPEA [26]) and a two-phase EA (TPEA [27]) to find different roots. Moreover, Ji et al. extended the existing work and proposed a dynamic tri-objective differential evolution (SaTriMODE), which showed competitive results [28].

Rao et al. [29] designed teaching–learning-based optimization (TLBO) to deal with optimization problems. TLBO simulates the traditional classroom teaching process, which is different from the principle of several other EAs. Because TLBO is easy to implement, efficient and does not require parameter setting, it is widely used in different optimization problems, such as solar parameter identification [30], engineering design [31], and medical disease diagnosis [32], etc. Although the TLBO variant achieves good performance in solving many optimization problems, it still has the following disadvantages: (i) most of the TLBO variants only seek an optimum, and it is difficult to find multiple optimum in a single run; (ii) there are few researches on how to use cooperation among populations in solving multi-root problems. Moreover, there are still few cases using TLBO to solve NESs.

Recently, multi-population cooperative technique is widely used to solve optimization problems [33, 34]. Combine the above mentioned deficiencies, this paper proposes a multi-population cooperative teaching–learning-based optimization, namely MCTLBO to solve NESs. To the best of our knowledge, this is the first case for using TLBO to solve NESs. The results show that the proposed MCTLBO can obtain promising success rate and root ratio.

The main novelties of this paper are shown below:

- Two niching techniques, (ie. crowding and an improved speciation), are integrated into TLBO to guide the algorithm to search the more promising region, enhancing population diversity.
- A fitness-ranking-based adaptive selection scheme is proposed to select the learning rules in the teaching phase.
- Learning rules based on experience learning are developed to promote search efficiency, improving the search efficiency of MCTLBO.
- The performance of MCTLBO is evaluated by solving NESs with different features.

The rest of the paper is organized as follows. Section Background introduces the NES objective transformation technique, TLBO, DE and two niching-based DE. Sect. Our approach describes MCTLBO. In Sects. Experimental studies and Study on the new test set, the experimental results are demonstrated. Finally, Conclusion summarizes this paper.

Background

NES formulation

Many practical problems can be transformed into nonlinear equations. The mathematical expressions of NESs are generally as follows:

$$F(\mathbf{x}) = \begin{cases} F_1(\mathbf{x}) = 0 \\ \quad\cdots \\ F_i(\mathbf{x}) = 0 \\ \quad\cdots \\ F_m(\mathbf{x}) = 0 \end{cases} \tag{1}$$

where m is the number of equations. $\mathbf{x} = [x_1, \cdots, x_D] \in \mathbf{\Omega}$ denotes the decision variables, where D and $\mathbf{\Omega}$ are the dimension and decision space. In general, $x_j \in [L(j), U(j)]$, where $L(j), U(j)$ represents the lower and upper boundaries of $\mathbf{x}$ in the j-th dimension, respectively. Eq. (1) usually has multiple roots that are equally important. They can provide decision-makers with multiple optimal options under different scenarios. Therefore, it makes sense to get more than one root at a time.

Before an EA can be applied to solve an NES, the NES must be transformed into an optimization problem. We transform the NES into a single objective problem. The transformed expression is as follows:

$$\min\ f(\mathbf{x}) = \sum_{i=1}^{m} F_i^2(\mathbf{x}), \tag{2}$$

TLBO

In TLBO, individuals constantly search for optimal solutions through the teaching and learning phases. The specific teaching–learning process is described as follows.

Teaching phase

In general, n students can make up a class. Each learner $\mathbf{x}_i$ is regarded as a potential optima, and the learner with the minimum fitness is the teacher. TLBO needs to improve the average grade of the whole class after the teacher phase. Teachers can impart their experience or knowledge to learners, and learners can improve their performance by absorbing useful knowledge. The formula of the teaching phase is:

$$\mathbf{x}_{i,new} = \mathbf{x} + r \cdot (\mathbf{x}_{teacher} - T_F \cdot \mathbf{x}_{mean}), \tag{3}$$

where $\mathbf{x}_{i,new}$ is the new individual produced after learning from the teacher; $r \in [0, 1]$; T_F is the teaching factor; and $T_F \in [1, 2]$; and $\mathbf{x}_{mean}$ is the mean individual of the class expressed as:

$$\mathbf{x}_{mean} = \frac{1}{N}\sum_{i=1}^{N} \mathbf{x}_i, \tag{4}$$

Learning phase

In the learning phase, students can learn from their classmates who are better than them to improve their knowledge level. The formula can be stated as follows:

$$\mathbf{x}_{i,new} = \begin{cases} \mathbf{x}_i + r \cdot (\mathbf{x}_i - \mathbf{x}_j), \text{ if } \mathbf{x}_i \text{ is better than } \mathbf{x}_j \\ \mathbf{x}_i + r \cdot (\mathbf{x}_j - \mathbf{x}_i), \quad otherwise \end{cases} \tag{5}$$

DE

DE [35] contains four operations: initialization, mutation, crossover, and selection.

Initialization: The initialization operator can randomly generate different distinct individuals in the decision space:

$$x_{i,j} = L(j) + rand * (U(j) - L(j)), \tag{6}$$

where $i = 1, \cdots, NP$, and NP represents the population size; $j \in [1, \ldots, D]$, D is the dimension of x_i; $L(j), U(j)$ represent lower and upper limits in different dimensions, respectively.

Mutation: The mutation operator is used to generate the mutation vector. “DE/rand/1” is the classical mutation strategy and is expressed as follows:

$$\mathbf{v}_i = \mathbf{x}_{r_1} + F \cdot (\mathbf{x}_{r_2} - \mathbf{x}_{r_3}), \tag{7}$$

where F is the scaling factor to control the magnitude of the difference vector $(\mathbf{x}_{r_2} - \mathbf{x}_{r_3})$. r_1, r_2, r_3 are three random integers between 1 and NP, and $r_1 \neq r_2 \neq r_3 \neq i$.

Crossover: The crossover operator is utilized to produce the trial vector. In this paper, the binary crossover is used and shown as follows:

$$u_{i,j} = \begin{cases} v_{i,j}, \text{ if } rand \leq CR \mid\mid j == j_d \\ x_{i,j}, \quad \text{otherwise} \end{cases} \tag{8}$$

where CR is the crossover rate, which controls how many components are inherited from the mutant vector, and $j_d \in [1, D]$ and it makes sure at least one component of trial vector is inherited from the mutant vector.

Selection: The selection operator ensures that the superior individual will enter the next evolution. The formula is shown in Eq. (9):

$$\mathbf{x}_i = \begin{cases} \mathbf{u}_i, \text{ if } f(\mathbf{u}_i) \leq f(\mathbf{x}_i) \\ \mathbf{x}_i, \quad \text{otherwise} \end{cases} \tag{9}$$

Two niching techniques

Qu et al. [13] developed DE with neighborhood-based mutation, named NCDE and NSDE, to deal with multimodal problems and obtained promising results. In NCDE and NSDE, crowding and speciation techniques are applied to preserve population diversity. Generally, the crowding technique forms a cluster near each individual, and the DE operators are completed in the cluster. The speciation technique divides the whole population into multiple sub-populations, and the DE operations are carried out among each sub-population. The pseudo-code of NCDE and NSDE are provided in Algorithm 1 and 2, where $\mathbb{OP}_i$ represents the sub-population, $\mathbb{P}$ is the whole population; and num is the neighborhood size and n is the number of sub-populations.

Algorithm 1 Process of NCDE

```
1: for $i = 1 : NP$ do
2:     Choose $num$ individuals closest to $\mathbf{x}_i$ to construct the subpopulation $\mathbb{OP}_i$
3:     Generate the trial vector $\mathbf{u}_i$ within $\mathbb{OP}_i$ via Eqs. (7) and (8)
4:     Select the most similar individual $\mathbf{x}_s$ to $\mathbf{u}_i$ from $\mathbb{P}$
5:     Compare the fitness between $\mathbf{x}_s$ and $\mathbf{u}_i$
6:     Choose the better one for the next evolution
7: end for
```

Our approach

Motivations

Building on the promising performance of TLBO, we employ it to deal with NESs. However, using TLBO directly to solve NESs has several drawbacks: i) TLBO can only find one

Algorithm 2 Process of NSDE

```
1: Sort the population $\mathbb{P}$ according to the fitness in ascending order
2: for $i = 1 : n$ do
3:     Select $n$ most similar individuals to $\mathbf{x}_i$ from $\mathbb{P}$ to form $\mathbb{OP}_i$
4:     Delete $\mathbf{x}_i$ from $\mathbb{P}$
5:     for $j = 1$ to $\|\mathbb{OP}_i\|$ do
6:         Generate the trial vector $\mathbf{u}_i$ within $\mathbb{OP}_i$
7:         Select the most similar individual $\mathbf{x}_s$ to $\mathbf{u}_i$ from $\mathbb{OP}_i$
8:         Compare the fitness between $\mathbf{x}_s$ and $\mathbf{u}_i$
9:         Choose the better one for the next evolution
10:    end for
11: end for
12: Incorporate all sub-populations $\mathbb{OP}_i$ to form $\mathbb{P}$
```

optima and cannot find different roots simultaneously. Thus, population diversity must be enhanced during the run; ii) In the teaching and learning phase of TLBO, the original learning rules limit the performance of the algorithm, which leads to a reduction in search efficiency; iii)

Motivated by this cue, we develop MCTLBO, where dual niching techniques, fitness-ranking-based adaptive strategy selection, and learning rules are presented to solve NESs, which are introduced in the following subsection.

Niching-based TLBO

As described in Sect. Motivation, TLBO must contain the population's diversity during the run. The niching technique helps find different promising areas by modifying the search characteristics of the algorithm. Hence, the crowding and speciation techniques are used in MCTLBO so that the learner can fully explore the search space.

Crowding in teaching phase

The crowding technique can form a subpopulation in the vicinity of each individual. This makes it easy for the algorithm to search in every small search region, which enables learners to use the neighborhood information. Algorithm 1 describes the detailed process of the technique. Notably, to improve the search performance of learning rules in the teaching phase, this paper will improve it. The details will be elaborated Sect. Learning in teaching phase.

Speciation in learning phase

The speciation technique can partition the population into multiple sub-populations and the algorithm searches the region of each sub-population. However, the original speciation technique may have a defect in determining seeds. In this section, we will improve it and the specific process is shown in Algorithm 3, where $\mathbb{A}$ is an external archive that saves the found root; size($\mathbb{A}$) is the size of $\mathbb{A}$; dis is the distance threshold given in advance; NFE is the number of fitness evaluations.

Algorithm 3 Process of the improved speciation technique

```
1: Sort the population $\mathbb{P}$ according to the fitness in ascending order
2: for $i = 1 : n$ do
3:     if size($\mathbb{A}$) $\neq 0$ then
4:         Calculate the minimum Euclidean distance $d_i$ between $\mathbf{x}_i$ and
   the roots in $\mathbb{A}$
5:         if $d_i < dis$ then
6:             Randomly re-initialize $\mathbf{x}_i$
7:             Evaluate the fitness $\mathbf{x}_i$
8:             $NFE = NFE + 1$
9:         end if
10:    end if
11: end for
12: Assign different individuals to each seed and form the sub-
   population $\mathbb{OP}_i$.
```

The main difference from the original speciation technique lies in the addition of the determination condition when determining the seed. In lines 3–9, if the algorithm does not find the root, the current individual can be considered as a seed. Otherwise, the distance between $\mathbf{x}_i$ and the individual in $\mathbb{A}$ is judged. If it is too close, $\mathbf{x}_i$ is initialized and its fitness value is reevaluated. The reasons for re-initialization are as follows: 1) If the seed is close to the found root, the algorithm will have a high probability of finding the same root in the subsequent search, which leads to a waste of computing resources; 2) other individuals have the opportunity to serve as seeds so the algorithm can explore other promising areas and increase the probability of finding a new root.

Multi-population cooperation mechanism

In Sect. Speciation in learning phase, the population will be divided into subpopulations. In this section, a multi-population cooperation mechanism is proposed to further improve the diversity of algorithms. The detailed process is as follows:

Step 1: Following the method in the previous section, subpopulations are arranged topologically so that each subpopulation can only transmit information to a specific subpopulation. Similarly, it can only receive information from certain subpopulations.

Step 2: The transfer of individual information is carried out by means of intergenerational communication. In other words, if $\frac{iter}{S} = 0$, then multi-population cooperation mechanism is triggered. $iter$ is the current iteration number, S is the interval generation.

Step 3: The algorithm selects the worst individual from the migrating population and replaces an individual randomly selected from the migrating subpopulation. Information

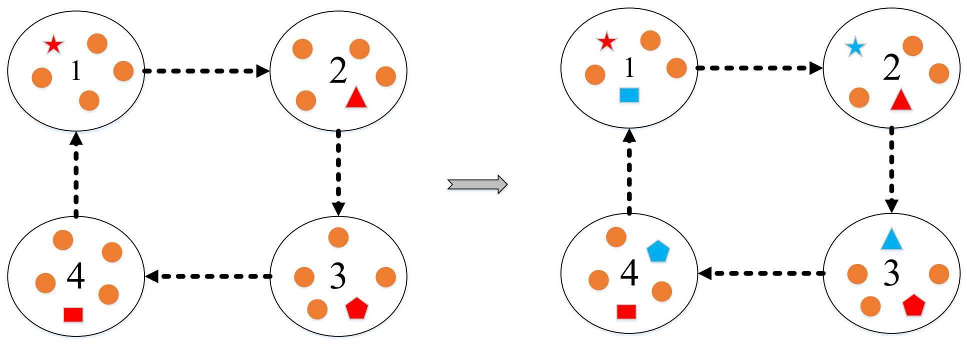


Fig. 1 Multi-population cooperative processes

exchange among subpopulations is realized through migration.

Figure 1 shows the process of multi-species cooperation. The graph on the left shows that there are four subpopulations, with the red individuals representing the worst individuals in each subpopulation. In the figure on the right, the blue individuals represent migrating individuals acquired from adjacent subpopulations. The cooperation among subpopulations can be realized by transferring individuals to improve the diversity of the algorithm.

Updating rules of learners

Learning in teaching phase

In the proposed method, the learner adopts a fitness-ranking-based adaptive selection scheme to choose learning rules in the teaching phase. First, a probability p_i is generated for each learner. Here, the value of p_i is between 0 and 1. If $rand < p_i$, an improved TLBO learning rule is applied by the learner; otherwise, the "DE/rand/1" in DE is applied. Algorithm 4 show the learning rule in the teaching phase. In line 5, p_i is calculated as follows:

$$p_i = \frac{R(i)}{num}, \tag{10}$$

where num is the neighborhood size and $R(i)$ is calculated as follows:

$$R(i) = num - RP(i), \tag{11}$$

where $RP(i)$ is the ranking of fitness of $\mathbf{x}_i$ in $\mathbb{OP}_i$. In line 7, if $rand < p_i$, an experience-learning-based learning rule is proposed to generate offspring:

$$\begin{aligned}\mathbf{x}_{i,new} = \mathbf{x}_i + rand \cdot (\mathbf{x}_{teacher,i} - T_F \cdot \mathbf{x}_{mean,i})\\ + rand \cdot (\mathbf{x}_{r1} - \mathbf{x}_{r2}),\end{aligned} \tag{12}$$

otherwise, in line 9, "DE/rand/1" is adopted:

$$\mathbf{x}_{i,new} = \mathbf{x}_i + F \cdot (\mathbf{x}_{r1} - \mathbf{x}_{r2}), \tag{13}$$

where $\mathbf{x}_{teacher,i}$ is the best individual in $\mathbb{OP}_i$; $\mathbf{x}_{mean,i}$ is the mean vector in $\mathbb{OP}_i$; $r1$ and $r2$ are randomly selected from $[1, num]$; $r1 \neq r2$.

The reasons for adopting improved learning strategies are as follows:

- On the basis of Eqs. (3), (12) adds vector disturbance. If p_i is large, it means that the individual has the highest ranking of fitness values in the current sub-population. The learners can absorb the experience of teachers and learn other information at the same time.
- If p_i is small, Eq. (13) can balance the population diversity and convergence.

Algorithm 4 Learning strategy in teaching phase

```
1: for $i = 1 : NP$ do
2:    Find $num$ individuals in the population that are closest to $\mathbf{x}_i$ and
      form a cluster $\mathbb{OP}_i$
3:    Sort $\mathbb{OP}_i$ according to the fitness in ascending order
4:    Determine the ranking of fitness of $\mathbf{x}_i$ in $\mathbb{OP}_i$
5:    Adopt Eq. (10) to calculate the probability $p_i$
6:    if $rand < p_i$ then
7:       Employ Eq. (12) to generate offspring
8:    else
9:       Adopt Eq. (13) to generate offspring
10:   end if
11: end for
```

Learning in learner phase

In Eq. (5), due to the loss of diversity, TLBO is prone to stagnation. To remedy this, an experience-learning-based learning rule is adopted in this paper:

$$\mathbf{x}_{i,new} = \mathbf{x}_i + rand \cdot (\mathbf{x}_{mean} - \mathbf{x}_i) + rand \cdot (\mathbf{x}_{r1} - \mathbf{x}_{r2}), \tag{14}$$

Compared with Eqs. (5), (14) adds the part of $\mathbf{x}_{mean}$ and experience learning($\mathbf{x}_{r1} - \mathbf{x}_{r2}$). First, $\mathbf{x}_{mean}$ can improve the exploration performance of learners. Second, experiential learning increases the search disturbance.

Implementation of MCTLBO

By integrating two improved niching techniques, fitness-ranking-based adaptive selection scheme and the experience-learning-based learning rules, we propose the MCTLBO framework. Algorithm 5 shows the steps of MCTLBO, where NFE is the number of fitness evaluations, and Max_NFE is the maximum NFE allowed. From Algorithm 5, in lines 4–6, the crowding technique and the enhanced learning rules are used in teaching phase. In lines 7–10, the improved speciation technique and the experience-learning-based learning strategy are utilized in learning phase. Lines 11–17, it determined if there are individuals satisfying the root condition in the population. If so, they are re-initialized to ensure the population diversity. ϵ is a threshold that determines whether the accuracy of $\mathbf{x}_i$ meets the condition.

Algorithm 5 MCTLBO

```
Require: Control parameters: $NP$, $Max\_NFE$
Ensure: $\mathbb{A}$
1: Set $NFE = 1$, $num = 8$, $n = 20$, $iter = 1$
2: Randomly initialize the population $\mathbb{P}$ and calculate their fitness
3: while $NFE < Max\_NFE$ do
4:     Adopt Algorithm 4 to learn in the teaching phase
5:     Update $NFE$
6:     Compare the fitness between parents and offspring and retain the better individual
7:     Apply Algorithm 3 to form multiple sub-populations
8:     Use multi-population cooperation mechanism to search
9:     Employ Eq. (14) to learn in the learning phase
10:    Update $NFE$
11:    Compare the fitness between parents and offspring
12:    for $i = 1 : NP$ do
13:        Determine whether the current individual $\mathbf{x}_i$ is the root
14:        if $\mathbf{x}_i \le \epsilon$ then
15:            Obtain the minimum Euclidean distance ($min\_dis$) between $\mathbf{x}_i$ and all elements of $\mathbb{A}$
16:            if $min\_dis \ge \delta$ then
17:                Save $\mathbf{x}_i$ into $\mathbb{A}$
18:                Re-initialize $\mathbf{x}_i$ and evaluate its fitness
19:            end if
20:            Update $NFE$
21:        end if
22:    end for
       $iter = iter + 1$
23: end while
```

Complexity of MCTLBO

The complexity of MCTLBO is as follows:

- On the teaching phase, the complexity is $\mathcal{O}(iter * NP)$, where $iter$ is the number of iterations and NP is the population size.
- On the learning phase, the complexity is $\mathcal{O}(iter * NP * log(NP))$.
- On the multi-population cooperation, the complexity is $\mathcal{O}(\frac{iter}{S} * NP)$.
- On the archive updating, the complexity is $\mathcal{O}(iter * NP * \mathcal{S}_{\mathbb{A}})$, where $\mathcal{S}_{\mathbb{A}}$ is the archive size of $\mathbb{A}$.

Therefore, the complexity of MCTLBO is $\mathcal{O}(iter * NP * log(NP))$.

Experimental studies

Test suite and evaluation indicators

Thirty classical NESs [22, 27, 36] are used to evaluate the performance of MCTLBO, which has been widely used in many papers. We use two classical evaluation indexes, root ratio and success rate, to compare the performance of different algorithms, which are demonstrated as below.

Root Ratio (RR): RR calculates the probability of finding the root of the algorithm in multiple runs, which can reflect the algorithm's root finding ability. The calculation formula is as follows:

$$RR = \frac{\sum_{i=1}^{N_{\text{run}}} N_{\text{rf},i}}{N_r * N_{\text{run}}}, \tag{15}$$

where N_{run} represents the total number of runs; $N_{\text{rf},i}$ is the number of roots found by the $i-$th run, and N_r is the number of known roots.

Success Rate (SR): SR is the probability of successfully finding all the roots of NESs in multiple runs. SR is defined as defined as

$$SR = \frac{N_{\text{success}}}{N_{\text{run}}} \tag{16}$$

where N_{success} denotes the number of times the algorithm can find all roots in multiple runs.

In addition, Wilcoxon and Friedman test are used to verify the statistical differences between various methods. In the multi-problem Wilcoxon test, R^+ represents the rank sum where the algorithm performs better than its competitors, while R^- is just the opposite.

Compared algorithms

In this section, we select several advanced algorithms classified as follows:

- *Repulsion-based methods*: RADE [16] and DREA [17].
- *Niching-based methods*: HNDE/2A [22], FNODE [18], DDE/R [20], KSDE [21], TPEA [27] and LSTP [37].
- *Multi-objective-based methods*: MONES [24], A-WeB [25] and SaTriMODE [28].

Table 1 Comparison of MCTLBO and other methods on RR

Prob	MCTLBO	RADE	DREA	FNODE	HNDE/2A	DDE/R	KSDE	TPEA	MONES	A-WeB	SaTriMODE	LSTP	C-FOA	C-GWO	C-ELATLBO	C-ITLBO
F01	**1.000**	**1.000**	0.000	**1.000**	**1.000**	0.930	**1.000**	0.920	0.980	0.620	**1.000**	**1.000**	**1.000**	0.000	0.667	0.00
F02	**1.000**	0.990	0.990	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	0.970	**1.000**	**1.000**	0.950	0.939	0.848	**1.000**	**1.00**
F03	**1.000**	0.990	0.950	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	0.940	0.950	**1.000**	0.980	0.911	0.867	0.956	0.87
F04	**1.000**	0.910	**1.000**	0.960	**1.000**	**1.000**	0.980	**1.000**	0.590	**1.000**	**1.000**	0.830	0.128	0.667	0.590	0.69
F05	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	0.990	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.00**
F06	**1.000**	0.990	0.950	0.990	**1.000**	**1.000**	0.980	**1.000**	**1.000**	0.940	0.860	0.960	0.167	0.667	0.500	0.63
F07	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	0.830	**1.000**	**1.000**	0.980	**1.000**	**1.000**	**1.000**	**1.00**
F08	**1.000**	0.990	**1.000**	**1.000**	**1.000**	**1.000**	0.900	**1.000**	0.770	0.830	0.826	0.970	0.667	0.952	0.905	**1.00**
F09	**1.000**	0.970	**1.000**	**1.000**	**1.000**	0.980	0.990	**1.000**	0.000	0.890	**1.000**	0.980	**1.000**	0.667	0.000	0.00
F10	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	0.500	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.00**
F11	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	0.910	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.00**
F12	0.980	0.630	0.870	0.860	0.923	**1.000**	0.990	0.940	0.430	0.880	**1.000**	0.640	0.233	0.533	0.667	0.80
F13	0.872	0.890	0.910	0.730	0.725	**1.000**	**1.000**	0.920	0.540	0.090	0.974	0.520	0.194	0.389	0.278	0.72
F14	**1.000**	0.980	**1.000**	**1.000**	**1.000**	**1.000**	0.990	**1.000**	0.190	0.970	0.987	0.980	0.333	0.963	0.852	0.81
F15	**1.000**	**1.000**	0.500	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	0.980	0.500	**1.000**	**1.000**	**1.00**
F16	**1.000**	0.990	**1.000**	**1.000**	**1.000**	**1.000**	0.930	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	0.667	0.923	**1.000**	**1.00**
F17	**1.000**	0.940	**1.000**	0.980	**1.000**	**1.000**	**1.000**	0.980	0.080	0.660	0.977	0.850	0.771	0.313	0.542	0.19
F18	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	0.900	**1.000**	**1.000**	**1.000**	0.940	**1.000**	**1.000**	0.944	**1.000**	**1.000**	**1.00**
F19	**1.000**	0.790	0.000	0.930	**1.000**	**1.000**	**1.000**	0.720	0.000	0.620	**1.000**	**1.000**	**1.000**	0.000	0.000	0.00
F20	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	0.710	0.950	1.000	0.990	0.952	**1.000**	**1.000**	**1.00**
F21	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	0.990	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.00**
F22	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	0.980	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.00**
F23	0.996	0.560	0.790	0.910	**1.000**	0.940	0.920	0.980	0.140	0.150	0.390	0.540	0.125	0.438	0.292	0.52
F24	**1.000**	0.990	0.910	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	0.730	0.850	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.00**
F25	0.983	0.830	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	0.310	0.230	**1.000**	**1.000**	0.167	0.667	0.500	0.67
F26	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.00
F27	**1.000**	0.990	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	0.778	**1.000**	**1.000**	**1.00**
F28	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	0.000	0.940	**1.000**	**1.000**	**1.000**	0.667	**1.000**	**1.00**
F29	**1.000**	0.990	0.950	0.990	**1.000**	**1.000**	0.500	**1.000**	0.140	0.930	**1.000**	0.960	**1.000**	0.600	0.933	0.73
F30	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.00
Avg	**0.994**	0.947	0.894	0.978	0.988	0.992	0.969	0.982	0.661	0.848	0.967	0.937	0.749	0.772	0.789	0.788

Bold values represent the best results obtained by the algorithms when solving the current NES

Table 2 Comparison of MCTLBO and other methods on SR

Prob	MCTLBO	RADE	DREA	FNODE	HNDE/2A	DDE/R	KSDE	TPEA	MONES	A-WeB	SaTriMODE	LSTP	C-FOA	C-GWO	C-ELATLBO	C-ITLBO
F01	**1.00**	**1.00**	0.00	**1.00**	**1.00**	0.86	**1.00**	0.82	0.96	0.36	**1.00**	**1.00**	**1.00**	0.00	0.33	0.000
F02	**1.00**	0.90	0.93	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	0.76	**1.00**	**1.00**	0.56	0.33	0.00	**1.00**	**1.000**
F03	**1.00**	0.95	0.46	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	0.43	0.58	**1.00**	0.80	0.00	0.00	0.33	0.333
F04	**1.00**	0.31	**1.00**	0.52	**1.00**	**1.00**	0.80	**1.00**	0.00	**1.00**	**1.00**	0.04	0.00	0.00	0.00	0.000
F05	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	0.96	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.000**
F06	**1.00**	0.93	0.66	0.98	**1.00**	**1.00**	0.86	**1.00**	**1.00**	0.60	0.24	0.72	0.00	0.00	0.00	0.000
F07	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	0.66	**1.00**	**1.00**	0.96	**1.00**	**1.00**	**1.00**	**1.000**
F08	**1.00**	0.98	**1.00**	**1.00**	**1.00**	**1.00**	0.40	**1.00**	0.00	0.12	0.14	0.84	0.00	0.67	0.67	**1.000**
F09	**1.00**	0.91	**1.00**	**1.00**	**1.00**	0.93	0.93	**1.00**	0.00	0.68	**1.00**	0.96	**1.00**	0.33	0.00	0.000
F10	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	0.03	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.000**
F11	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	0.43	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.000**
F12	0.90	0.00	0.03	0.28	0.40	**1.00**	0.93	0.4	0.00	0.28	**1.00**	0.00	0.00	0.00	0.00	0.000
F13	0.23	0.19	0.26	0.02	0.00	0.00	**1.00**	0.92	0.36	0.00	0.73	0.00	0.00	0.00	0.00	0.000
F14	**1.00**	0.89	**1.00**	**1.00**	**1.00**	**1.00**	0.93	**1.00**	0.00	0.76	0.88	0.88	0.00	0.67	0.00	0.000
F15	**1.00**	**1.00**	0.00	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	0.96	0.00	**1.00**	**1.00**	**1.000**
F16	**1.00**	0.94	**1.00**	**1.00**	**1.00**	**1.00**	0.86	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	0.00	0.00	**1.00**	**1.000**
F17	**1.00**	0.43	0.03	0.76	**1.00**	**1.00**	**1.00**	0.82	0.00	0.00	0.78	0.00	0.00	0.00	0.00	0.000
F18	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	0.66	**1.00**	**1.00**	0.67	**1.00**	**1.00**	**1.000**
F19	**1.00**	0.69	0.00	0.86	**1.00**	0.80	**1.00**	0.64	0.00	0.24	**1.00**	**1.00**	**1.00**	0.00	0.00	0.000
F20	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	0.03	0.70	**1.00**	0.96	0.67	**1.00**	**1.00**	**1.000**
F21	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	0.98	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.000**
F22	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	0.96	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.000**
F23	0.93	0.00	0.00	0.28	**1.00**	0.33	0.43	0.98	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.000
F24	**1.00**	0.99	0.43	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	0.03	0.14	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.000**
F25	0.97	0.67	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	0.07	0.02	**1.00**	**1.00**	0.00	0.33	0.00	0.333
F26	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.000
F27	**1.00**	0.99	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	0.33	**1.00**	**1.00**	**1.000**
F28	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	0.00	0.88	**1.00**	**1.00**	**1.00**	0.33	**1.00**	**1.000**
F29	**1.00**	0.97	0.76	0.98	**1.00**	**1.00**	0.00	**1.00**	0.00	0.66	**1.00**	0.80	**1.00**	0.00	0.67	0.333
F30	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.000
Avg	**0.97**	0.82	0.72	0.89	0.95	0.93	0.88	0.93	0.47	0.66	0.89	0.78	0.53	0.51	0.60	0.60

Bold values represent the best results obtained by the algorithms when solving the current NES

Table 3 Average ranks of MCTLBO and other comparison algorithms obtained by the Friedman test on RR and SR

Algorithm	Ranking (RR)	Ranking (SR)
MCTLBO	5.6500	**5.6667**
RADE	8.4000	8.5167
DREA	7.6833	8.0333
FNODE	6.5167	6.4333
HNDE/2A	**5.6167**	5.7500
DDE/R	5.9333	6.1667
KSDE	7.2667	7.2167
TPEA	5.9833	5.8167
MONES	11.3333	10.8833
A-WeB	9.9333	9.8500
SaTriMODE	6.4833	6.4167
LSTP	8.5363	8.7497
C-FOA	10.4500	10.2000
C-GWO	9.8833	10.0667
C-ELATLBO	9.8333	9.9500
C-ITLBO	9.1667	9.3500

Bold values represent the best ranking of the algorithm compared with other algorithms

In addition, we selected other optimization algorithms such as FOA [38], GWO [39], ELATLBO [30] and ITLBO [40] for comparison. FOA and GWO are popular swarm intelligence algorithms while ELATLBO and ITLBO are variations of TLBO. Crowding techniques were added to each algorithm to improve their performance when solving equations, since these methods could only find an optimal solution at a time. Therefore, the algorithms to be compared can be called C-FOA, C-GWO, C-ELATLBO, and C-ITLBO.

The parameter of the comparison algorithms remain the same as in their original papers.. The parameters of MCTLBO are set as follows:

- Population size $NP = 100$; Neighborhood size $num = 8$; Number of the sub-population $n = 20$;Distance threshold $dis = 0.01$.
- The criteria for the root: $\epsilon = 10^{-6}$ when $D \leq 5$; while $\epsilon = 10^{-4}$ for $D > 5$.

Experimental results

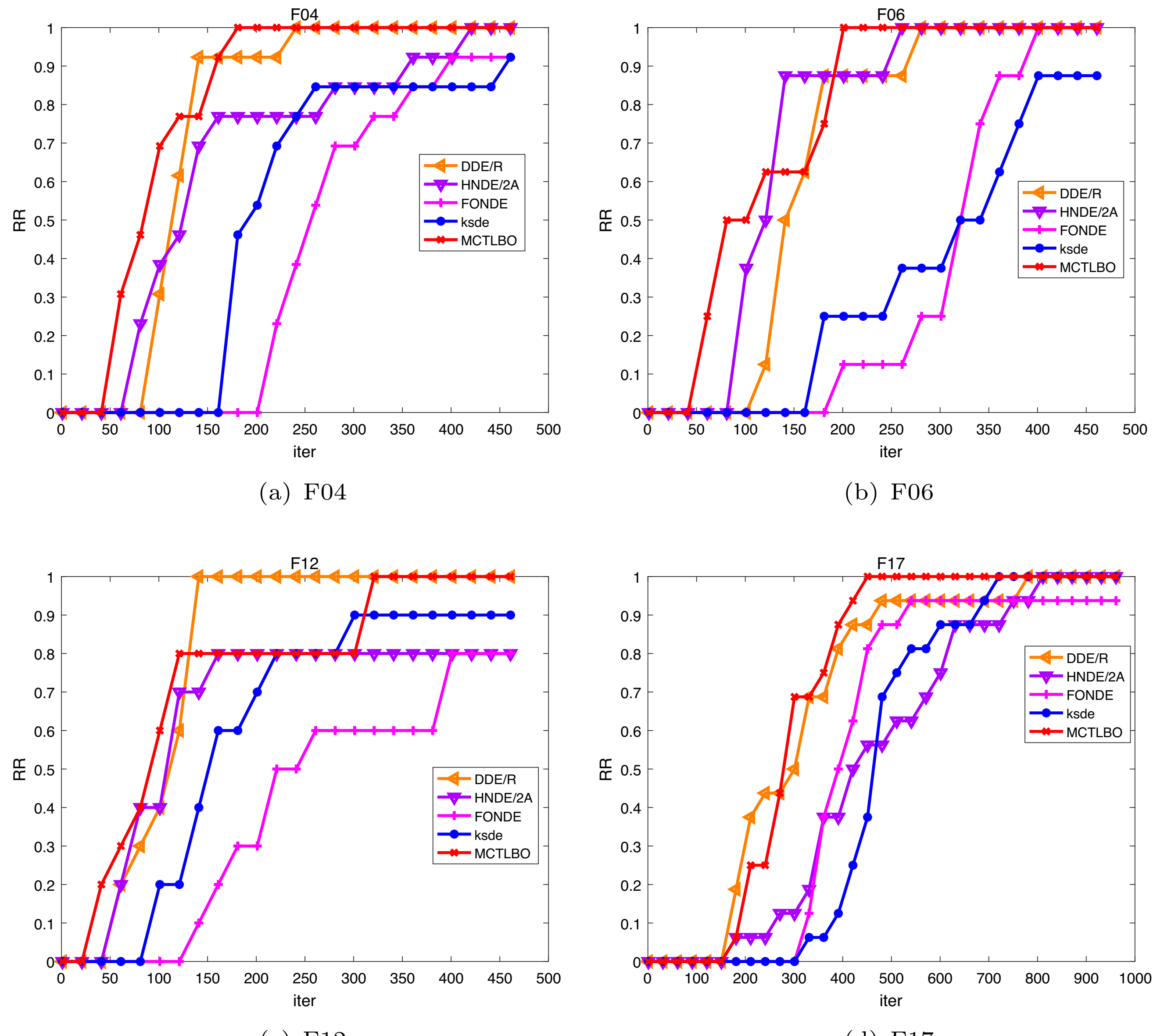
The RR and SR obtained by MCTLBO and its competitors are shown in Tables 1 and 2. It is clear that MCTLBO obtained the highest RR and SR in solving 30 NESs ($RR = 0.99$, $SR = 0.97$), followed by HNDE/2A, TPEA, and DDE/R. Moreover, MCTLBO can locate all roots in 26 out of 30 NESs while HNDE/2A can successfully find all roots in 28 out of 30 NESs. Nevertheless, MCTLBO achieved better overall performance than HNDE/2A.

To further demonstrate the superiority of MCTLBO, the Friedman test and the Wilcoxon test results are given in Tables 3 and 4. From Table 3, it can be observed HNDE/2A achieves the smallest ranking of RR (5.6167) and MCTLBO obtains the best ranking of SR (5.6667). From Table 4, MCTLBO is significantly better than RADE, DREA,

Table 4 Results obtained by the Wilcoxon test for MCTLBO in terms of RR and SR compared with other algorithms

VS	RR			SR		
	R^+	R^-	p-value	R^+	R^-	p-value
RADE	390.0	75.0	**1.00E-05**	411.0	54.0	**1.04E-04**
DREA	338.5	126.5	**2.64E-02**	347.0	118.0	**1.19E-02**
FONDE	322.0	143.0	*5.10E-02*	322.0	143.0	*5.10E-02*
HNDE/2A	234.5	230.5	*9.58E-01*	234.5	230.5	*9.58E-01*
DDE/R	247.0	218.0	*7.45E-01*	275.5	189.5	*3.46E-01*
KSDE	314.5	150.5	*6.59E-02*	319.0	146.0	*5.85E-02*
TPEA	272.5	192.5	*3.81E-01*	247.0	218.0	*7.45E-01*
MONES	437.5	27.5	**1.92E-05**	437.5	27.5	**0.00E+00**
A-WeB	426.0	39.0	**5.10E-05**	426.0	39.0	**6.31E-05**
SaTriMODE	262.5	202.5	*5.30E-01*	264.5	200.5	*5.03E-01*
LSTP	349.0	116.0	**9.12E-03**	366.0	99.0	**1.1E-03**
C-FOA	412.5	52.5	**1.98E-04**	412.5	52.5	**2.01E-06**
C-GWO	396.0	39.0	**6.60E-05**	396.0	39.0	**0.00E+00**
C-ELATLBO	397.0	68.0	**6.68E-04**	397.0	68.0	**1.76E-04**
C-ITLBO	367.0	68.0	**1.26E-04**	342.0	93.0	**6.28E-03**

Bold values represent that MCTLBO significantly outperforms these compared algorithms
Italic values show that there is no significant difference between the results obtained by MCTLBO and these algorithms when solving NESs



(a) F04 (b) F06

(c) F12 (d) F17

Fig. 2 Convergence curves of seven algorithm on F04 and F12

MONES, A-WeB, LSPT, C-FOR, C-GWO, C-ELATLBO and C-ITLBO since all $p-$values are lower than 0.05. Although there is no significant difference in the comparison results between MCTLBO and FNODE, HNDE/2A, DDE/R, KSDE, TPEA, and SaTriMODE, all R^{+} values are higher than R^{-}, which also indicates the superiority of MCTLBO.

Convergence on *RR*

To further investigate the convergence of MCTLBO on RR, Fig. 2 shows the convergence of RR obtained by MCTLBO and several algorithms[1] on F04, F06, F12 and F17. Concretely, MCTLBO can completely find all the roots of these functions. When solving F04, F06 and F12, our algorithm converges faster in the initial stage, slows down in the middle stage and is weaker than other algorithms. However, in the last stage, it starts to rise again and finds all the roots. When solving F17, the convergence of MCTLBO is weaker than that of DDE/R at the beginning, but after nearly 300 iterations, MCTLBO obviously obtained better convergence than other algorithms. Hence, the proposed MCTLBO can

[1] To facilitate the viewing, DDE/R, HNDE/2A, FONDE and ksde with better effects are selected for comparison. In addition, since TPEA did not obtain the original code, we did not choose to make a comparison.

Table 5 Comparison of MCTLBO and its variants on RR and SR

Prob.	PR					SR				
	MCTLBO	MCTLBO-1	MCTLBO-2	MCTLBO-3	MCTLBO-4	MCTLBO	MCTLBO-1	MCTLBO-2	MCTLBO-3	MCTLBO-4
F01	**1.00**	**1.00**	**1.00**	**1.00**	0.00	**1.00**	**1.00**	**1.00**	**1.00**	0.00
F02	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
F03	**1.00**	**1.00**	**1.00**	**1.00**	1.00	**1.00**	**1.00**	**1.00**	**1.00**	0.93
F04	**1.00**	1.00	**1.00**	0.99	0.73	**1.00**	0.97	**1.00**	0.87	0.00
F05	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
F06	**1.00**	**1.00**	**1.00**	1.00	0.65	**1.00**	**1.00**	**1.00**	0.97	0.00
F07	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
F08	**1.00**	**1.00**	**1.00**	**1.00**	0.96	**1.00**	**1.00**	**1.00**	**1.00**	0.77
F09	**1.00**	**1.00**	**1.00**	**1.00**	0.34	1.00	1.00	1.00	1.00	0.00
F10	**1.00**	**1.00**	**1.00**	**1.00**	0.65	**1.00**	**1.00**	**1.00**	**1.00**	0.33
F11	**1.00**	**1.00**	**1.00**	**1.00**	0.99	**1.00**	**1.00**	**1.00**	**1.00**	0.97
F12	**0.98**	0.96	0.98	0.93	0.65	**0.90**	0.80	0.90	0.73	0.03
F13	**0.87**	0.85	0.83	0.83	0.38	0.23	0.10	0.03	0.07	0.00
F14	**1.00**	**1.00**	**1.00**	**1.00**	0.87	**1.00**	**1.00**	**1.00**	**1.00**	0.20
F15	1.00	1.00	1.00	1.00	1.00	**1.00**	**1.00**	**1.00**	**1.00**	1.00
F16	**1.00**	**1.00**	**1.00**	**1.00**	0.88	**1.00**	**1.00**	**1.00**	**1.00**	0.13
F17	**1.00**	**1.00**	**1.00**	**1.00**	0.06	**1.00**	**1.00**	**1.00**	**1.00**	0.00
F18	**1.00**	**1.00**	**1.00**	**1.00**	0.99	**1.00**	**1.00**	**1.00**	**1.00**	0.97
F19	**1.00**	**1.00**	**1.00**	**1.00**	0.00	**1.00**	**1.00**	**1.00**	**1.00**	0.00
F20	**1.00**	**1.00**	**1.00**	**1.00**	0.72	**1.00**	**1.00**	**1.00**	**1.00**	0.03
F21	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
F22	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
F23	**1.00**	0.94	0.98	0.99	0.19	**1.00**	0.43	0.86	0.90	0.00
F24	1.00	1.00	1.00	1.00	0.98	1.00	1.00	1.00	1.00	0.80
F25	**0.98**	0.97	0.97	**0.98**	0.33	**0.97**	0.93	0.93	**0.97**	0.27
F26	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
F27	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
F28	1.00	1.00	1.00	1.00	0.08	1.00	1.00	1.00	1.00	0.03
F29	1.00	1.00	1.00	1.00	0.42	1.00	1.00	1.00	1.00	0.00
F30	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
Avg	**0.99**	0.99	0.99	0.99	0.70	**0.97**	0.94	0.96	0.95	0.48

Bold values represent the best results obtained by the algorithms when solving the current NES

Table 6 Average ranks of MCTLBO variants obtained by the Friedman test on RR and SR

Algorithm	Ranking (RR)	Ranking (SR)
MCTLBO	**2.4667**	**2.5000**
MCTLBO-1	2.6500	2.6500
MCTLBO-2	2.6833	2.6167
MCTLBO-3	2.8000	2.8333
MCTLBO-4	4.4000	4.4000

Bold values represent the best ranking of the algorithm compared with other algorithms

balance the diversity and convergence well, and can effectively solve nonlinear equations.

Analysis of different components of MCTLBO

Compared with the original TLBO, the search method and learning strategy of MCTLBO are modified in the teaching and learning phases. This section discuss the impact of different components on performance. The different combinations of MCTLBO are as follows:

- MCTLBO-1: only Eq. (12) is used in the teaching phase;
- MCTLBO-2: only Eq. (13) is used in the teaching phase;
- MCTLBO-3: Remove the improved determining seeds method in the learning phase;
- MCTLBO-4: Replace Eq. (14) with Eq. (5) in the learning phase.

The detailed results obtained by MCTLBO variants are shown in Table 5. In addition, the statistical results achieved by the Friedman and the Wilcoxon test are given in Tables 6 and 7, respectively. The analysis results are follow:

- MCTLBO-1 and MCTLBO-2: In these two MCTLBO variants, various learning rules are used. It can be seen that no matter which rule is used alone, both MCTLBO-1 and MCTLBO-2 performed worse than that of MCTLBO. In particular, the RR and SR values obtained on F12, F13 and F25 are all decreased. This shows the effectiveness of the fitness-ranking- based adaptive selection scheme.
- MCTLBO-3: In MCTLBO-3, the determining seeds method is removed. From the results, MCTLBO-3 lost roots on more NESs, such as F04, F05, F12, F13, F23 and F25. The reason may be that the lack of seed determination technology reduces the accuracy of subpopulation division and limits the algorithm's search efficiency.
- MCTLBO-4: MCTLBO-4 adopted the original learning rule in TLBO. It can be observed that the overall performance of MCTLBO-4 has a very significant decline. The main reason is that the original learning rule is easy to fall into the local optimal. Using this learning strategy in multiple sub-populations makes it difficult for the algorithm to find multiple roots in a single run.
- In summary, MCTLBO obtained the highest average RR and SR values. Moreover, MOTLBO has the best ranking in Table 6. Additionally, MCTLBO provides higher R^+ than R^- in all cases, which shows that MCTLBO has better search efficiency. Thus, combining two niching techniques, fitness-ranking-based adaptive selection scheme, the experience-learning-based learning rules can enhance the algorithm's ability to locate the roots of NESs.

Discussions

In MCTLBO, two niching technique (crowding and the improved speciation), are used to improve the population diversity, but two parameters (num and n) are involved. In MCTLBO, $num = 8$ and $n = 20$. In this section, the impact on the algorithm will be studied. The parameters are as follows: $num = [5, 10, 15], n = [10, 20]$. To facilitate comparison, we execute a combination tuning test for the two parameters. Thus, MCTLBO is compared with the following methods:

- MCTLBO-5: $[num = 5, n = 20]$
- MCTLBO-6: $[num = 5, n = 10]$
- MCTLBO-7: $[num = 10, n = 20]$
- MCTLBO-8: $[num = 10, n = 10]$
- MCTLBO-9: $[num = 15, n = 20]$
- MCTLBO-10: $[num = 15, n = 10]$

Figure 3 shows the average RR and SR obtained for different parameters. The analysis results are as follows:

- In addition to MCTLBO-10, MCTLBO and MCTLBO-5 $\sim$ MCTLBO-9 achieve similar performance on RR and SR. The reason is that the MCTLBO-10 uses a larger num and a smaller n. Large num may limit the convergence speed of algorithm while small n make the algorithm search inadequate, resulting in the phenomenon of root loss.
- The statistical result achieved by Friedman test is given in Table 8. It can be seen that MCTLBO obtains the best RR and SR rankings. It also shows that proper parameters have great influence on the performance of MCTLBO. Therefore, the feasible range for the fusion of the two parameters is: $num\ in[5, 10]$ and $n \in [10, 20]$.

Study on the new test set

We select 18 new NESs [37] to further verify the MCTLBO's performance. In the new test set, the 18 equations have higher complexity, more roots and scalability. In this experiment, 13 algorithms from Sect. Compared algorithms are selected to compare with MCTLBO. It is worth noting that we do not have codes that can obtain TPEA and SaTriMODE, these two algorithms are excluded from the experiment in this section. All the algorithms were run for 30 times, and the relevant algorithms parameters were consistent with the original paper.

Tables 9 and 10 demonstrate the average RR and SR obtained by different algorithms on 18 new NESs. It can be seen that LSTP achieves the best result ($RR = 0.54$, $SR = 0.19$), followed by MCTLBO and HNDE/2A. In addition, Table 11 exhibits the rankings of the different algorithms obtained by Friedman test. LSTP get the highest ranking, followed by MCTLBO and HNDE/2A.

Concretely, our approach MCTLBO exhibits the promising result on MNE1, MNE4–MNE6, MNE8–MNE15. Although the overall performance of MCTLBO is lower than that of LSTP, it has better performance than other algorithms.

Table 8 Average ranks of MCTLBO variants obtained by the Friedman test on RR and SR

Algorithm	Ranking (RR)	Ranking (SR)
MCTLBO	**3.8833**	**3.8167**
MCTLBO-5	4.0333	4.0000
MCTLBO-6	4.0167	3.9833
MCTLBO-7	3.9333	3.9833
MCTLBO-8	4.1000	4.1167
MCTLBO-9	3.7667	3.7833
MCTLBO-10	4.2667	4.3167

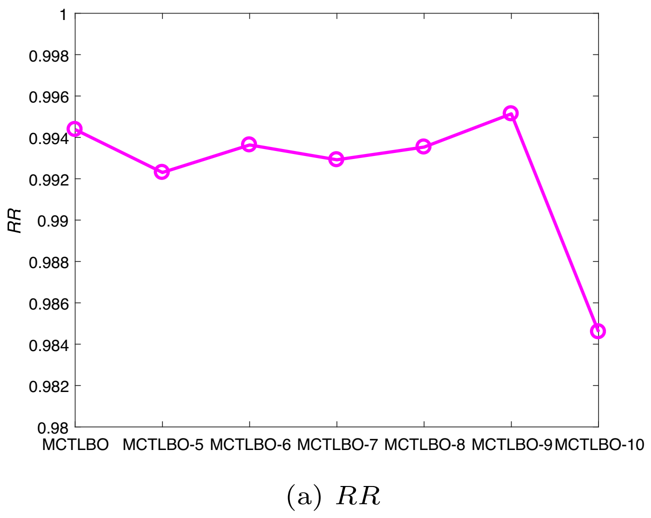
Bold values represent the best ranking of the algorithm compared with other algorithms

It is worth noting that LSTP is specifically designed to solve these 18 problems and thus achieves the best performance. When it tested the 30 NESs problems in Sect. Experimental results, its performance was far inferior to MCTLBO. Therefore, we need to improve MCTLBO according to the characteristics of these 18 test functions in the future, so that it can obtain better results when solving new test sets.

Table 7 Results obtained by the Wilcoxon test for MCTLBO in terms of RR and SR compared with other algorithms

VS	RR			SR		
	R^+	R^-	p-value	R^+	R^-	p-value
MCTLBO-1	274.0	191.0	*3.63E-01*	274.5	190.5	*3.56E-01*
MCTLBO-2	274.5	190.5	*3.58E-01*	247.5	217.5	*7.37E-01*
MCTLBO-3	300.0	165.0	*1.41E-01*	300.0	165.0	*1.40E-01*
MCTLBO-4	432.0	33.0	**1.43E-01**	432.0	33.0	**0.00E+00**

Bold values represent that MCTLBO significantly outperforms these compared algorithms
Italic values show that there is no significant difference between the results obtained by MCTLBO and these algorithms when solving NESs



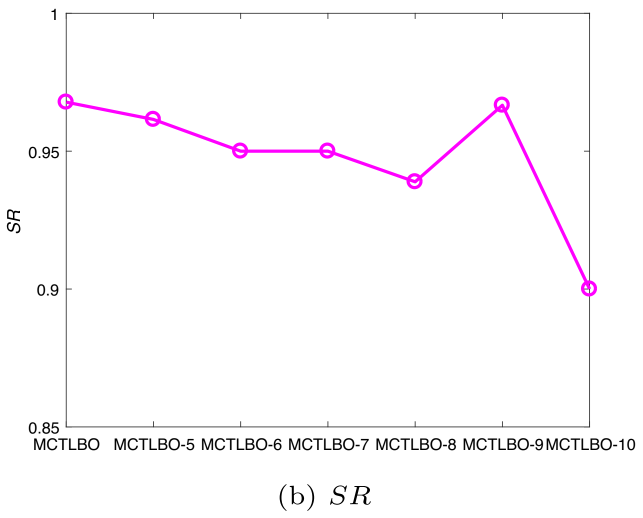


Fig. 3 Average RR and SR obtained by MCTLBO with differential parameters

Table 9 RR obtained by MCTLBO and other algorithms on the new test set

Instance	MCTLBO	RADE	DREA	FNODE	HNDE/2A	DDE/R	KSDE	MONES	A-WeB	LSTP	C-FOA	C-GWO	C-ELATLBO	C-ITLBO
MNE1	**0.50**	**0.50**	0.06	0.47	0.50	0.00	0.48	0.25	0.25	**0.50**	0.00	0.00	**0.50**	**0.50**
MNE2	0.17	**0.42**	0.00	0.00	0.48	0.00	0.00	0.25	0.25	0.00	0.00	0.00	0.00	**0.42**
MNE3	0.50	0.50	0.50	0.49	0.00	0.52	0.50	0.00	0.11	**1.00**	0.50	0.50	0.50	0.50
MNE4	**0.52**	0.50	0.10	0.50	0.50	0.12	0.00	0.50	0.50	0.50	0.00	0.00	0.50	0.50
MNE5	**0.50**	**0.50**	0.00	**0.50**	**0.50**	0.00	0.00	**0.50**	**0.50**	**0.50**	0.00	0.00	**0.50**	**0.50**
MNE6	**1.00**	**1.00**	**1.00**	0.40	**1.00**	0.17	**1.00**	0.60	0.32	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**
MNE7	0.73	0.56	0.46	0.67	0.78	0.43	**0.98**	0.47	0.09	0.77	0.44	0.53	0.47	0.42
MNE8	**0.92**	0.80	0.43	0.67	0.90	0.66	0.78	0.20	0.18	0.90	0.40	0.53	0.80	0.73
MNE9	**0.55**	0.57	0.40	0.60	0.47	0.42	0.45	0.20	0.10	0.53	0.35	0.35	0.53	0.48
MNE10	**0.66**	0.58	0.42	0.42	0.50	0.47	0.43	0.25	0.08	0.56	0.50	0.42	0.63	0.63
MNE11	**0.47**	0.44	0.01	0.11	0.51	0.03	0.39	0.17	0.17	0.45	0.00	0.00	0.44	0.33
MNE12	**0.27**	0.39	0.00	0.20	0.37	0.00	0.00	0.17	0.17	0.18	0.00	0.00	0.44	0.17
MNE13	**0.45**	0.50	0.00	0.34	0.29	0.00	0.32	0.00	0.13	0.37	0.00	0.00	0.11	0.17
MNE14	**0.66**	0.61	0.48	0.47	0.67	0.28	0.54	0.17	0.17	0.61	0.00	0.06	0.50	0.44
MNE15	**0.88**	0.64	0.57	0.17	0.88	0.49	0.53	0.31	0.06	0.28	0.08	0.10	0.59	0.41
MNE16	0.00	0.00	0.44	0.20	0.00	0.26	0.00	0.18	0.01	**0.46**	0.00	0.00	0.00	0.00
MNE17	0.00	0.00	**0.30**	0.26	0.00	0.24	0.00	0.04	0.00	0.29	0.00	0.00	0.00	0.00
MNE18	0.00	0.00	0.52	0.66	0.00	0.42	0.00	0.50	0.02	**0.90**	0.00	0.00	0.00	0.00
Avg	0.49	0.47	0.32	0.40	0.46	0.25	0.36	0.26	0.17	**0.54**	0.18	0.19	0.42	0.40

Bold values represent the best results obtained by the algorithms when solving the current NES

Table 10 SR obtained by MCTLBO and other algorithms on the new test set

Instance	MCTLBO	RADE	DREA	FNODE	HNDE/2A	DDE/R	KSDE	MONES	A-WeB	LSTP	C-FOA	C-GWO	C-ELATLBO	C-ITLBO
MNE1	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
MNE2	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
MNE3	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	**1.00**	0.00	0.00	0.00	0.00
MNE4	**0.03**	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
MNE5	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
MNE6	**1.00**	**1.00**	**1.00**	0.00	**1.00**	0.00	**1.00**	0.00	0.04	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**
MNE7	0.00	0.00	0.00	0.04	0.00	0.00	**0.72**	0.00	0.00	0.12	0.00	0.00	0.00	0.00
MNE8	**0.67**	0.00	0.00	0.00	0.52	0.00	0.00	0.00	0.00	0.52	0.00	0.00	0.00	0.00
MNE9	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
MNE10	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
MNE11	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
MNE12	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
MNE13	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
MNE14	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
MNE15	**0.08**	0.00	0.00	0.00	0.08	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
MNE16	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
MNE17	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
MNE18	0.00	0.00	0.04	0.00	0.00	0.16	0.00	0.00	0.00	**0.80**	0.00	0.00	0.00	0.00
Avg	0.10	0.06	0.06	0.00	0.09	0.01	0.10	0.00	0.00	**0.19**	0.06	0.06	0.06	0.06

Bold values represent the best results obtained by the algorithms when solving the current NES

Table 11 The average ranking of RR and SR obtained by different algorithms on the new test set

Algorithm	Ranking (RR)	Ranking (SR)
MCTLBO	4.5278	6.6667
RADE	4.9722	7.7500
DREA	8.5278	7.4167
FNODE	6.8333	7.8333
HNDE/2A	5.2222	6.8889
DDE/R	9.0833	7.7778
KSDE	7.9722	7.3056
MONES	8.8889	8.1667
A-WeB	9.7500	8.0556
LSTP	**4.2222**	**6.1389**
C-FOA	10.9167	7.7500
C-GWO	10.7500	7.7500
C-ELATLBO	6.2222	7.7500
C-ITLBO	7.1111	7.7500

Bold values represent the best ranking of the algorithm compared with other algorithms

Conclusion

To locate multiple roots of NESs, we have proposed a multi-population cooperative teaching–learning-based optimization. In our approach, two niching technique–crowding and an improved speciation, are integrated into TLBO to enhance population diversity. Then new learning rules are designed and fitness-ranking-based adaptive selection scheme promotes MCTLBO's root location efficiency. Experimental results summary that MCTLBO achieved the best RR and SR when compared with several state-of-the-art algorithms. In addition, MCTLBO exhibited promising performance on the new test set.

In future work, we intend to apply the proposed MCTLBO to solve several real-world problems in motor systems [41] and robot control [42]. We will also focus on modifying MCTLBO for the new test set. Furthermore, multi-task evolution [43, 44] may be utilized to design new approach, where multiple NES problems can be simultaneously solved in a single run.

Acknowledgements This work was partly supported by Special talent Project of Guangxi Science and Technology Base under Grant No. GuiKe AD21220119, and Major Research Development Program of Hubei Province (No. 2022BAD084).

Data Availability Data sharing is not applicable to this article as no new data were created or analyzed in this study.

Declarations

Conflict of interest The authors declare no conflict of interest, and the data generated during and/or analysed during the current study are available from the corresponding author on reasonable request.

References

1. Xiao L, Zhang Z, Li S (2019) Solving time-varying system of nonlinear equations by finite-time recurrent neural networks with application to motion tracking of robot manipulators. IEEE Trans Syst Man Cybern Syst 49(11):2210–2220
2. Bartosiewicz Z, Kaldmäe A, Kawano Y, Kotta U, Pawluszewicz E, Simha A, Wyrwas M (2021) Accessibility and system reduction of nonlinear time-delay control systems. IEEE Trans Autom Control 66(8):3781–3788
3. Doban AI, Lazar M (2018) Computation of lyapunov functions for nonlinear differential equations via a massera-type construction. IEEE Trans Autom Control 63(5):1259–1272
4. Jafari R, Razvarz S, Gegov A (2020) Neural network approach to solving fuzzy nonlinear equations using z-numbers. IEEE Trans Fuzzy Syst 28(7):1230–1241
5. Gong W, Liao Z, Mi X, Wang L, Guo Y (2021) Nonlinear equations solving with intelligent optimization algorithms: a survey. Compl Syst Model Simul 1(1):15–32
6. Schwandt H (2007) Parallel interval Newton-like Schwarz methods for almost linear parabolic problems. J Comput Appl Math 199(2):437–444
7. Gritton KS, Seader J, Lin W-J (2001) Global homotopy continuation procedures for seeking all roots of a nonlinear equation. Comput Chem Eng 25(7–8):1003–1019
8. Back T, Hammel U, Schwefel H-P (1997) Evolutionary computation: comments on the history and current state. IEEE Trans Evol Comput 1(1):3–17
9. Li S, Gong W, Gu Q (2021) A comprehensive survey on metaheuristic algorithms for parameter extraction of photovoltaic models. Renew Sustain Energy Rev 141:110828
10. Li S, Gong W, Yan X, Hu C, Bai D, Wang L, Gao L (2019) Parameter extraction of photovoltaic models using an improved teaching-learning-based optimization. Energy Convers Manag 186:293–305
11. Feng L, Zhou L, Gupta A, Zhong J, Zhu Z, Tan K-C, Qin K (2021) Solving generalized vehicle routing problem with occasional drivers via evolutionary multitasking. IEEE Trans Cybern 51(6):3171–3184
12. Liang Z, Luo T, Hu K, Ma X, Zhu Z (2021) An indicator-based many-objective evolutionary algorithm with boundary protection. IEEE Trans Cybern 51(9):4553–4566
13. Qu B, Suganthan P, Liang J (2012) Differential evolution with neighborhood mutation for multimodal optimization. IEEE Trans Evol Comput 16(5):601–614
14. Li S, Gong W, Wang L, Yan X, Hu C (2020) Optimal power flow by means of improved adaptive differential evolution. Energy 198:117314
15. Li S, Gong W, Wang L, Gu Q (2022) Multi-objective optimal power flow with stochastic wind and solar power. Appl Soft Comput 114:108045

16. Gong W, Wang Y, Cai Z, Wang L (2020) Finding multiple roots of nonlinear equation systems via a repulsion-based adaptive differential evolution. IEEE Trans Syst Man Cybern Syst 50(4):1499–1513
17. Liao Z, Gong W, Yan X, Wang L, Hu C (2020) Solving nonlinear equations system with dynamic repulsion-based evolutionary algorithms. IEEE Trans Syst Man Cybern Syst 50(4):1590–1601
18. He W, Gong W, Wang L, Yan X, Hu C (2019) Fuzzy neighborhood-based differential evolution with orientation for nonlinear equation systems. Knowl-Based Syst 182:104796
19. Liao Z, Gong W, Wang L (2020) Memetic niching-based evolutionary algorithms for solving nonlinear equation system. Expert Syst Appl 149:113261
20. Liao Z, Gong W, Wang L, Yan X, Hu C (2020) A decomposition-based differential evolution with reinitialization for nonlinear equations systems. Knowl-Based Syst 191:105312
21. Wu J, Gong W, Wang L (2021) A clustering-based differential evolution with different crowding factors for nonlinear equations system. Appl Soft Comput 98:106733
22. Wang K, Gong W, Liao Z, Wang L (2022) Hybrid niching-based differential evolution with two archives for nonlinear equation system. IEEE Trans Syst Man Cybern Syst 52(12):7469–7481
23. Liao Z, Zhu F, Gong W, Li S, Mi X (2022) AGSDE: archive guided speciation-based differential evolution for nonlinear equations. Appl Soft Comput 122:108818
24. Song W, Wang Y, Li H-X, Cai Z (2015) Locating multiple optimal solutions of nonlinear equation systems based on multiobjective optimization. IEEE Trans Evol Comput 19(3):414–431
25. Gong W, Wang Y, Cai Z, Yang S (2017) A weighted biobjective transformation technique for locating multiple optimal solutions of nonlinear equation systems. IEEE Trans Evol Comput 21(5):697–713
26. Gao W, Luo Y, Xu J, Zhu S (2020) Evolutionary algorithm with multiobjective optimization technique for solving nonlinear equation systems. Inf Sci 541:345–361
27. Gao W, Li G, Zhang Q, Luo Y, Wang Z (2021) Solving nonlinear equation systems by a two-phase evolutionary algorithm. IEEE Trans Syst Man Cybern Syst 51(9):5652–5663
28. Ji J-Y, Wong ML (2021) An improved dynamic multi-objective optimization approach for nonlinear equation systems. Inf Sci 576:204–227
29. Rao R, Savsani V, Vakharia D (2023) Teaching-learning-based optimization: a novel method for constrained mechanical design optimization problems. Comput Aided Des 43(3):303–315
30. Mi X, Liao Z, Li Sh, Gu Q (2021) Adaptive teaching-learning-based optimization with experience learning to identify photovoltaic cell parameters. Energy Rep 7:4114–4125
31. Zhang Y, Jin Zh, Chen Y (2020) Hybrid teaching-learning-based optimization and neural network algorithm for engineering design optimization problems. Knowl Based Syst 187:104836
32. Alneamy J, Alnaish Z, Hashim S, Alnaish R (2019) Utilizing hybrid functional fuzzy wavelet neural networks with a teaching learning-based optimization algorithm for medical disease diagnosis. Comput Biol Med 112:103348
33. Yang N, Tang Z, Cai X (2022) Cooperative multi-population Harris Hawks optimization for many-objective optimization. Complex Intell Syst 8:3299–3332
34. Zheng M, Fukuyama Y, El-Abd M, Iizaka T, Matsui T (2020) Optimization Overall, of Smart City by Multi-population Global-best Brain Storm Optimization using Cooperative Coevolution, IEEE Congress on Evolutionary Computation (CEC). Glasgow 2020:1–7
35. Storn R, Price K (1997) Differential evolution - a simple and efficient heuristic for global optimization over continuous spaces. J Global Optim 11(4):341–359
36. Jin J (2021) A robust zeroing neural network for solving dynamic nonlinear equations and its application to kinematic control of mobile manipulator. Complex Intell Syst 7:87–99
37. Gao W, Li Y (2021) Solving a new test set of nonlinear equation systems by evolutionary algorithm. IEEE Trans Cybern 53(1):406–415
38. Manizheh G, Mohammad R (2014) Forest optimization algorithm. Expert Syst Appl 41(15):6676–6687
39. Seyedali M, Seyed M, Mirjalilib A (2014) Grey wolf optimizer. Adv Eng Softw 69:46–61
40. Li Sh, Gong W, Yan X (2019) Parameter extraction of photovoltaic models using an improved teaching-learning-based optimization. Energy Convers Manag 786(15):293–305
41. Do TD, Choi HH, Jung J (2012) SDRE-based near optimal control system design for PM synchronous motor. IEEE Trans Ind Electron 59(11):4063–4074
42. Guo D, Nie Z, Yan L (2017) The application of noise-tolerant zd design formula to robots kinematic control via time-varying nonlinear equations solving. IEEE Trans Syst Man Cybern Syst 48(12):2188–2197
43. Gupta A, Ong Y-S, Feng L (2016) Multifactorial evolution: toward evolutionary multitasking. IEEE Trans Evol Comput 20(3):343–357
44. Wei T, Wang S, Zhong J, Liu D, Zhang J (2021) A review on evolutionary multi-task optimization: Trends and challenges. IEEE Trans Evol Comput 2:1