Information Sciences 619 (2023) 2–18

Contents lists available at ScienceDirect

Information Sciences

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

Co-evolutionary competitive swarm optimizer with three-phase for large-scale complex optimization problem

Chen Huang [a], Xiangbing Zhou [b,*], Xiaojuan Ran [b,c], Yi Liu [d], Wuquan Deng [e], Wu Deng [b,f]

[a] College of Civil Aviation, Shenyang Aerospace University, Shenyang 110136, China
[b] School of Information and Engineering, Sichuan Tourism University, Chengdu 610100, China
[c] International College of Digital Innovation, Chiang Mai University, Chaing Mai 50200, Thailand
[d] Research Center of Big Data and Information Management, Civil Aviation Management Institute of China, Beijing 100102, China
[e] Department of Endocrinology, Chongqing University Central Hospital, Key Laboratory for Biorheological Science and Technology of Ministry of Education, Chongqing University, Chongqing 400014, China
[f] School of Electronic Information and Automation, Civil Aviation University of China, Tianjin 300300, China

ARTICLE INFO

Article history:
Received 29 August 2022
Received in revised form 26 October 2022
Accepted 7 November 2022
Available online 11 November 2022

Keywords:
CSO
Three-phase co-evolutionary strategy
Update strategy
Optimization performance
Large-scale

ABSTRACT

Practical optimization problems often involve a large number of variables, and solving them in a reasonable amount of time becomes a challenge. Competitive swarm optimizer (CSO) is an efficient variant of particle swarm optimization (PSO) algorithm and has been applied extensively to deal with a variety of practical large-scale optimization problems. In this article, a novel co-evolutionary method with three-phase, namely TPCSO, is developed by incorporating a novel multi-phase cooperative evolutionary technique to enhance the convergence and the search ability of CSO. In the modified CSO, the population is evenly decomposed into two sub-populations, then the update strategy of each sub-population is adjusted by the requirements of the diversity and convergence during the evolution process. In the first phase, the diversity is paid more attention in order to explore more regions. And in the second phase, the promising area in two sub-populations are exploited by introducing excellent particles of two sub-populations. The third phase focuses on the convergence by learning from the global best solution. Finally, the performance of TPCSO is evaluated and proved by large-scale benchmark functions selected from CEC'2010 and CEC'2013. The experimental and statistical results show that TPCSO can effectively solve these large-scale problems and fast obtain the optimal results with higher accuracy by comparing with several algorithms.

1. Introduction

In recent years, as a powerful and promising swarm intelligence technique, evolutionary computation (EC) algorithm has been successfully applied to deal with a large number of complex science and practical engineering problems [1–5]. Since the continuous expansion of the application fields, the studies of the evolutionary computation (EC) algorithms have been witnessed the rapid development and become a research hotspot[6–10].The representative evolutionary computation (EC) algorithms mainly include genetic algorithm (GA) [11], ant colony optimization (ACO) [12], particle swarm optimization (PSO) [13], differential evolution (DE) [14–16], and so on.

* Corresponding author.
E-mail address: zhouxb@uestc.edu.cn (X. Zhou).

https://doi.org/10.1016/j.ins.2022.11.019
0020-0255/

It is worth noting that PSO is a famous and effective evolutionary computation (EC) algorithm inspired by the mimicry of birds flocking. Although the traditional PSO introduced in 1995 [17] has rapidly progressed in recent years and demonstrated competitive performance in various research fields, such as shop scheduling, parameter selection, function optimization, neural network training and path planning due to simple structure and search ability, it still exists premature convergence and local optima when dealing with complicated problems.

To overcome these shortcomings, some PSO variants have been addressed and discussed over time [18–22]. Zhan *et al.* [23] proposed an adaptive PSO (APSO) and elite learning strategy to avoid local optima. Dadvar *et al.* [24] proposed a modified PSO by introducing DE algorithm to prevent population stagnation. Zeng *et al.* [25] proposed a new switching PSO mechanism by utilizing dynamic-neighborhood to avoid premature problem. Lynn *et al.* [26] developed the ensemble PSO (EPSO) by hybridizing different PSO variants with self-adaptive scheme to solve real-parameter optimization problems. Wang *et al.* [27] designed multiple subpopulations co-evolution mechanism depending on machine-learning techniques to strengthen the optimization ability of the algorithm. Lu *et al.* [28] proposed hybrid PSO-RVM algorithm by associating with a real-valued mutation operator to enhance global search capability. For premature convergence of PSO, Chaitanya *et al.* [29] proposed memory retention and multiple sub-swarms to avoid the prematurity. Zhao *et al.* [30] proposed improved a gradient-based adaptive PSO to alleviate the problem of local optimality. Yan *et al.* [31] proposed a novel learning strategy to alleviate premature convergence of PSO.

PSO algorithm has achieved good results in solving low-dimensional problems. Nevertheless, similar to most EC algorithms, it is often reported that the existing performance degrades dramatically as the dimensions increases, Cheng *et al.* [32] proposed a novel variant of PSO, named CSO, for large-scale optimization. The evolution of CSO is driven by a pairwise competition mechanism. After each competition, only those loser particles need to be updated according to the positions of the winners, whereas the winner particles in the competition will directly enter into the next iteration. In CSO optimization process, there are two challenges to deal with in designing adaptive strategies. Firstly, the learning strategy of CSO focuses more on diversity, while the convergence is ignored to some extent. Secondly, the single update strategy of the loser has certain blindness in the searching direction and is difficult to satisfy the diversity and convergence of the algorithm in different search stages of the whole evolutionary process. Therefore, in order to further improve the search efficiency of the CSO algorithm, a novel three-phase co-evolutionary strategy based on multi-phase cooperative evolutionary technique is designed to develop an enhanced CSO (i.e. TPCSO) to enhance the diversity and global search ability of CSO.

The main contributions of this paper are summarized as follows:

(1) A modified competitive swarm optimizer by introducing three-phase co-evolutionary strategy (i.e. TPCSO) is developed to enhance exploration and exploitation ability to deal with large-scale problems, where the population is evenly into two subpopulations and the losers in each sub-population are co-evolved.
(2) Three-phase co-evolutionary strategy is designed to control the exploration and exploitation ability of two subpopulations in different evolutionary phases.
(3) A novel update strategy with the information exchange between two subpopulations is designed to explore more promising regions, which can effectively help the population to improve the efficiency of evolution process.

The remainder of this article is organized as follows. In section 2, the research works of high dimensional optimization and some PSO variants are briefly reviewed. Section 3 describes the details of TPCSO. Section 4 introduces the detailed steps of TPCSO. In Section 5, some large-scale optimization functions are utilized to verify the global search ability of TPCSO. Finally, the conclusions are described in Section 6.

2. Related works

For small-scale optimization problems, the performance of evolutionary algorithms outperforms non-evolutionary algorithms. However, as the dimension number of the problem grows, the performance of the traditional EC algorithms begins to show a sharp decline trend because the search space is huge. Many practical optimization problems have become a large-scale optimization problem (LOP) (more than 500 dimensions) and caused the slow convergence and exponentially increasing local optima. Since the presence of these two important challenges, most evolutionary algorithms often have the problem that their performance degrades sharply when the dimension increases greatly. The recent advances in solving high-dimensional problems may be roughly grouped into the two following categories.

2.1. Decomposition-based ECs

Collaborative co-evolutionary (CC) mechanism is one potential method to solve the dimensional issue in many high dimension problems, which was firstly introduced into EC algorithms (CCECs) in 1994 [33]. In CCECs, a divide-and-conquer decomposition approach is adopted to decompose the original high dimensional problem into several low dimensional sub-problems and then each sub-problem is optimized separately. Due to grouping strategy, the CCECs preserve higher diversity than the evolutionary algorithms. In recent years, many ECs variants employed CC framework to enhance of search performance in such a complex environment. Yang *et al.* [34] developed a decomposition method (DECC-G) by adopting random grouping of decision variables to deal with nonseparable problems. The literature [35] proposed a multi-level CC algorithm (MLCC) to replace a fixed number for subcomponent. Li *et al.* [36] presented an improved PSO (CCPSO) algorithm, where random grouping scheme was used in PSO. On the basis of CCPSO, CCPSO2 with new adaptive scheme was developed to solve more complex multimodal problems [37]. For the design of the decomposition strategy, the literature [38] proposed differential grouping strategy (DECC-DG) to decompose a large number of decision variables. A variant of the differential grouping algorithm (DG2) is proposed to detect a reliable threshold value to reuse the sample points [39]. Ma *et al.* [40] proposed a merged differential grouping (MDG) algorithm based on subset-subset interaction and binary search to the efficiency of problem decomposition. Yang et al. [41] developed a recursive differential grouping method by utilizing the historical information on interrelationship examination to reduce computation cost. Hasanzadeh *et al.* [42] proposed an intelligent cooperative PSO based on learning automata algorithm for high-dimensional multimodal problems. Zhao *et al.* [43] developed dynamic grouping method by regrouping the subpopulations according to the fitness values. Ma *et al.* [44] proposed a novel decomposition approach based on localized control variable analysis (LSMOEA/D) to incorporate the guidance of reference vectors into decision variable analysis.

2.2. Non-decomposition-based ECs

Non-decomposition-based ECs, unlike decomposition-based ECs, takes all variables as a whole and mainly depends on different effective evolutionary schemes to preserve higher diversity. In order to save space, some representative works for PSO variants are listed. The literature [25] designed a social learning PSO (SL-PSO) based on historical information by learning from better particles according to the result of the fitness comparison. In addition, the literature [45] developed a level-based learning swarm optimizer, which has achieved a considerable success. CSO algorithm is also an improved PSO algorithm by introducing the competition of the particles, where a pair of particles is randomly selected and compared for fitness value, and the particle that lost the comparison was driven by learning from the winner, while the particle that wins the comparison directly entered into the next generation. The loser is updated [35] as

$$V_L(t+1) = r_1(t)V_L(t) + r_2(t)(X_W(t) - X_L(t)) + \varphi r_3(t)\left(\bar{X}(t) - X_L(t)\right) \tag{1}$$

$$X_L(t+1) = X_L(t) + V_L(t+1) \tag{2}$$

where $V_L(t)$ and $X_L(t)$ is the velocity and position of the loser in the t-th generation, respectively. $X_W(t)$ denotes the position of the winner.$r_1(t)$, $r_2(t)$ and $r_3(t)$ are the random variable within [0, 1], $\bar{X}(t)$ is the mean position of the population, and φ denotes the controlling parameter, which represents the impact of$\bar{X}(t)$.

3. An enhanced CSO (TPCSO)

3.1. Motivation

To avoid premature convergence, CSO focuses more on diversity and less on convergence in the solution search by introducing a competition mechanism. Because a good optimizer should adjust exploration ability and exploitation ability to traverse the search space when the dimension increases, it means that the algorithm should provide different convergence and diversity in different stages of the whole search. However, the single update strategy of CSO makes it unable to meet the requirement of diversity and convergence during the evolution process. Therefore, the different update learning strategies are required in the different phases. In this way, the search abilities can be quickly enhanced by learning from different superior particles. Therefore, a novel three-phase co-evolutionary strategy is designed by making full use of the advantages of multi-phase cooperative evolutionary technique to develop an enhanced CSO (namely TPCSO). The following section describes the TPCSO algorithm in detail.

3.2. Description of TPCSO

A feasible and appropriate update operation can effectively facilitate the solution procedure to quickly converge to a high-quality solution. The preference of the method is relative to the update strategy. Therefore, on the basis of the CSO, a novel and efficient update strategy is developed to explore the more promising regions by exchanging the information between two subpopulations. In the proposed TPCSO, the swarm is equally decomposed into two sub-populations depending on the competition of the fitness. The winners of two sub-populations are represented by W_1 andW_2, and the loser particles are represented by L_1 andL_2. After the comparison, the particle having a better fitness value, called the winner, can directly enter into the next iteration, and the loser is updated based on three-phase learning mechanism. The realization process of TPCSO with the three phases is shown in Fig. 1. Depending on the requirement of diversity and convergence in different evolution phase, the corresponding evolution strategy is adopted to lead the evolution of the algorithm. For the early phase of the algorithm, more diversity is expected to achieve a wide range of search over the entire space. Both the convergence and diversity is required to focus on in the middle stage. In this stage, each loser is guided by two winners from different

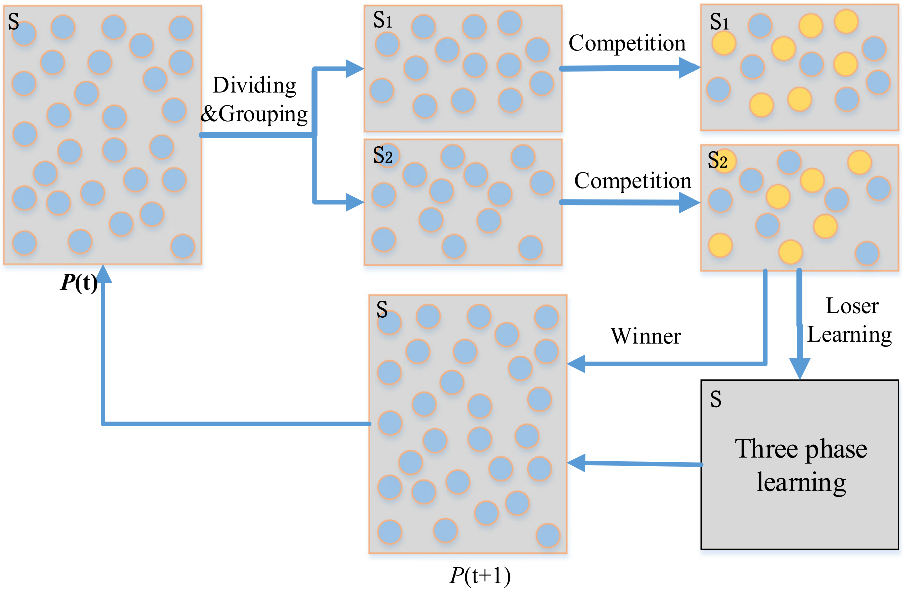


Fig. 1. The realization process of the TPCSO by the three phases.

sub-populations. And then in the late phase of the evolution process, all individuals are expected to converge to the optimal solution as fast as possible.

The realization descriptions of the TPCSO are described as follows.

(1) Phase 1

In the evolution of TPCSO, since the search space of large scale problems is huge for the exploration, the more diversity should be implicitly taken into consideration during the first phase of the evolution. The exploration is a main task in the initial phase to traverse the entire space and locate the optimum regions. The update strategy is developed to provide the evolution direction of the losers in each sub-population. For each sub-population, the update rules of each loser are given as.

$$\begin{cases} V_{LS_i}(t+1) = r_1(t)V_{LS_i}(t) + r_2(t)\left(X_{WS_i}(t) - X_{LS_i}(t)\right) + \varphi_1 r_3(t)\left(\bar{X}_{S_i}(t) - X_{LS_i}(t)\right) \\ X_{LS_i}(t+1) = X_{LS_i}(t) + V_{LS_i}(t+1) \end{cases} (i = 1 \quad or \quad 2) \tag{3}$$

where $X_{WS_i}(t)$ denotes the position of the winner in i-th $(i = 1 \quad or \quad 2)$ sub-population, $X_{LS_i}(t)$ is the position of the loser in i-th $(i = 1 \quad or \quad 2)$ sub-population, i represents the sub-population index, φ_1 is a parameter, which is used to control the influence of$\bar{X}_{S_i}(t).r_1(t)$ and $r_2(t)$ are random numbers between 0 and 1.

(2) phase 2

In the middle phase (i.e. Phase 2) of the evolution process, in order to accelerate the optimization process and search the optimal solution, a mutual learning approach is employed to exchange the information between two sub-populations to expedite the search of the optimal solution. In here, the convergence rate and diversity can be further enhanced. At each iteration, a pair of particles from each sub-population compare their fitness values. The winners in two sub-populations are used to update the losers in the other sub-population. The new update strategy generated by information sharing of two sub-populations can find more accurate solutions and improve the convergence. The exchange of the information between two sub-populations can avoid to falling into local optimum.

Each loser in two sub-populations is implemented through parallelization mechanism. For sub-component 1, the velocity and position update rules of each loser are given as follows.

$$\begin{cases} V_{LS_1}(t+1) = r_1(t)V_{LS_1}(t) + r_2(t)\left(X_{WS_1}(t) - X_{LS_1}(t)\right) + \varphi_2 r_3(t)\left(X_{WS_2}(t) - X_{LS_1}(t)\right) \\ X_{LS_1}(t+1) = X_{LS_1}(t) + V_{LS_1}(t+1) \end{cases} \tag{4}$$

where $X_{WS_1}(t)$ and $X_{WS_2}(t)$ denote the positions of the winners in two sub-populations. $X_{LS_1}(t)$ and $X_{LS_2}(t)$ are the positions of the losers in two sub-populations. φ_2 is one control parameter, which represents the influence of$X_{WS_2}(t)$.$r_1(t)$,$r_2(t)$ and $r_3(t)$ are random numbers between 0 and 1.

For sub-component 2, the velocity and position update rules of each loser are given as follows.

$$\begin{cases} V_{LS_2}(t+1) = r_1(t)V_{LS_2}(t) + r_2(t)\big(X_{WS_2}(t) - X_{LS_2}(t)\big) + \varphi_2 r_3(t)\big(X_{WS_1}(t) - X_{LS_2}(t)\big) \\ X_{LS_2}(t+1) = X_{LS_2}(t) + V_{LS_2}(t+1) \end{cases} \tag{5}$$

where $X_{WS_1}(t)$ and $X_{WS_2}(t)$ denote the positions of the winners in two sub-populations. $X_{LS_1}(t)$ and $X_{LS_2}(t)$ are the positions of the losers in two sub-populations. φ_2 is one control parameter, which represents the influence of$X_{WS_2}(t)$.$r_1(t)$,$r_2(t)$ and $r_3(t)$ are random numbers between 0 and 1.

(3) Phase 3

Phase 3 is the last phase of the evolution process. The exploitation seems to need more effectiveness and efficiency in the later phase of the search, which aims to improve the equality of the found solutions in the exploration phase. Along with the evolution process iteratively, the individual in population should explore the region near global optimum. To further enhance the convergence speed and solution accuracy, the operator adapts the best particle in two sub-populations to improve the population convergence of CSO moderately.

For each sub-population, the update formula of the winner is described as follows.

$$\begin{cases} V_{LS_i}(t+1) = r_1(t)V_{LS_i}(t) + r_2(t)\big(X_{WS_i}(t) - X_{LS_i}(t)\big) + \varphi_3 r_3(t)\big(gbest(t) - X_{LS_i}(t)\big) \\ X_{LS_i}(t+1) = X_{LS_i}(t) + V_{LS_i}(t+1) \end{cases} \quad (i = 1 \quad \text{or} \quad 2) \tag{6}$$

where $gbest(t)$ is the best solution found by all particles so far, $X_{WS_i}(t)$ denotes the position of the winner in two sub-populations. $X_{LS_i}(t)$ is the position of the loser in two sub-populations. φ_3 is one control parameter, which represents the influence of$gbest(t)$.$r_1(t)$,$r_2(t)$ and $r_3(t)$ are random numbers between 0 and 1.

Through the analysis of the diversity in the evolutionary process, the division of evolutionary stages is determined to choose appropriate evolutionary strategies.

4. Model, steps and complexity of TPCSO

4.1. Model of TPCSO

According to the three-phase co-evolutionary strategy, TPCSO algorithm is demonstrated in Algorithm 1. Algorithm 1 shows how three-phase co-evolutionary strategy is incorporated into CSO algorithm. The TPCSO algorithm starts by generating a random initial population, and then the entire population is randomly divided into two subpopulations. Two particles were randomly selected from the two subpopulations and compare their fitness. The particles with larger fitness values are considered as the losers, while the particles with smaller fitness values are considered as the winners. The winners directly enter into the next generation. Lines 6 and 19 of Algorithm 1 display this process. In order to enhance the efficiency of evolution process, different excellent particles are used to lead the losers in each subpopulation, which are shown in lines 20 and 29 of Algorithm 1. Specifically, in the first phase, the winner particle and the mean position of each subpopulation are adopted to update the positions of the losers. This is the same as the CSO algorithm. Then, in the second phase, the winner particle in other subpopulation is used to replace the mean position of each subpopulation for finding better solutions. In the third stage, the global best particle in entire swarm is selected to optimize the losers in each subpopulation to enhance the convergence ability of the algorithm. The same process is repeated until the terminal condition is met.

Algorithm 1 The main procedure of the TPCSO.

N is the size of swarm, t is the iteration number, the terminal condition is the maximum number of iteration (*Maxgen*).

1: Initialize the iteration number t = 0.

Generate a random initial population$P_i(0) = (x_{i1}, x_{i2}, \cdots, x_{iD})$; D is the dimension of the particle, *FEs = 0*;

*2: **while** the terminal condition is not met **do***

3: Calculate the fitness value of each particle;

4: Divide evenly N particles into 2 sub-populations S_1 *and* S_2 *randomly*;

5:$U_{S_i} = P_{S_i}(t)$,$P_{S_i}(t+1) = (i = 1 \quad or \quad 2)$

*6: **while*** $U_{S_i} \neq$ ***do***

7: randomly select two particles$X_{S_1P_1}(t)$, $X_{S_1P_2}(t)$ *from*U_{S_1} $(i = 1)$;

randomly select two particles$X_{S_2P_1}(t)$, $X_{S_2P_2}(t)$ *from* U_{S_2} $(i = 2)$

8: if $f(X_{S_1P_1}(t)) \leqslant f(X_{S_1P_2}(t))$ *then*

$X_{WS_1}(t) = X_{S_1P_1}(t), X_{LS_1}(t) = X_{S_1P_2}(t)$;
else
$X_{WS_1}(t) = X_{S_1P_2}(t), X_{LS_1}(t) = X_{S_1P_1}(t)$;
end if
Put $X_{WS_1}(t)$ into $P(t+1)$;
if $f(X_{S_2P_1}(t)) \leqslant f(X_{S_2P_2}(t))$ then
$X_{WS_2}(t) = X_{S_2P_1}(t), X_{LS_2}(t) = X_{S_2P_2}(t)$;
else
$X_{WS_2}(t) = X_{S_2P_2}(t), X_{LS_2}(t) = X_{S_2P_1}(t)$;
end if
Put $X_{WS_2}(t)$ into $P(t+1)$;
if $0 \leqslant t \leqslant p_1$
Update $X_{LS_i}(t)$ using (3);
else if $p_1 < t \leqslant p_2$
Update $X_{LS_1}(t)$ using (4);
Update $X_{LS_2}(t)$ using (5);
else if $p_2 < t \leqslant Maxgen$
update $X_{LS_i}(t)$ using (6);
end if
Add the updated $X_{LS_1}(t+1)$ and $X_{LS_2}(t+1)$ into $P_s(t+1)$;
remove $X_{S_1}(t)$, $X_{S_2}(t)$ from U_s
end while
$t = t + 1$;
end while

4.2. Steps of TPCSO

The steps of the TPCSO are demonstrated as follows.

Step 1. The initial position $P_i(0) = (x_{i1}, x_{i2}, \cdots, x_{iD})$ of the population is randomly initialized within $(x_{j\min}, x_{j\max})$ $(1 \leqslant j \leqslant D)$.

$$x^0_{i,j} = x_{j\min} + rand(0,1) \times (x_{j\max} - x_{j\min}) \tag{7}$$

where $i = 1, 2, \cdots, n$, and $j = 1, 2, \cdots, D$. Initialize the swarm size (N), the maximum number of fitness evaluations ($MaxFEs$), the maximum number of iteration ($Maxgen$), the number of initial fitness evaluation($FEs = 0$), the initial number of iteration($t = 0$), and the three phases are initialized p_1 and p_2, respectively.

Step 2. The swarm is evenly decomposed into two sub-populations randomly, and calculate the fitness value of each particle.
Step 3. Divide each sub-population into $N/4$ pairs randomly.
Step 4. The particles of each pair participate in the comparison of fitness values, the winners and losers are generated according to the competition.
Step 5. The TPCSO with three-phase learning strategy is used to update the loser in each subpopulation.
Step 6. If $0 \leqslant t \leqslant p_1$, the update strategy of the phase 1 is employed to optimize the position of the loser in each sub-population. If $p_1 < t \leqslant p_2$, the update strategy of the phase 2 is employed to optimize the position of the loser in each sub-population. If $p_2 < t \leqslant Maxgen$, the update strategy of Phase 3 is employed to optimize the position of the loser in each sub-population.
Step 7. If the stopping condition is not met, return to **Step 2**. Otherwise, TPCSO is end and the obtained results are output.

4.3. Time complexity

According to the main procedure of TPCSO, the time complexity of each iteration is analysed and calculated. For D-dimensional problem, it is noted that the losers of each sub-population take $O(NP/4D)$ to update at each generation for the function evaluation time. Therefore, the time complexity of two sub-populations is $O((NP/4D) \times 2)$. So the time complexity of TPCSO can represented as $O(N/2D)$. In conclusion, TPCSO takes the same time complexity by comparing with the CSO.

5. Experimental study

5.1. Test functions

The effectiveness and efficiency of TPCSO is assessed in this section, where the benchmark test functions are selected from the CEC'2010 benchmark sets [46] and CEC'2013 benchmark sets [47]. CEC2010 includes 20 benchmark functions. Compared with CEC'2010, CEC'2013 is more complex due to the introduction of unbalanced contributions from subcomponents and overlapping functions. The CEC'2013 test function set includes 15 large-scale benchmark problems. TPCSO is adopted to compare with CSO [32], DECC-G [33], MLCC [34], CCPSO2 [37], DECC-DG [38], DMS-L-PSO [43].

5.2. Experimental environment and parameter setting

All the comparison algorithms are run independently for 25 turns to evaluate the optimization performance. To be fair, the number of fitness evaluations is set to be the same and equal to $3000 \times D$ for each test function in all the algorithm. *NP* is set to 500. Through several simulation results, we set the maximum number of the iterations to be 12,000 and find that $0 < p_1 \leqslant \frac{1}{6}Maxgen$ for phase 1, $\frac{1}{6}Maxgen < p_2 \leqslant \frac{5}{6}Maxgen$ for phase 2 and $\frac{5}{6}Maxgen < p_3 \leqslant Maxgen$ for phase 3 are compatible for TPCSO when *MaxFEs* is set to be $3000 \times D$. For the TPCSO, φ_2 and φ_3 are set to 0.3 for all the functions except two functions f_{13} of CEC'2013 test suite. In the functions f_{13} of the CEC'2013 function sets,$\varphi_2 = \varphi_3 = 0.1$. The key parameters settings of other algorithms are the same as those in the corresponding literatures for fair comparison.

5.3. Evaluation indexes

In order to demonstrate the advantage of the TPCSO, some evaluation indexes, such as the mean valueμ, standard deviation σ and *t*-value are selected [48], which are calculated as follows.

$$\mu = \frac{\sum_{r=1}^{run} x_r}{run} \tag{8}$$

$$\sigma = \sqrt{\frac{\sum_{r=1}^{run} (x_r - \mu)^2}{run}} \tag{9}$$

where *run* is number of runs, x_r is the best solution in *r*-th run.

To compare two algorithms, the *t*-value can be obtained by the mean value and standard deviations as follow.

$$t = \frac{\mu_1 - \mu_2}{\sqrt{\frac{|\sigma_1^2 - \sigma_2^2|}{run}}} \tag{10}$$

where μ_1 and μ_2 are the means of two algorithms respectively,σ_1 and σ_2 are standard deviations of two algorithms respectively.

5.4. Experimental results and comparison for CEC'2010 test suite

Six PSO variants were chosen to compare with the TPCSO on 20 test functions with 1000 dimensions from CEC'2010. The reason for the comparison is to demonstrate the performance of the TPCSO. The mean experiment results are presented in 25 independent runs in Table 1. The best mean value on each function is highlighted with boldface. *w*/*t*/l reflects that TPCSO wins its competitor in *w* functions, ties in *t* functions, and loses in *l* functions. The convergence curves of CSO and TPCSO are shown in Fig. 2.

As can be seen from the Tab1e 1 and Fig. 2, the TPCSO consistently outperforms other six algorithms: CSO(20/20), DECC-G (19/20), MLCC(18/20), CCPSO2(15/20), DECC-DG(16/20), DMS-L-PSO(13/20) on most of the CEC'2010 functions. Compared with other six popular PSO variants, TPCSO performs a significant effectiveness on more than half of the CEC'2010 functions set. We discussed the performance of the algorithms from three categories. For the first category (fully separable functions f_1 - f_3), the statistical results show that the TPCSO obtains the better fitness value than other comparative algorithms except functionf_2, where CCPSO2 and DECC-G have the slightly better than TPCSO, but TPCSO dominates CSO, DMS-L-PSO, DECC-DG, and DECC-G on functionf_2. For the second category (partially separable functionsf_4 - f_{18}, the TPCSO significantly outperforms the CSO, DECC-G, MLCC, CCPSO2, DECC-DG, DMS-L-PSO on the functions$f_5 f_6 f_7 f_8 f_{10} f_{11} f_{13} f_{15} f_{16}$ andf_{18}. TPCSO is better than CSO, DECC-G, MLCC, CCPSO2, DECC-DG, DMS-L-PSO on the functions f_4, f_9 and f_{14} except for DMS-L-PSO. For the functions f_{12} andf_{17}, although TPCSO is not the best value among all comparison algorithms, the obtained results are better than those of CSO, MLCC, CCPSO2, DECC-G. The results of Table 1 reveal that TPCSO provides the best solutions for over half of the partially separable functions. For the third category (non-separable functions f_{19} - f_{20}), TPCSO performs better than CSO, MLCC, CCPSO2, DECC-G and DECC-DG on the function f_{20}, while it fails on the functionf_{19}. In general, TPCSO shows

Table 1
The obtained experiment results on test functions with 1000*D* (CEC'2010).

Fun	Quality	TPCSO	CSO	MLCC	CCPSO2	DMS-L-PSO	DECC-DG	DECC-G
f_1	MeanStd.	**1.57E-21**	4.21E-12	8.73E-13	2.38E+00	1.65E+07	6.90E+02	2.68E-07
	t -Values	1.83E-22	7.69E-13	2.10E-12	5.07E+00	1.44E+06	1.54E+03	3.86E-08
		-	**−2.74E+01**	**−2.08E+00**	**−2.35E+00**	**−5.73E+01**	**−2.24E+00**	**−3.47E+01**
f_2	Mean	4.51E+02	7.63E+03	**2.67E+00**	4.26E+00	6.12E+03	4.50E+03	1.32E+03
	Std.	1.05E+01	1.72E+02	1.62E+00	1.20E+00	2.46E+02	4.42E+01	4.42E+02
	t -Values	-	**−2.09E+02**	2.16E+02	2.14E+02	**−1.15E+02**	**−4.72E+02**	**−9.83E+00**
f_3	Mean	**3.41E-14**	2.67E-09	2.05E-07	4.63E-03	1.64E+01	1.65E+01	1.26E+00
	Std.	1.95E-15	2.91E-10	4.53E-07	1.78E-03	6.88E-01	2.30E-01	6.84E-01
	t -Values	-	**−4.59E+01**	**−2.26E+00**	**−1.30E+01**	**−1.19E+02**	**−3.59E+02**	**−9.21E+00**
f_4	Mean	8.60E+11	1.27E+12	9.37E+12	2.05E+12	**4.95E+11**	4.28E+12	2.53e+13
	Std.	1.42E+11	1.78E+11	2.86E+12	1.24E+12	3.42E+10	1.39E+12	3.85E+12
	t -Values	-	**−1.91E+01**	**−1.49E+01**	**−4.83E+00**	1.32E+01	**−1.24E+01**	**−3.18E+01**
f_5	Mean	**3.18E+06**	5.73e+06	5.16E+08	4.27E+08	9.41E+07	1.17E+08	3.02e+08
	Std.	1.66E+06	1.08e+06	1.14E+08	1.37E+08	1.19E+07	1.93E+07	6.83E+07
	t -Values	-	**−1.01E+01**	**−2.25E+01**	**−1.55E+01**	**−3.86E+01**	**−2.96E+01**	**−2.19E+01**
f_6	Mean	**3.58E-09**	7.98E-07	1.89E+07	1.86E+07	3.62E+01	1.66E+01	4.49E+06
	Std.	2.77E-11	7.92e-08	3.64E+06	5.14E+06	1.38E+01	4.07E-01	2.93E+05
	t -Values	-	**−5.02E+01**	**−2.60E+01**	**−1.81E+01**	**−1.31E+01**	**−2.04E+02**	**−7.66E+01**
f_7	Mean	**2.93E+02**	1.90E+044.19E+03	1.48E+08	3.24E+08	4.49E+06	8.91E+03	5.21e+08
	Std.	1.24E+02	**−2.23E+01**	1.35E+08	3.56E+08	2.57E+05	6.78E+03	3.59E+08
	t -Values	-		**−5.48E+00**	**−4.55E+00**	**−8.73E+01**	**−6.36E+00**	**−7.26E+00**
f_8	Mean	**2.79E+07**	3.81E+07	5.85E+07	4.05E+07	5.15E+07	3.31E+07	7.37E+07
	Std.	5.08E+06	7.25E+05	3.24E+07	3.58E+07	2.88E+07	2.84E+07	1.79E+07
	t -Values	-	**−1.01E+01**	**−4.78E+00**	−1.78E+00	**−4.16E+00**	−9.30E-01	**−1.33E+01**
f_9	Mean	3.97E+07	5.89E+07	2.45E+08	9.97E+07	**2.09E+07**	4.35E+07	4.80E+08
	Std.	6.04E+06	5.51E+06	2.26E+07	3.42E+07	1.09E+06	3.35E+06	7.01E+07
	t -Values	-	**−3.88E+01**	**−4.71E+01**	**−8.91E+00**	1.58E+01	**−3.78E+00**	**−3.15E+01**
f_{10}	Mean	**4.22E+02**	9.62E+03	4.37E+03	5.10E+03	8.21E+03	4.32E+03	1.01E+04
	Std.	3.73E+01	6.97E+01	1.68E+03	7.69E+02	4.24E+02	8.69E+01	2.45E+02
	t -Values	-	**−7.81E+02**	**−1.18E+01**	**−3.05E+01**	**−9.22E+01**	**−2.48E+02**	**−2.00E+02**
f_{11}	Mean	**1.87E-13**	4.40E-08	1.96E+02	1.96E+02	1.65E+02	9.80E+00	2.53E+01
	Std.	9.70E-14	2.28E-09	1.73E+00	2.25E+00	2.88E+00	9.14E-01	1.36E+00
	t -Values	-	**−9.65E+01**	**−5.66E+02**	**−4.36E+02**	**−2.86E+02**	**−5.36E+01**	**−9.30E+01**
f_{12}	Mean	3.24E+04	5.48E+05	1.03E+05	3.68E+04	**3.61E+01**	1.33E+03	1.04E+05
	Std.	1.14E+04	3.74E+04	1.54E+04	1.72E+04	3.10E+00	1.03E+02	6.73E+03
	t -Values	-	**−7.24E+01**	**−3.41E+01**	−1.71E+00	1.42E+01	1.36E+01	**−3.89E+01**
f_{13}	Mean	**5.13E+02**	7.02E+02	4.37E+03	1.34E+03	2.05E+04	4.77E+03	2.72E+03
	Std.	1.69E+02	5.36E+01	4.29E+03	1.28E+02	5.14E+03	2.59E+06	5.64E+02
	t -Values	-	**−5.90E+00**	**−4.50E+00**	**−3.75E+01**	**−1.95E+01**	**−8.22E-03**	**−2.05E+01**
f_{14}	Mean	1.47E+08	2.78E+08	5.76E+08	2.85E+08	**1.58E+07**	3.24E+08	1.02E+09
	Std.	8.83E+06	9.44E+06	5.23E+07	1.33E+08	3.62E+06	2.02E+07	4.99E+07
	t -Values	-	**−1.96E+02**	**−4.16E+01**	**−5.20E+00**	8.15E+01	**−4.87E+01**	**−8.89E+01**
f_{15}	Mean	**7.96E+02**	1.01E+04	8.97E+03	1.06E+04	6.43E+03	5.88E+03	1.29E+04
	Std.	1.55E+02	6.32E+01	2.15E+03	1.37E+03	3.62E+02	9.18E+01	5.83E+02
	t -Values	-	**−3.29E+02**	**−1.91E+01**	**−3.60E+01**	**−8.61E+01**	**−2.04E+02**	**−1.08E+02**
f_{16}	Mean	**3.23E-13**	5.78E-08	3.82E+02	3.98E+02	3.20E+02	6.08E-13	6.66E+01
	Std.	7.71E-14	2.40E-09	5.74E+01	5.87E-01	1.29E+01	8.16E-14	4.88E+00
	t -Values	-	**−1.20E+02**	**−3.33E+01**	**−3.39E+03**	**−1.24E+02**	**−5.33E+01**	**−6.82E+01**
f_{17}	Mean	1.78E+05	2.22E+06	3.45E+05	1.19E+05	**5.88E+01**	4.08E+04	3.07E+05
	Std.	4.98E+04	7.89E+04	3.27E+04	5.13E+04	2.24E+01	2.68E+03	9.67E+03
	t -Values	-	**−1.67E+02**	**−2.22E+01**	2.40E+01	1.79E+01	1.38E+01	**−1.32E+01**
f_{18}	Mean	**1.20E+03**	2.24E+03	1.78E+04	3.09E+03	2.59E+04	1.16E+10	3.69E+04
	Std.	3.66E+02	4.09E+02	8.63E+03	2.86E+02	1.19E+03	2.23E+09	1.64E+04
	t -Values	-	**−2.85E+01**	**−9.63E+00**	**−4.14E+01**	**−1.09E+02**	**−2.60E+01**	**−1.09E+01**
f_{19}	Mean	2.95E+06	1.11E+07	1.98E+06	1.54E+06	2.07E+06	1.70E+06	**1.15E+06**
	Std.	2.98E+05	5.00E+05	1.37E+05	1.06E+05	1.62E+05	8.74E+04	5.86E+04
	t -Values	-	**−1.01E+02**	1.83E+01	2.53E+01	1.76E+01	2.19E+01	3.08E+01
f_{20}	Mean	9.96E+02	1.06E+03	2.30E+03	2.13E+03	**9.85E+02**	6.31E+10	4.14E+03
	Std.	5.24E+01	1.02E+02	2.25E+02	1.73E+02	5.95E+01	5.36E+06	3.61E+02
	t -Values	-	**−3.66E+00**	**−2.98E+01**	**−3.44E+01**	1.95E+00	**−5.89E+04**	**−4.40E+01**
w/*t*/l			20/0/0	18/0/2	15/2/3	13/1/6	16/1/3	19/0/1

its advantages on functions$f_1 f_3 f_5 f_6 f_7 f_8 f_{10} f_{11} f_{13} f_{15} f_{16}$, and$f_{18}$, respectively. In contrast with CSO, TPCSO obtains better optimization performance. It can further observe that the TPCSO can provide better fitness value in the most of the first two types of functions. Therefore, the exploitation capability of the TPCSO is enhanced by adopting three-phase co-evolutionary strategy. From the Fig. 2, the convergence curve of the proposed TPCSO is compared with CSO for all the 20 functions. TPCSO obtains faster convergence in most of these 20 functions. In the early the evolution process, TPCSO has

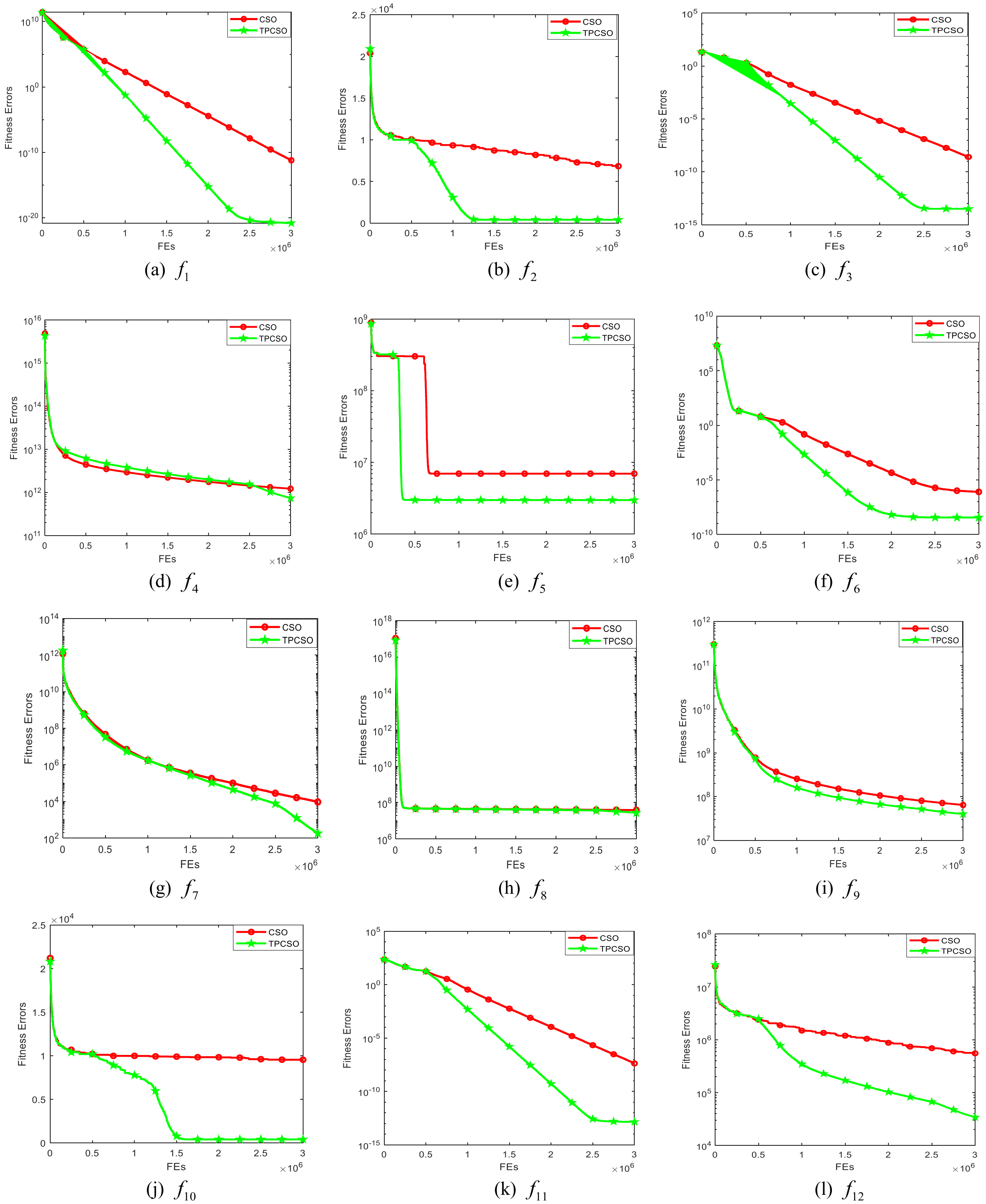
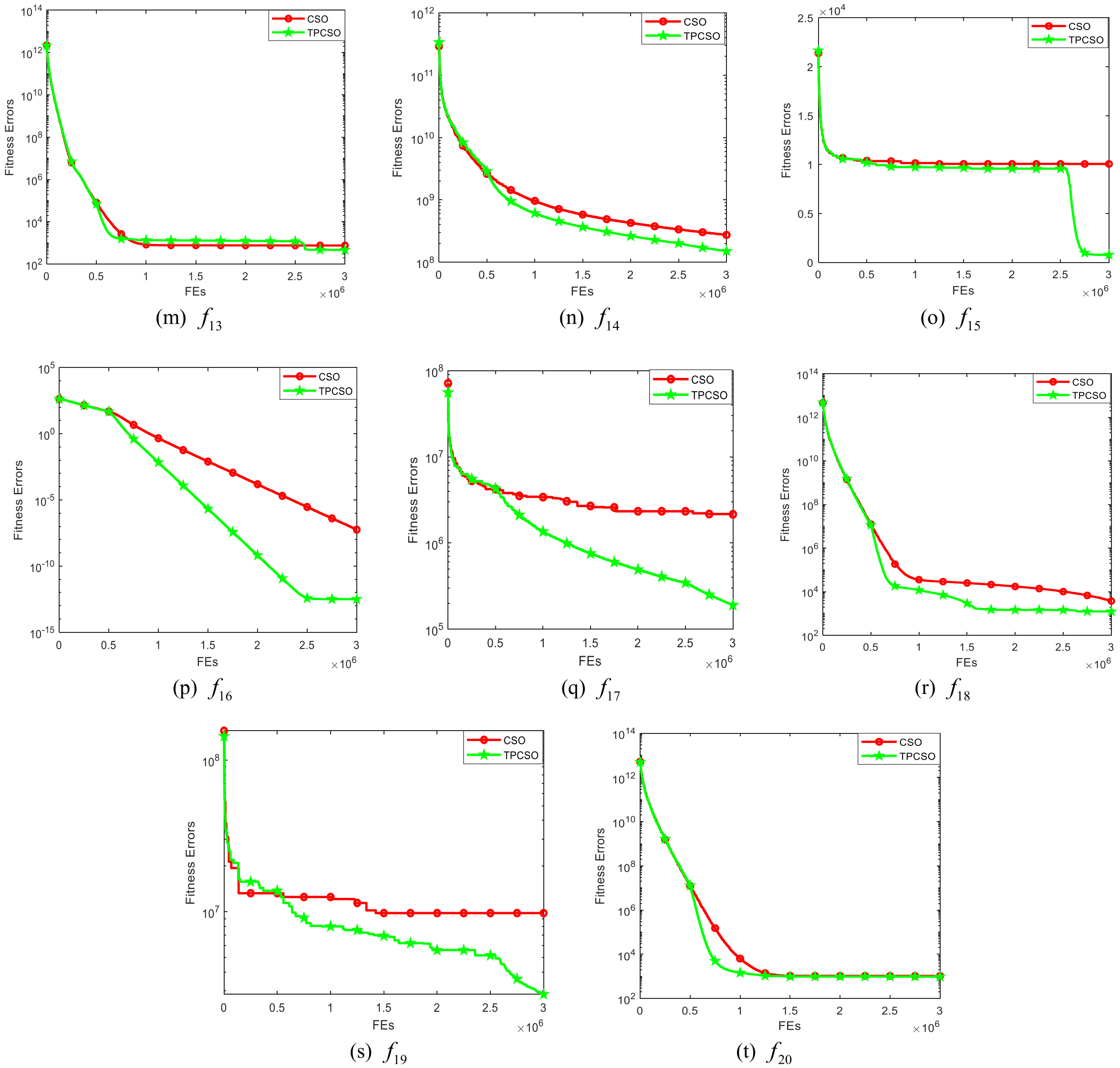


Fig. 2. The convergence curves of the CSO and TPCSO on test functions with 1000D (CEC'2010).

the same convergence as the traditional CSO algorithm. The convergence curves effectively reflect that the TPCSO is more accurate than CSO. With the increase of iterations, TPCSO exceeds traditional CSO except for two partially separable functions f_4 andf_{13}. In the late phase of the evolution process, it shows that TPCSO is superior to CSO on the functions$f_4 f_7 f_8 f_{12} f_{13} f_{17} f_{19}$. Hence conclusively, the TPCSO can obtain better optimization ability than CSO.



(m) f_{13} (n) f_{14} (o) f_{15}

(p) f_{16} (q) f_{17} (r) f_{18}

(s) f_{19} (t) f_{20}

Fig. 2 (*continued*)

5.5. Experimental results and comparison for CEC'2013 test suite

To further assess the efficiency of the TPCSO in solving large-scale problems, the CEC'2013 large-scale benchmark set is selected. CEC'2013 presents some new features. In CEC'2013 benchmark set, it contains Sphere, Elliptic, Rastrigin's, Ackley's, Schwefel's, and Rosenbrock's functions. As a result, CEC'2013 is much more complex and more difficult to solve than CEC'2010. The comparison results of seven algorithms are summarized in Table 2. The convergence curves of CSO and TPCSO are shown in Fig. 3.

As can be seen from the Table 2 and Fig. 3, for the separable functionsf_1 - f_3, TPCSO displays its superiority, which is significantly better than those of other several comparison algorithms on the functionf_1. The TPCSO is superior to CSO, DMS-PSO, DECC-G and DECC-DG on the functionf_2, but loses MLCC and CCPSO2. Although TPCSO is a little worse than MLCC, CCPSO2 and DECC-G, it still performs significantly better than CSO on the functionf_3. For the partially separable functionsf_4 - f_{11}, TPCSO shows best optimization ability than other several algorithms on the functionsf_4 - f_{11}, which dominates all the other algorithms in solving these eight functions. The experiment results show that TPCSO is very stable. For the three overlapping functionsf_{12} - f_{14}, TPCSO can obtain better than most compared algorithms, which displays its superiority and stability. For the last non-separable function, although TPCSO performs a little worse than CCPSO2, it dominates the other compared algorithms. As can be seen from the Fig. 3, the three-phase strategy can help TPCSO converge to better solutions

Table 2
The obtained experiment results on test functions with 1000D (CEC'2013).

Fun	Quality	TPCSO	CSO	MLCC	CCPSO2	DMS-L-PSO	DECC-DG	DECC-G
f_1	Mean	**2.38E-21**	7.15E-12	8.58E-10	3.86E+01	2.30E+09	3.35E+00	1.93E-06
	Std.	2.30E-22	4.20E-13	2.01E-09	2.34E+01	1.35E+08	4.91E+00	7.65E-07
	t –Values	-	**−8.51E+01**	**−2.13E+00**	**−8.25E+00**	**−8.52E+01**	**−3.41E+00**	**−1.26E+01**
f_2	Mean	5.55E+02	8.50E+03	**3.74E+00**	3.62E+01	8.72E+03	1.30E+04	1.29E+03
	Std.	1.35E+01	2.41E+02	1.75E+00	5.34E+00	4.95E+02	1.30E+03	4.59E+01
	t -Values	-	**−1.65E+02**	2.06E+02	2.09E+02	**−1.01E+02**	**−4.79E+01**	**−8.38E+01**
f_3	Mean	2.14E+01	2.16E+01	**2.00E+01**	**2.00E+01**	2.14E+01	2.14E+01	2.02E+01
	Std.	1.36E-02	4.30E-03	2.67E-02	1.36E-04	1.96E-01	1.89E-02	5.83E-02
	t -Values	-	**−7.75E+01**	3.05E+02	5.15E+02	0.00E+00	0.00E+00	1.06E+02
f_4	Mean	**5.88E+09**	1.57E+10	2.29E+11	3.45E+10	3.78E+11	4.07E+10	1.83E+11
	Std.	7.24E+08	1.87E+09	1.27E+11	2.01E+10	2.05E+10	8.27E+09	5.26E+10
	t -Values	-	**−2.85E+01**	**−8.78E+00**	**−7.12E+00**	**−9.08E+01**	**−2.11E+01**	**−1.68E+01**
f_5	Mean	**5.00E+05**	6.74E+05	1.28E+07	1.57E+07	5.57E+06	4.76E+06	9.38E+06
	Std.	1.35E+04	7.93E+04	3.59E+06	4.38E+06	4.39E+05	3.97E+05	2.55E+06
	t -Values	-	**−1.11E+01**	**−1.71E+01**	**−1.74E+01**	**−5.78E+01**	**−5.37E+01**	**−1.74E+01**
f_6	Mean	**1.02E+06**	1.06E+06	1.05E+06	1.05E+06	1.03E+06	1.06E+06	1.06E+06
	Std.	3.70E+03	1.23E+03	4.56E+03	7.92E+03	4.98E+03	1.03E+03	1.62E+03
	t -Values	-	**−5.73E+01**	**−5.63E+01**	**−2.14E+01**	**−1.50E+01**	**−5.63E+01**	**−6.01E+01**
f_7	Mean	**1.04E+06**	3.98E+06	1.44E+09	3.85E+08	3.36E+09	7.52E+07	7.01E+08
	Std.	3.49E+05	9.23E+05	1.27E+09	9.31E+08	6.87E+08	3.31E+07	2.11E+08
	t -Values	-	**−1.72E+01**	**−5.67E+00**	**−2.06E+00**	**−2.44E+01**	**−1.12E+01**	**−1.66E+01**
f_8	Mean	**1.17E+14**	2.28E+14	8.59E+15	1.16E+15	3.45E+14	4.04E+14	9.16E+15
	Std.	1.52E+13	3.36E+13	5.87E+15	4.98E+14	5.98E+14	2.03E+14	7.58E+15
	t -Values	-	**−1.85E+01**	**−7.22E+00**	**−1.05E+01**	−1.91E+00	**−7.09E+00**	**−5.97E+00**
f_9	Mean	**2.89E+07**	3.18E+07	9.84E+08	3.73E + 09	4.72E+08	6.61E+08	5.94E+08
	Std.	2.86E+06	7.18E+06	2.63E+08	1.04E+09	2.84E+07	1.32E+08	1.06E+08
	t -Values	-	**−2.20E+00**	**−1.82E+01**	**−1.78E+01**	**−7.84E+01**	**−2.39E+01**	**−2.67E+01**
f_{10}	Mean	**9.11E+07**	9.40E+07	9.27E+07	9.30E+07	9.31E+07	9.29E+07	9.25E+07
	Std.	3.54E+05	1.56E+05	6.04E+05	5.04E+05	1.03E+06	5.29E+05	4.27E+05
	t -Values	-	**−4.56E+01**	**−1.63E + 01**	**−2.65E+01**	**−1.03E+01**	**−2.29E+01**	**−2.93E+01**
f_{11}	Mean	**7.50E+07**	1.85E+09	9.03E+10	9.36E+11	1.57E+11	6.10E+10	8.61E+10
	Std.	9.42E+06	1.02E+09	7.65E+10	1.57E+10	7.53E+10	6.06E+10	4.12E+10
	t -Values	-	**−8.70E+00**	**−5.90E+00**	**−2.98E+02**	**−1.04E+01**	**−5.03E+00**	**−1.04E+01**
f_{12}	Mean	**1.06E+03**	**1.07E+03**	2.47E+03	2.13E+03	6.82E+04	1.04e+08	4.42E+03
	Std.	2.60E+01	2.89E+01	7.54E+02	1.85E+02	5.37E+04	1.10E+08	6.48E+02
	t -Values	-	**−3.96E+00**	**−9.29E+00**	**−2.89E+01**	**−6.25E+00**	**−4.73E+00**	**−2.59E+01**
f_{13}	Mean	**1.68E+08**	4.99E+08	7.20E+09	4.15E+09	1.37E+10	2.05E+10	9.25E+09
	Std.	3.98E+07	3.49E+08	2.01E+09	1.72E+09	6.41E+09	3.86E+09	2.37E+09
	t -Values	-	**−4.77E+00**	**−1.75E+01**	**−1.16E+01**	**−1.06E+01**	**−2.63E+01**	**−1.92E+01**
f_{14}	Mean	**9.15E+07**	3.19E+09	1.58E+11	9.34E+10	2.59E+11	1.98E+10	1.27E+11
	Std.	2.72E+07	3.05E+09	8.35E+10	8.79E+10	1.43E+11	1.42E+10	5.79E+10
	t -Values	-	**−5.08E+00**	**−9.46E+00**	**−5.31E+00**	**−9.05E+00**	**−6.94E+00**	**−1.10E+01**
f_{15}	Mean	8.11E+06	7.34E+07	7.98E+06	**5.42E+06**	1.65E+07	9.87E+06	1.13E+07
	Std.	4.02E+05	2.30E+06	8.52E+05	5.09E+06	3.72E+06	2.35E+06	1.86E+06
	t –Values	-	**−1.44E+02**	8.65E-01	2.65E+00	**−1.13E + 01**	**−3.80E+00**	**−8.78E+00**
w/t/l			14/1/0	12/1/2	12/0/3	13/2/0	14/1/0	14/0/1

quickly, which significantly performs better than CSO. Compared with CSO, TPCSO can adopt suitable strategies according to the evolution process and control the evolution to improve the search capability, the experimental results demonstrates that TPCSO has advantages over CSO.

Overall, the obtained results show that TPCSO perform well in terms of mean value and standard deviation. The improved TPCSO can provide a better optimization performance to obtain a high-quality solution in solving large-scale problems. Three-phase strategy can effectively enhance global search ability of CSO. The reason is that TPCSO with a three-stage control strategy can divide the whole evolution process into three phases. During the first phase, depending on the single neighbour control, the loser is guided by single winner and mean position of the whole swarm. During the second stage, the winner in the comparison is employed to guide the loser instead of the average position of the swarm, thus each loser takes two immediate neighbours to update its position. This way will further enhance the population diversity of traditional CSO. During the third phase, the global best particle is introduced to guide the evolution of the loser. Therefore, TPCSO with three-stage evolutionary strategy not only can enhance high population diversity, but also improve the convergence of the algorithm.

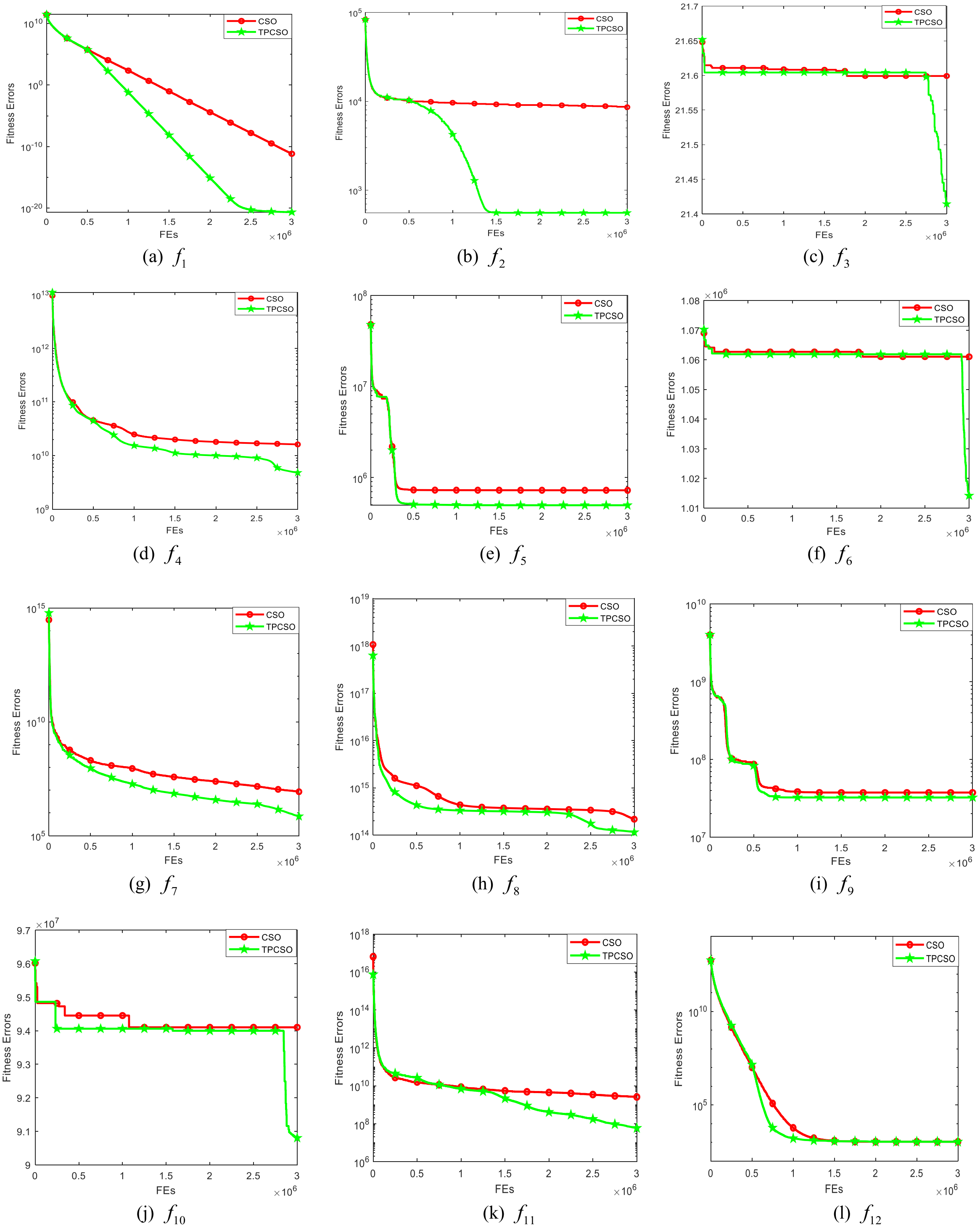
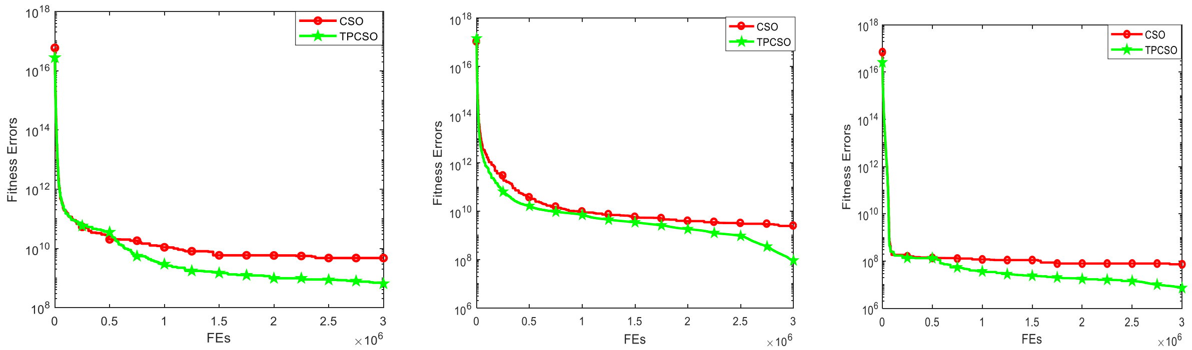


Fig. 3. The convergence curves of the CSO and TPCSO on test functions with 1000*D* (CEC'2013).



(m) f_{13} (n) f_{14} (o) f_{15}

Fig. 3 (*continued*)

5.6. The influence of parameters

The main parameters of TPCSO include the population size, control parameter of φ_2 andφ_3. Among them, φ_2 reflects the influence of the winner in the other sub-populations, while φ_3 denotes the influence of the global best particle in each sub-population.

(1) population size

The population size directly impacts on the preference of the TPCSO algorithm. In order to accurately acquire the impact of population size on the optimization performance of TPCSO, several experiments for different population sizes *m* = 200, 400 and 600 are conducted on CEC'2010 functions, and the statistical results are recorded in Table 3.

As can be seen from the Table 3, the size number of particles on 1000*D* should be 400. The optimization performance will deteriorate if the population size is larger than certain values. This is because the termination condition of this experiment is the maximum number of FEs, the larger population size need to spend more iterations cost to finish the task. It can also see from Table 3 that the larger population does not mean better optimization performance of TPCSO. For the 1000*D* problem, the smaller than 200 may be too small. Based on the above analysis, the population size is selected in the scope of no less than 400 and no more than 600 ($D \geq 1000$).

In addition, CEC'2010 contains 20 functions, which are composed of unimodal and multimodal functions. Among them, f_1,f_4,f_7,f_9,f_{12},f_{14},f_{17} and f_{19} are unimodal functions, while f_2,f_3,f_5,f_6,f_8,f_{10},f_{11},f_{13},f_{15},f_{16},f_{18} and f_{20} are multimodal functions. It can be seen from Table 3, for unimodal function, the optimization performance of *m* = 200 is better than *m* = 400 and the optimization performance of *m* = 400 is better than *m* = 600. For the multimodal functions, the larger population size is required to provide high diversity of the population. For example, the functionf_2, the optimization performance of *m* = 600 is best in three population sizes. Its reason is that the multimodal functions with numerous local extreme are more complex and difficult than unimodal functions. Compared with the unimodal functions, the multimodal functions will need higher diversity of the population.

(2) Control parameters φ_2 and φ_3

The optimization performance of TPCSO depends on control parameters of φ_2 and φ_3 to some extent. In order to reasonably set the two parameters, some test cases with different φ_2 and φ_3 varying from 0.1 to 0.3 are employed. The statistical results of different combinations of these two parameters are summarized in Table 3. From Table 3, it is noted that the social factor φ_2 and φ_3 can influence the convergence of TPCSO. If the number of particles is smaller and more diversity is needed, TPCSO with small φ_2 and φ_3 are better than those with larger parameters. If the number of particles is larger, the convergence is focused, the bigger φ_2 and φ_3 would perform better due to the convergence when the population size is larger. In general, the experiment results show that$\varphi_2 = 0.3$, and $\varphi_3 = 0.3$ can generate good result on these test functions with1000-*D* when *N* = 400 and *N* = 600 (*N* is the swarm size).

Information Sciences 619 (2023) 2–18

Table 3
The statistical results with different m, φ_2 andφ_3.

Fun	Swarm Size	$\varphi_2 = 0.1\varphi_3 = 0.1$	$\varphi_2 = 0.1\varphi_3 = 0.2$	$\varphi_2 = 0.1\varphi_3 = 0.3$	$\varphi_2 = 0.2\varphi_3 = 0.1$	$\varphi_2 = 0.2\varphi_3 = 0.2$	$\varphi_2 = 0.2\varphi_3 = 0.3$	$\varphi_2 = 0.3\varphi_3 = 0.1$	$\varphi_2 = 0.3\varphi_3 = 0.2$	$\varphi_2 = 0.3\varphi_3 = 0.3$
f_1	$m = 200$	1.18E-20	1.16E-19	5.72E-10	**7.53E-21**	3.72E-19	2.03E-07	9.72E-21	1.01E-11	0.0029
	$m = 400$	6.47E-15	7.05E-21	8.53E-21	4.63E-21	3.02E-21	3.07E-21	2.95E-21	1.84E-21	**1.15E-21**
	$m = 600$	1.56E-05	1.68E-05	1.94E-05	2.88E-18	3.70E-18	3.76E-18	5.54E-21	6.25E-21	**5.15E-21**
f_2	$m = 200$	1.18E+03	**1.16E+03**	1.19E+03	1.23E+03	1.28E+03	1.36E+03	1.30E+03	1.21E+03	1.17E+03
	$m = 400$	5.66E+03	630.1947	673.587	**520.3635**	542.2525	548.2223	541.2576	524.3433	554.1921
	$m = 600$	9.35E+03	9.42E+03	9.57E+03	366.1449	380.0756	392.0138	**361.1701**	376.0945	374.1046
f_3	$m = 200$	1.25E-00	**8.71E-01**	1.03E-00	1.41E-00	1.34E-00	1.28E-00	1.24E-00	1.36 E-00	1.23 E-00
	$m = 400$	8.16E-11	4.62E-14	6.04E-14	3.91E-14	2.85E-14	3.91E-14	3.55E-14	**2.84E-14**	3.20E-14
	$m = 600$	5.61E-06	5.64E-06	5.62E-06	2.72E-12	2.54E-12	2.70E-12	4.38E-14	4.32E-14	**4.26E-14**
f_4	$m = 200$	6.95E+11	6.89E+11	1.40E+12	7.41E+11	**5.16E+11**	2.09E+12	7.65E+11	5.57E+11	9.73e+11
	$m = 400$	9.71E+11	1.14E+12	5.96E+11	1.32E+12	8.33E+11	1.01E+12	1.16E+12	1.07E+12	**5.88E+11**
	$m = 600$	1.63E+12	1.45E+12	1.65E+12	1.72E+12	1.37E+12	1.62E+12	1.63E+12	1.49E+12	**1.06E+12**
f_5	$m = 200$	1.29E+07	1.49E+07	1.69E+07	1.49E+07	1.13E+07	2.07E+07	**9.95E+06**	1.19E+07	1.78E+07
	$m = 400$	4.99E+06	7.96E+06	5.97E+06	9.95E+06	5.00E+06	5.97E+06	5.97E+06	5.97E+06	**4.98E+06**
	$m = 600$	2.00E+06	5.98E+06	2.52E+08	9.95E+05	3.98E+06	2.01E+06	2.04E+06	2.99E+06	**1.99E+06**
f_6	$m = 200$	**3.65E-09**	1.03E-00	9.13E-01	4.68E-00	2.73E-00	2.61E-00	4.76E-00	4.51E-00	1.18E+01
	$m = 400$	1.46E-08	3.86E-09	3.87E-09	3.58E-09	3.55E-09	3.57E-09	3.56E-09	**3.54E-09**	3.56E-09
	$m = 600$	3.20E-05	3.04E-05	3.45E-05	4.34E-09	4.39E-09	4.64E-09	3.62E-09	3.64E-09	**3.60E-09**
f_7	$m = 200$	**1.35E+03**	1.98E+04	1.49E+05	1.73E+04	7.34E+04	8.62E+05	1.06E+05	7.21E+04	3.10E+05
	$m = 400$	4.06E+03	524.5343	354.3099	1.86E+03	474.7987	328.3426	2.35E+03	797.5611	**135.5641**
	$m = 600$	4.06E+04	3.06E+04	4.70E+04	1.47E+04	9.97E+03	1.10E+04	1.35E+04	8.27E+03	**3.30E+03**
f_8	$m = 200$	2.61E+07	1.61E+07	3.29E+07	1.69E+08	**1.38E+07**	3.46E+07	2.46E+07	1.51E+07	3.56E+07
	$m = 400$	3.68E+07	3.34E+07	2.93E+07	3.44E+07	3.11E+07	2.71E+07	3.33E+07	3.01E+07	**2.60E+07**
	$m = 600$	4.04E+07	4.04E+07	4.05E+07	3.77E+07	3.78E+07	3.78E+07	3.67E+07	3.67E+07	**3.65E+07**
f_9	$m = 200$	**2.55E+07**	4.23E+07	7.39E+07	3.51E+07	4.90E+07	1.07E+08	3.46E+07	4.53E+07	1.08E+08
	$m = 400$	5.41E+07	4.85E+07	5.11E+07	5.05E+07	4.15E+07	4.64E+07	4.58E+07	3.99E+07	**3.65E+07**
	$m = 600$	8.78E+07	9.38E+07	8.50E+07	5.75E+07	5.76E+07	4.85E+07	4.95E+07	**4.79E+07**	4.87E+07
f_{10}	$m = 200$	**806.2466**	1.01E+03	1.17E+03	1.15E+03	1.09E+03	1.20E+03	1.18E+03	1.10E+03	1.12E+03
	$m = 400$	9.62E+03	708.4784	778.4361	525.4571	**464.1266**	506.434	554.192	520.3634	543.2475
	$m = 600$	9.99E+03	1.01E+04	9.76E+03	7.57E+03	7.45E+03	7.58E+03	341.8652	322.6715	**298.5476**
f_{11}	$m = 200$	2.2244	1.3642	**1.2697**	19.1496	19.6475	18.8127	19.4614	20.3844	22.0775
	$m = 400$	1.42E-09	3.38E-13	3.41E-13	2.24E-13	1.95E-13	2.34E-13	2.13E-13	1.71E-13	**1.67E-13**
	$m = 600$	7.37E-05	6.99E-05	7.27E-05	3.08E-11	3.57E-11	3.60E-11	4.33E-13	**3.73E-13**	4.12E-13
f_{12}	$m = 200$	**9.06E+03**	2.13E+04	9.03E+04	1.46E+04	1.94E+04	9.29E+04	1.43E+04	2.66E+04	1.28E+05
	$m = 400$	4.00E+05	8.91E+04	9.75E+04	6.03E+04	3.04E+04	4.32E+04	5.26E+04	**2.78E+04**	3.23E+04
	$m = 600$	1.70E+06	1.55E+06	1.47E+06	9.61E+04	9.70E+04	1.03E+05	7.03E+04	**6.98E+04**	7.39E+04
f_{13}	$m = 200$	892.1433	1.05E+03	872.0985	1.22E+03	1.18E+03	**872.265**	2.68E+03	1.55E+03	1.66E+03
	$m = 400$	580.4533	629.9006	500.9419	765.3521	624.0737	613.2468	1.05E+03	1.51E+03	**466.0337**
	$m = 600$	569.6041	714.2734	554.3549	872.9069	748.9979	1.54E+03	1.14E+03	934.5108	**607.8929**
f_{14}	$m = 200$	**9.99E+07**	1.49E+08	2.74E+08	1.02E+08	1.69E+08	3.86E+08	1.09E+08	1.75E+08	4.28E+08
	$m = 400$	2.50E+08	1.79E+08	2.24E+08	1.72E+08	**1.32E+08**	1.79E+08	1.43E+08	1.44E+08	1.39E+08
	$m = 600$	4.78E+08	4.94E+08	4.49E+08	2.40E+08	1.90E+08	2.10E+08	1.83E+08	2.08E+08	**1.75E+08**
f_{15}	$m = 200$	1.02E+04	1.06E+03	1.20E+03	982.0615	**925.3112**	1.04E+03	1.05E+03	929.291	1.04E+03
	$m = 400$	1.02E+04	811.4653	727.3147	9.74E+03	648.6621	653.2338	486.5349	496.4844	**457.6811**
	$m = 600$	1.02E+04	1.02E+04	1.01E+04	9.65E+03	9.62E+03	9.73E+03	9.62E+03	9.61E+03	**9.57E+03**
f_{16}	$m = 200$	16.7135	**10.7095**	20.0514	20.6679	42.5893	22.1796	16.8435	24.7185	31.3355
	$m = 400$	1.82E-09	3.98E-13	4.09E-13	3.30E-13	2.77E-13	2.84E-13	3.13E-13	2.81E-13	**2.59E-13**

(continued on next page)

Table 3 (*continued*)

Fun	Swarm Size	$\varphi_2 = 0.1\varphi_3 = 0.1$	$\varphi_2 = 0.1\varphi_3 = 0.2$	$\varphi_2 = 0.1\varphi_3 = 0.3$	$\varphi_2 = 0.2\varphi_3 = 0.1$	$\varphi_2 = 0.2\varphi_3 = 0.2$	$\varphi_2 = 0.2\varphi_3 = 0.3$	$\varphi_2 = 0.3\varphi_3 = 0.1$	$\varphi_2 = 0.3\varphi_3 = 0.2$	$\varphi_2 = 0.3\varphi_3 = 0.3$
	$m = 600$	1.12E-04	1.43E-04	1.33E-04	5.63E-11	5.66E-11	5.58E-11	6.15E-13	**5.89E-13**	6.25E-13
f_{17}	$m = 200$	1.03E+05	1.00E+05	3.67E+05	9.34E+04	**8.87E+04**	3.18E+05	9.90E+04	1.17E+05	4.07E+05
	$m = 400$	1.81E+06	4.64E+05	4.23E+05	3.10E+05	1.60E+05	1.73E+05	2.70E+05	**1.46E+05**	1.58E+05
	$m = 600$	3.86E+06	3.93E+06	3.72E+06	6.61E+05	6.11E+05	5.79E+05	**3.54E+05**	3.87E+05	3.73E+05
f_{18}	$m = 200$	3.09E+03	3.31E+03	1.52E+04	3.41E+03	4.71E+03	5.61E+03	**2.77E+03**	3.32E+03	6.28E+03
	$m = 400$	2.02E+03	1.10E+03	1.52E+03	2.15E+03	1.99E+03	1.28E+03	3.26E+03	3.20E+03	**1.06E+03**
	$m = 600$	1.69E+04	1.31e+04	1.39E+04	1.37E+03	1.47E+03	**1.11E+03**	1.38E+03	1.78E+03	1.17E+03
f_{19}	$m = 200$	7.21E+06	1.71E+06	3.34E+06	3.12E+06	1.51E+06	2.11E+06	2.50E+06	**1.38E+06**	2.34E+06
	$m = 400$	8.65E+06	4.95E+06	3.20E+06	6.00E+06	4.24E+06	2.75E+06	4.81E+06	3.56E+06	**2.30E+06**
	$m = 600$	1.06E+07	1.08E+07	1.11E+07	6.89E+06	6.46E+06	7.21E+06	5.84E+06	**5.38E+06**	5.94E+06
f_{20}	$m = 200$	**1.78E+03**	1.90E+03	2.22E+03	1.91E+03	2.31E+03	3.30E+03	2.15E+03	2.58E+03	4.05E+03
	$m = 400$	1.09E+03	1.14E+03	**981.0304**	1.24E+03	1.19E+03	1.08E+03	1.29E+03	1.17E+03	1.08E+03
	$m = 600$	986.2091	986.2885	986.1739	1.04E+03	983.9315	1.04E+03	1.04E+03	1.03E+03	**982.0137**

6. Conclusion

In this paper, in order to satisfy the requirement of diversity and convergence at different stages of the evolution process, a novel three-phase co-evolutionary strategy is designed by merging multi-phase cooperative evolutionary technique into CSO (i.e. TPCSO). This new optimizer allows different update operations to adjust the diversity and convergence of the algorithm. Therefore, the TPCSO works as three-phase optimizer, where the first phase is to focus on the diversity of the swarm, the second phase is to provide a new optimal strategy based on superior particles of two sub-populations in order to further enhance the diversity and convergence of CSO, and the third phase is to promote the convergence of the algorithm. Some large scale benchmark functions are used to prove the optimization capacity of TPCSO. The results depicted demonstrate that TPCSO can obtain the relative mean value and standard deviation than CSO. TPCSO can provide a better optimization performance to obtain a high-quality solution in solving large-scale problems, and three-phase search strategy can effectively enhance global search ability of CSO. Overall, TPCSO has higher convergence, better optimization ability, better global search capability, and has good accuracy and time complexity.

In the future work, we will put more focus on the dynamic feedback mechanism of the population information, and implement rapid strategy adjustment by real-time detection of population information. TPCSO will further improve its global search capability and help other ECs as well.

CRediT authorship contribution statement

Chen Huang: Conceptualization, Methodology, Writing – original draft. **Xiangbing Zhou:** Writing – review & editing, Supervision. **Xiaojuan Ran:** Conceptualization, Software. **Yi Liu:** Supervision, Funding acquisition. **Wuquan Deng:** Resources, Software. **Wu Deng:** Supervision, Funding acquisition, Resources, Writing – review & editing.

Data availability

Data will be made available on request.

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.

Acknowledgments

The authors would like to thank all the reviewers for their constructive comments. This work was supported in part by the National Natural Science Foundation of China (Grant No. U2133205, U2033214, 61771087, 42271387), the Sichuan Science and Technology Program Project, China under Grant 2021YFS0407, 2022YFS0593, R21ZYZF0006, the China National Key R&D Program (2018YFB1601200), Scientific Research Fund Program of Liaoning Province Education Administration (LJKZ0214), and the Research Foundation for Civil Aviation University of China (Grant No. 2020KYQD123 and 3122022PT02).

References

[1] T.Y. Li, J.Y. Shi, W. Deng, Z.D. Hu, Pyramid particle swarm optimization with novel strategies of competition and cooperation, Appl. Soft Comput. 121 (2022) 108731.
[2] F. Liu H. Huang B. Li F.F. Xi A parallel learning particle swarm optimizer for inverse kinematics of robotic manipulator International Journal of Intelligent Systems 36 10 org/10.1002/int.22543. 2021, pp. 6101–6132 https://doi.
[3] W. Deng, J. Xu, X.Z. Gao, H. Zhao, An enhanced MSIQDE algorithm with novel multiple strategies for global optimization problems, IEEE Transactions on Systems, Man, and Cybernetics: Systems 52 (3) (2022) 1578–1587, https://doi.org/10.1109/TSMC.2020. 3030792.
[4] H. Zhao, J. Liu, H. Chen, et al, Intelligent diagnosis using continuous wavelet transform and gauss convolutional deep belief network, IEEE Transactions on Reliability (2022), https://doi.org/10.1109/TR.2022.3180273.
[5] W. Deng, L. Zhang, X. Zhou, et al, Multi-strategy particle swarm and ant colony hybrid optimization for airport taxiway planning problem, Information Sciences 612 (2022) 576–593, https://doi.org/10.1016/j.ins.2022.08.115.
[6] L. Ma, S. Cheng, Y. Shi, Enhancing learning efficiency of brain storm optimization via orthogonal learning design, IEEE Transactions on Systems, Man, and Cybernetics: Systems 51 (11) (2021) 6723–6742, https://doi.org/10.1109/TSMC.2020.2963943.
[7] L. Ma, N. Li, Y. Guo, X. Wang, S. Yang, M. Huang, H. Zhang, Learning to optimize: reference vector reinforcement learning adaption to constrained many-objective optimization of industrial copper burdening system, IEEE Transactions on Cybernetics (2021) 1–14, https://doi.org/10.1109/TCYB.2021.3086501.
[8] X. Zhou, H. Ma, J. Gu, H. Chen, W. Deng, Parameter adaptation-based ant colony optimization with dynamic hybrid mechanism[J], Engineering Applications of Artificial Intelligence 114 (2022) 105139.
[9] Y. Yang, H. Chen, A.A. Heidari, et al, Hunger games search: Visions, conception, implementation, deep analysis, perspectives, and towards performance shifts, Expert Systems with Applications 177 (2021), https://doi.org/10.1016/j.eswa.2021.114864 114864.
[10] R. Yao, C. Guo, W. Deng, H. Zhao, A novel mathematical morphology spectrum entropy based on scale-adaptive techniques, ISA transactions 126 (2022) 691–702.
[11] H.Y. Chen, F. Miao, Y.J. Chen, Y.J. Xiong, T., Chen.A hyperspectral image classification method using multifeature vectors and optimized KELM, IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens 14 (2021) 2781–12279.

[12] Z. Ren, X. Han, X. Yu, R. Skjetne, B. Johan, S. Leira, M. Zhu, Data-driven simultaneous identification of the 6DOF dynamic model and wave load for a ship in waves, Mech. Syst. Signal Process. 184 (2023) 109422.
[13] A.H. Ben, Y.W. Ben, O. Ayadi, F. Masmoudi, Design of efficient multiobjective binary PSO algorithms for solving multi-item capacitated lot-sizing problem, International Journal of Intelligent Systems 37 (2) (2022) 1723–1750, https://doi.org/10.1002/int.22693.
[14] W. Deng, S. Shang, X. Cai, et al, An improved differential evolution algorithm and its application in optimization problem, Soft Computing 25 (7) (2021) 5277–5298, https://doi.org/10.1007/s00500-020-05527-x.
[15] W. Deng, H. Ni, Y. Liu, et al, An adaptive differential evolution algorithm based on belief space and generalized opposition-based learning for resource allocation, Applied Soft Computing 127 (2022), https://doi.org/10.1016/j.asoc.2022.109419 109419.
[16] Y. Song, X. Cai, X. Zhou, et al, Dynamic hybrid mechanism-based differential evolution algorithm and its application, Expert Systems with Applications 213 (2023), https://doi.org/10.1016/j.eswa.2022.118834 118834.
[17] J. Kennedy, R. Eberhart, Particle swarm optimization, Proc. Int. Conf. Neural Networks 4 (1995) 1942–1948, https://doi.org/10.1109/ICNN.1995.488968.
[18] W.B. Liu, Z.D. Wang, Y. Yuan, et al, A novel sigmoid-function-based adaptive weighted particle swarm optimizer, IEEE Transactions on Cybernetics 51 (2) (2021) 1085–1093, https://doi.org/10.1109/TCYB.2019.2925015.
[19] F. Liu, H. Huang, B. Li, F. Xi, A parallel learning particle swarm optimizer for inverse kinematics of robotic manipulator, International Journal of Intelligent Systems. 36 (2021) 6101–6132, https://doi.org/10.1002/int.22543.
[20] F. Han, J.S. Zhu, Improved particle swarm optimization combined with backpropagation for feedforward neural networks, International Journal of Intelligent Systems 28 (3) (2013) 271–288, https://doi.org/10.1002/int.21569.
[21] W.N. Chen, J. Zhang, Y. Lin, E. Chen, Particle swarm optimization with an aging leader and challengers, IEEE Transactions on Evolutionary Computation 17 (2) (2013) 241–258, https://doi.org/10.1109/TEVC.2011.2173577.
[22] R. Cheng, Y. Jin. A social learning particle swarm optimization algorithm for scalable optimization. Information Sciences, 291 (2015) 43-60. http://dx.doi.org/ 10.1016/j.ins.2014.08.039.
[23] Z.H. Zhan, J. Zhang, Y. Li, et al, Adaptive particle swarm optimization, IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics 39 (6) (2009) 1362–1381, https://doi.org/10.1109/TSMCB.2009.2015956.
[24] M. Dadvar, H. Navidi, H.H.S. Javadi, et al, A cooperative approach for combining particle swarm optimization and differential evolution algorithms to solve single-objective optimization problems, Applied Intelligence 52 (4) (2022) 4089–4108, https://doi.org/10.1007/s10489-021-02605-x.
[25] N. Zeng, Z. Wang, W. Liu, H. Zhang, et al, A dynamic neighborhood-based switching particle swarm optimization algorithm, IEEE Transactions on Cybernetics 52 (9) (2022) 9290–9301, https://doi.org/10.1109/TCYB.2020.3029748.
[26] N. Lynn, P.N. Suganthan, Ensemble particle swarm optimizer, Applied Soft Computing 55 (2017) 533–548, https://doi.org/10.1016/j.asoc.2017.02.007.
[27] Z.J. Wang, Z.H. Zhan, S. Kwong, H. Jin, et al, Adaptive granularity learning distributed particle swarm optimization for large-scale optimization, IEEE Transactions on Cybernetics 51 (3) (2021) 1175–1188, https://doi.org/10.1109/TCYB.2020.2977956.
[28] H. Lu, P. Sriyanyong, Y.H. Song, T. Dillon, et al, Experimental study of a new hybrid PSO with mutation for economic dispatch with non-smooth cost function, International Journal of Electrical Power & Energy Systems 32 (9) (2010) 921–935, https://doi.org/10.1016/j.ijepes.2010.03.001.
[29] K. Chaitanya, D.V.L.N. Somayajulu, P. Radha Krishna. Memory-based approaches for eliminating premature convergence in particle swarm optimization. Applied Intelligence, 51(7) (2021) 4575-4608. https://doi.org/10.1007/s10489-020-02045-z.
[30] X. Zhao, J.N. Hwang, Z. Fang, G. Wang, Gradient-based adaptive particle swarm optimizer with improved extremal optimization, Applied Intelligence, 48 (12) (2018) 4646–4659, https://doi.org/10.1007/s10489-018-1228-9.
[31] B. Yan, Z. Zhao, Y. Zhou, W. Yuan, et al, A particle swarm optimization algorithm with random learning mechanism and Levy flight for optimization of atomic clusters, Computer Physics Communications 219 (2017) 79–86, https://doi.org/10.1016/j.cpc.2017.05.009.
[32] R. Cheng, Y. Jin, A competitive swarm optimizer for large scale optimization, IEEE Transactions on Cybernetics 45 (2) (2015) 191–204, https://doi.org/10.1109/TCYB.2014.2322602.
[33] M.A. Potter, K.A.D. Jong. A cooperative coevolutionary approach to function optimization. In: Proc. International conference on parallel problem solving from nature, (1994) pp.249-257. https://doi.org/10.1007/3-540-58484-6_269.
[34] Z. Yang, K. Tang, Y. Xin, Large scale evolutionary optimization using cooperative coevolution, Information Sciences 178 (15) (2008) 2985–2999, https://doi.org/10.1016/j.ins.2008.02.017.
[35] Z. Yang, K. Tang, X. Yao. Multilevel cooperative coevolution for large scale optimization. In: Proc. IEEE Congress on Evolutionary Computation, (2008) pp.1663-1670. https://doi.org/10.1109/CEC.2008.4631014.
[36] X. Li, X. Yao. Tackling high dimensional nonseparable optimization problems by cooperatively coevolving particle swarms. In: Proc. IEEE Congress on Evolutionary Computation. (2009) pp.1546-1553. https://doi.org/10.1109/CEC.2009.4983126.
[37] X. Li, Y. Xin, Cooperatively coevolving particle swarms for large scale optimization, IEEE Transactions on Evolutionary Computation 16 (2) (2012) 210–224, https://doi.org/10.1109/TEVC.2011.2112662.
[38] M.N. Omidvar, X. Li, Y. Mei, X. Yao, Cooperative co-evolution with differential grouping for large scale optimization, IEEE Transactions on Evolutionary Computation 18 (3) (2014) 378–393, https://doi.org/10.1109/TEVC.2013.2281543.
[39] M.N. Omidvar, M. Yang, Y. Mei, X. Li, X. Yao, DG2: a faster and more accurate differential grouping for large-scale black-box optimization, IEEE Transactions on Evolutionary Computation 21 (6) (2019) 929–942, https://doi.org/10.1109/TEVC.2017.2694221.
[40] X. Ma, Z. Huang, X. Li, L. Wang, et al, Merged differential grouping for large-scale global optimization, IEEE Transactions on Evolutionary Computation (2022), https://doi.org/10.1109/TEVC.2022.3144684.
[41] M. Yang, A. Zhou, C. Li, X. Yao, An efficient recursive differential grouping for large-scale continuous problems, IEEE Transactions on Evolutionary Computation 25 (1) (2021) 159–171, https://doi.org/10.1109/TEVC.2020.3009390.
[42] M. Hasanzadeh, M.R. Meybodi, M.M. Ebadzadeh, Adaptive cooperative particle swarm optimizer, Applied Intelligence 39 (2013) 397–420, https://doi.org/10.1007/s10489-012-0420-6.
[43] S.Z. Zhao, J.J. Liang, P.N. Suganthan, M.F. Tasgetiren, Dynamic multi-swarm particle swarm optimizer with local search for large scale global optimization, In Proc, IEEE Congress on Evolutionary Computation (2008) 3845–3852, https://doi.org/10.1109/CEC.2008.4631320.
[44] L. Ma, M. Huang, S.X. Yang, R. Wang, X.W. Wang, An adaptive localized decision variable analysis approach to large-scale multiobjective and many-objective optimization, IEEE Transactions on Cybernetics 52 (7) (2022) 6684–6696, https://doi.org/10.1109/TCYB.2020.3041212.
[45] Q. Yang, W.N. Chen, J.D. Deng, et al, A level-based learning swarm optimizer for large-scale optimization, IEEE Transactions on Evolutionary Computation 22 (4) (2018) 578–594, https://doi.org/10.1109/TEVC.2017.2743016.
[46] K. Tang, X. Li, P. Suganthan, Z. Yang, T. Weise, Benchmark functions for the CEC,, special session and competition on large scale global optimization, Inspired Comput. Appl. Lab., Univ. Sci. Technol. China, Hefei, China, Rep, Nat, 2010, p. 2009.
[47] X. Li, K. Tang, M.N. Omidvar, Z. Yang, K. Qin, Benchmark functions for the CEC,, special session and competition on large scale global optimization, Comput. Mach. Learn. Subpopulation, RMIT Univ., Melbourne, VIC, Australia, Rep, Evol, 2013, p. 2013.
[48] P. Mohapatra, K.N. Das, S. Roy, A modified competitive swarm optimizer for large scale optimization problems, Applied Soft Computing 59 (2017) 340–362, https://doi.org/10.1016/j.asoc.2017.05.060.