Expert Systems With Applications 203 (2022) 117444

Contents lists available at ScienceDirect

Expert Systems With Applications

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

A two-stage cooperative scatter search algorithm with multi-population hierarchical learning mechanism

Fuqing Zhao [a,*], Gang Zhou [a], Ling Wang [b], Tianpeng Xu [a], Ningning Zhu [a], Jonrinaldi [c]

[a] School of Computer and Communication, Lanzhou University of Technology, Lanzhou 730050, China
[b] Department of Automation, Tsinghua University, Beijing 10084, China
[c] Department of Industrial Engineering, Universitas Andalas, Padang 25163, Indonesia

ARTICLE INFO

Keywords:
Scatter search
Multi-population
Hierarchical learning mechanism
Interactive learning strategy
Pattern search

ABSTRACT

Scatter search (SS) is a population-based metaheuristic algorithm, which has been proved high efficiency and effective optimizer for complex continuous real value problems. A two-stage cooperative SS guided with the multi-population hierarchical learning mechanism (TCSSMH) to overcome the slow convergence speed of the original SS is proposed. Three strategies are applied to the original SS. Firstly, TCSSMH adopts an adaptive two-way selection search strategy based on the elite reference set (*RefSet*), which is elite-oriented and ensures the quality of the population. Secondly, the multi-group hierarchical learning mechanism is embedded in the updating process of the *RefSet*, and the population of the candidates is divided into three levels including excellent candidates, medium candidates, and inferior candidates according to the fitness value of the function. These three subpopulations cooperate to balance the exploration and exploitation ability of the algorithm in the process of evolution. Finally, each subpopulation adopts an interactive learning strategy to increase the diversity of the population and avoid premature convergence of solutions. The optimum of each subpopulation with high accuracy is obtained by the pattern search (PS) optimization. The stronger search ability and higher search efficiency of these additional proposed strategies are verified by extensive experiments. The TCSSMH algorithm is tested on the CEC2017 benchmark test suite and practical engineering problems. The experimental results show that the TCSSMH algorithm is superior to other state-of-the-art algorithms in global search ability and convergence on the benchmark problems.

1. Introduction

Global optimization problems widely exist in the field of natural science and engineering. The goal of optimization is to find a group of decision variables that make the objective function gain the minimum or maximum value. In recent years, intelligent optimization algorithms have been widely popularized and applied in various fields, such as production scheduling (Paithankar & Chatterjee, 2019), smart transportation (Singh & Singh, 2021), automatic control (Çelik, 2020), and intelligent medical treatment (Chen, He, Zeng, Li, & Liang, 2021). The practical engineering problems in the real world have the characteristics of high complexity, high computational cost, non-linearity, multi-constraints, and multivariable. The theoretical optimal solutions of these problems are difficult to find. The deterministic optimization algorithms include the gradient descent method, the mountain climbing method, the Newton method, the quasi-Newton method, etc. In the case of discontinuous and non-derivable objective functions, deterministic optimization algorithms are generally unable to address these problems and are easy to fall into local optimization. In addition, the optimization problem becomes more complex with the increase of scale, and the satisfactory solution is difficult to be solved by the traditional optimization method in a reasonable time. However, the intelligent optimization algorithm is an important approach to obtain the optimal solution because of its strong searching ability.

In the past few decades, various metaheuristics have been proposed. Metaheuristic algorithms are divided into four categories (Del Ser et al., 2019), including evolution-based methods, swarm intelligence optimization algorithms, physical rule-based methods, and human behavior-based methods. The evolutionary algorithm simulates the process of biological evolution in nature, which mainly includes genetic algorithm (GA) (Holland, 1992), evolutionary strategy (ES) (Li & Zhang, 2018), evolutionary programming (EP) (Yao, Liu, & Lin, 1999), genetic

* Corresponding author.
E-mail addresses: fzhao2000@hotmail.com (F. Zhao), wangling@tsinghua.edu.cn (L. Wang), xutp@lut.edu.cn (T. Xu), jonrinaldi@eng.unand.ac.id (Jonrinaldi).

https://doi.org/10.1016/j.eswa.2022.117444
Received 20 December 2021; Received in revised form 27 April 2022; Accepted 27 April 2022
Available online 30 April 2022

programming (GP) (Angeline, 1994), and differential evolution (DE) (Das, Mullick, & Suganthan, 2016; Zhao, Zhao, Wang, & Song, 2020). The swarm intelligence optimization algorithm is a bionic algorithm based on biological behavior, which has the ability to solve complex tasks through interaction between individuals, typical algorithms include particle swarm optimization (PSO) (Espitia & Sofrony, 2018), ant colony optimization (ACO) (Sreelaja, 2021), artificial bee swarm algorithm (ABC) (Xue, Jiang, Zhao, & Ma, 2018), fruit fly optimization algorithm (FOA) (Zhao, Ding, Wang, Cao, & Tang, 2021), firefly algorithm (FA) (Chen, He et al., 2021; Chen, Huang et al., 2021), cuckoo search algorithm (CS)(Cuong-Le et al., 2021), bat algorithm (BA) (Chen, Huang, Zeng, Lu, & Yang, 2021), gray wolf optimization algorithm (GWO)(Yu, Xu, & Li, 2021), whale optimization algorithm (WOA) (Chen, Li, & Yang, 2020), etc. The Swarm intelligence optimization algorithms focus on the collaboration among individuals in a group, while evolutionary algorithms focus on the evolution of individuals. The method based on physical rules refers to the simulation of physical phenomena in nature such as inertia force, gravity, light projection, and cooling, etc. Typical algorithms include water wave optimization (WWO) (Zhao, Zhang, et al., 2020), gravity search algorithm (GSA) (Wang, Gao, Yu, Cai, & Wang, 2021a), charging system search (CSS) (D'Ambrosio, Spiller, & Curti, 2020), black hole search algorithm (BHA) (Pashaei & Aydin, 2017), thermal exchange optimization algorithm (TEO) (Kaveh & Dadras, 2017), and water cycle algorithm (WCA) (Chen, Wang, Dong, & Wang, 2020), etc. There are also some intelligent search algorithms inspired by human behavior, such as the teaching–learning based optimization algorithms (TLBO) (Ma, Zhang, Song, & Chen, 2021), harmony search (HS)(Talaei, Rahati, & Idoumghar, 2020), socio evolution and learning optimization algorithm (SELO) (Kumar, Kulkarni, & Satapathy, 2018), and cultural evolution algorithm (CEA) (Maheri, Jalili, Hosseinzadeh, Khani, & Miryahyavi, 2021), etc.

SS is an evolutionary method that has been successfully applied to various kinds of complex optimization problems (Pardo, Argüeso-Alejandro, González, Banga, & Doallo, 2020). SS is mainly to build, maintain and evolve a set of solutions in the process of evolution. Unlike other meta-heuristic algorithms, SS does not utilize mutation and crossover but generates new solutions by combining existing elements. These high-quality solutions define a set, known as *RefSet*. In most modern metaheuristic algorithms, the typical population is composed of 100 uniformly and randomly generated individuals, or even more, and then these solutions are used to generate new individuals. These algorithms do not utilize small population size because it usually leads to premature convergence and diversity is difficult to maintain. In contrast, the *RefSet* in SS has only 10 solutions. Two or three individuals are selected from the *RefSet* in a systematic way to combine to produce a new solution. The method of generating new solutions by combination is highly dependent on the *RefSet* and has some limitations. In the original SS improvement stage, the traditional local search method is used to improve each candidate solution generated by the combination method. The experimental results indicate that this method is not effective in dealing with high-dimensional optimization problems, and consumes a lot of computing resources. At the same time, SS lacks the treatment of precocious convergence in the process of evolution, which leads to the loss of population diversity. Therefore, it is worth studying to design an effective strategy to update the *RefSet*. SS algorithm has been widely applied to solve practical optimization problems due to its advantages of fewer parameters and flexible algorithm framework (Soman & Patil, 2020).

To improve the global performance of SS, the main framework follows SS, and the multi-population hierarchical learning mechanism is used to update *RefSet* to generate new solutions. In order to balance the ability of exploration and exploitation, excellent individuals with different characteristics are selected to guide population evolution in two stages. The first stage focuses on global search ability, and the second stage focuses on local search ability. The main contributions of this paper are as follows:

(1) In the *RefSet* update stage, the multi-population hierarchical learning mechanism guided by elite individuals achieves a balance between diversity and convergence rate.
(2) An interactive learning information sharing strategy is proposed, in which the inferior individuals in each subpopulation learn from the optimal individuals in the adjacent subgroups and the global optimal individuals. This mechanism is conducive to information sharing and increases the diversity of the population.
(3) An adaptive bi-directional selection search strategy, which is based on the elite *RefSet*, is adopted to improve the global exploration ability and correctly guide the evolution direction of the proposed TCSSMH.
(4) The pattern search algorithm is employed to improve the exploitation capability and convergence speed without the loss of population diversity.

The rest of this paper is organized as follows. The related literature is reviewed in Section 2. Section 3 presents a framework of the canonical SS. Section 4 presents TCSSMH and analyzes the complexity of the algorithm. Then, the experimental results and detailed analysis are presented in Section 5. Section 6 discusses the ability of TCSSMH to address practical engineering optimization problems and analyzes the experimental results. The conclusions and the outlook of the work are given in Section 7.

2. Literature review

SS was first introduced in 1977 (Glover, 1977) as a heuristic for integer programming. Glover defined the template of SS in 1988 (Glover, 1998) and proposed the implementation method of key parts. Laguna (Laguna & Marti, 2003) further expanded the algorithm in 2003 to form the basic framework of SS, and completely described the algorithm in C. The SS is a population-based algorithm that uses an intelligent "Scatter-converge" iterative mechanism to generate high-quality and diverse solutions in the *RefSet*. The global optimal solution or satisfactory solution of the problem is obtained by the subset merging method and the *RefSet* updating method. It is worth noting that each mechanism in the framework of SS can be implemented in a variety of ways, which makes the algorithm rapidly attain a satisfactory solution and avoid falling into the local optimal solution prematurely. At present, SS has been widely applied to various fields, such as software testing (Liu, Huang, Yang, Hao, & Wang, 2019), bioinformatics (Remli, Mohamad, Deris, & Samah, 2019), vehicle routing problems (Soman & Patil, 2020), etc.

SS is one of several metaheuristic algorithms widely concerned by researchers in recent years. Experimental results show that it performs well in solving some combinatorial optimization and continuous optimization problems. It can deal with various complex practical problems very well (Kalra et al., 2021). However, it still lacks a balance between exploration and exploitation, especially when it comes to optimizing complex multimodal problems. We introduce a multi-population hierarchical learning mechanism, adaptive bi-directional selection search strategy, and a cooperative interactive learning mechanism in the original SS. A new SS variant TCSSMH is proposed to improve the convergence speed, avoid falling into the local optimal solution, and improve the balance between exploration and development, so as to better solve the complex practical optimization problems. These three strategies improve the performance of SS, and the TCSSMH implementation achieves better performance compared to other contrasting algorithms. We apply T to engineering design problems for testing. The experimental results indicate that it can be used as an effective tool to solve engineering design problems. According to the literature review, the improvement of SS is mainly divided into four categories, including improved component, hybrid and multi-objective.

(1) Improve the components of SS

To enhance the performance of SS, an effective strategy is to improve SS components. In (Remli, Deris, Mohamad, Omatu, & Corchado, 2017), Remli et al. proposed an enhanced SS combined opposition-based learning (SSCOL) to solve the problem of large-scale parameter estimation in the biochemical system dynamics models. Important improvements are made to the *RefSet*, which is the core part of SS, including *RefSet* construction, *RefSet* combination, and *RefSet* enhancement. The *RefSet* is further combined with opposition-based learning to improve optimization performance. Experimental results show that the proposed strategy significantly improves the performance of global optimization and the efficiency of parameter estimation. On the basis of SSCOL, Remli et al. (2019) proposed a collaborative enhanced scatter search based on an opposite learning scheme (CeSSOL), which was used for parameter estimation in large-scale biological models. Experimental results show that the proposed strategy has better performance. In addition, the results show that this method can not only shorten the calculation time but also effectively find the near-optimal solution. In, an enhanced scattering search method using the control vector parameters (CVP) method for global dynamic optimization of nonlinear processes is proposed. This new algorithm provides a good balance between robustness and efficiency on the global stage, combined with a local search process to accelerate convergence to the optimal solution. The results showed that the proposed algorithm performed better than the other methods. A continuous SS (CSS) algorithm integrating several combination methods and improved strategies was proposed in (Herrera, Lozano, & Molina, 2006) to strengthen the best solutions for complex problems. The proposed algorithm was tested on 45 functions and three real-world problems, and the results showed that it had better performance for simple and complex problems. Hvattum, Duarte, Glover, and Martí (2013) studied the performance of SS combined with eight different improved methods to solve the global optimization problem. The proposed algorithm was compared with 38 test problems. The experimental results show that the improved method is the key factor to determine the relative performance of decentralized search, and has good performance in solving the unconstrained global optimization problem. Laguna and Martí (2005) designed various alternative mechanisms to update the *RefSet* in SS to maintain the balance between diversity and intensification. The performance of SS was validated on 40 different test questions. The results indicate that SS can generate the best solution for most problems. Hakli and Ortacay (2019) presented an improved SS for the uncapacitated facility location problem (UFL). The proposed algorithm adopts integration and other different crossover technologies to enhance the global search ability of SS. At the same time, the mutation operation of the optimal solution improves the local search ability. The experimental results indicate that the optimal solutions are attained for 13 of the 15 different UFL problems.

(2) Hybrid SS algorithms

The advantages of other heuristic algorithms are integrated into the SS algorithm by the hybrid algorithm to make up for its shortcomings and make the algorithm gain better performance. The SS has various applications in both continuous and discrete optimization problems. In (Abd Alradha Alsaidi, Muhsen, & Ali, 2020), Alsaidi et al. applied an improved SS based on the meerkat clan algorithm (MCA-SS) to solve NP-hard problems. Experimental results show that the proposed hybrid algorithm has better performance than the existing ones. Li and Tian (2015) integrated DE into the *RefSet* update process to solve the global optimization problems, in which the adaptive mechanism is employed for selecting the candidate mutation operator. The simulation results show that the hybrid algorithm achieves good performance in most benchmark tests. Sarakhsi, Fatemi Ghomi, and Karimi (2016) proposed a hybrid algorithm of SS and Nelder-Mead algorithms to optimize joint economic lot-sizing problems. The simulation results displayed its great performance for a bounded or constrained problem.

Sagheer, Sadiq, and Ibrahim (2012) implement a hybrid algorithm based on the bee algorithm and SS to solve the traveling salesman problem (TSP). The bee algorithm is added to the improved method to enhance the searchability of SS. This method (Bee-SS) is superior to the original SS in solving the optimal solution. However, the computing time of this new hybrid algorithm is longer than that of SS. Al-Obaidi (Sadiq, 2013) proposed an enhanced SS algorithm based on the cuckoo algorithm. The introduction of CS into the Refset update stage provides diversity for the new solutions generated. The test is carried out on solving the TSP problem. The experimental results indicate that the enhanced SS is comparable to the original one. Compared with the SS of, the average fitness value is improved by 23%.

(3) Multi-objective SS

SS has achieved good performance in optimizing multi-objective problems (Kalra et al., 2021). Nebro et al. (2008) propose an archive-based hybrid SS that optimizes multi-objective problems (AbYSS). The ABYSS utilizes an external archive to store non-dominated solutions found during the search. The overall framework adopts the SS structure while applying mutation and crossover operators in evolutionary algorithms. The experimental results on the standard test set show that AbYSS is competitive in terms of solution diversity, and outperforms the selected comparison algorithms on most of the selected 33 test problems. Kerkhove and Vanhoucke (2017) proposed and implemented a novel multi-objective scatter-search heuristic method utilizing parallelization. Using parallelization by creating a single population, and then using separate processors to search these populations, different populations evolve in different directions. Experimental results demonstrate that the algorithm outperforms the variable weighted sum method in terms of computational speed and quality of the solution. Bajestani, Rabbani, Rahimi-Vahed, and Baharian Khoshkhou (2009) proposed a new multi-objective SS (MOSS) for solving dynamic cell formation problems. Minimizing the sum of total cell load changes and miscellaneous costs was also considered. Computational results show that MOSS is superior to SPEA-II and NSGA-II.

The multi-population method is one of the most effective methods to maintain population diversity, and it is helpful for the optimization algorithm to find the global optimal solution efficiently (Ma et al., 2019). At present, a large number of studies on a variety of group optimization algorithms show that the performance of multi-population optimization algorithm is usually better than that of single population optimization algorithm in solving numerical optimization problems and practical application problems. The multi-population method divides the population into multiple subpopulations located in different search spaces. These subpopulations search multiple different regions to find the candidate region of the optimal solution at a faster speed. Sun and Chen (2021) proposed a multi-population WOA for high-dimensional optimization. The population is divided into better and worse groups to improve exploitation performance and exploration performance respectively. The simulation results on 30 high-dimensional benchmark functions show that the algorithm has good performance in high-dimensional optimization problems. Liu et al. (2021) presented a multi-swarm Salp swarm algorithm with the chaos-assisted exploitation strategy, which maintained multiple swarm populations that applied three sub-strategies to enhance the global exploration capabilities of the algorithm. The simulation results on 35 benchmark problems showed the reliability and efficiency in solving complex optimization problems.

Wei et al. (2020) proposed a particle swarm optimization algorithm based on multiple adaptive strategies. The population is divided into multiple sub-populations to maintain diversity. The learning sample adaptive strategy and the population size adaptive strategy balance the exploration and exploitation capabilities well. Simulation results on CEC2013 and CEC2017 test sets demonstrated the effectiveness of the proposed algorithm. In TAPSO (Xia, Gui, Yu, et al., 2020), three archives are used to design effective learning models and select suitable paradigms, and TAPSO has higher performance in solution accuracy and convergence speed on 30 benchmark functions and four practical applications. Xia, Gui, He, et al. (2020) proposed an extended PSO (XPSO) with three strategies to improve the performance of PSO in solving optimization problems. This algorithm has been tested on the benchmark functions of the CEC'13 test suite, and the results show improved accuracy and convergence speed of the algorithm.

The cooperative learning strategy is to decompose the complex optimization problem into multiple problems. Each subproblem is solved by a separate sub-population. The complete solution of the problem is obtained by cooperative coevolution between sub-populations. The cooperative coevolution framework has been adopted by various metaheuristic algorithms and successfully applied to various optimization problems in the real world (Liaw, 2021). Zhao et al. (2021) proposed a hierarchical guided policy-assisted fruit fly optimization algorithm with a collaborative learning mechanism (HGCLFOA). The algorithm balances exploration and exploitation through cooperation between elite and disadvantaged populations. The simulation results on the benchmark functions showed that the proposed algorithm provided much better performance than the other algorithms. Yang, Zhang, Yu, Li, and Tang (2022) proposes an NCS-friendly Cooperative Coevolution framework for large-scale search spaces. Validating the proposed algorithm on 10 popular Atari games, the experimental results show that the amplified NCS can reduce the computation time by 50% compared to the three state-of-the-art DRL methods. Furthermore, in the process of evolution, each particle is in a different evolution state, and the particles can be divided into different grade states according to the fitness value, and the individuals in different grade states have different abilities to search and use the search space. Therefore, the population divided into different levels can maintain the diversity of the population and achieve a reasonable balance between exploration and development. Wang, Gao, Yu, Cai, and Wang (2021b) proposed a gravitational search algorithm with a hierarchical structure and a distributed framework (DGSA). The distributed framework divides the entire population into several sub-populations, and a three-level hierarchy guides the search for each subpopulation. Experimental results on the standard test set show that DGSA has higher search performance and efficiency compared to some variants of GSA. In the MLGSA (Wang, Gao, Member, Zhou, & Yu, 2021), a multi-layer gravitational search algorithm is proposed, which has a population layer, an iterative optimal layer, an individual optimal layer, and a global optimal layer. Among the layers, the interaction of the three layers effectively enhances the exploration and development capabilities of their populations. The simulation results of 29 benchmark problems and 22 practical optimization problems indicate the superiority of this method. In the LLSO (Yang, Member, Chen, & Deng, 2017), a level-based PSO learning strategy is proposed, which divides particles into different levels, treats them differently, and utilizes the two main particles in the current swarm to guide the learning of particles to find the global optimal solution. Comparisons with several state-of-the-art algorithms on two sets of widely used large-scale benchmark functions confirm that the proposed optimizer is competitive in terms of solution quality and computational efficiency.

In recent years, people have done extensive research work and put forward many variants of the SS algorithm. These algorithms attempt to balance exploration and development capabilities by controlling parameters, mixing with other optimization techniques, designing new strategies in the process of evolution, and improving the convergence of the solution. However, with the increasing complexity of optimization problems, the existing algorithms still can not guarantee the diversity and effectiveness of solutions in reality. In this study, the multi-population hierarchical learning mechanism is introduced, and the elite Refset information is used to guide the *RefSet* update process of SS to generate new solutions. This elite guidance update *RefSet* mechanism breaks the original framework of SS. The Multi-population cooperation strategy effectively improves the optimization performance of the Algorithm. At present, there is no latest report about using the multi-population cooperation strategy to improve the performance of the SS algorithm.

3. Canonical SS algorithm

The SS is a population-based meta-heuristic algorithm. The framework of SS consists of five system sub-methods, including diversification generation, solution improvement, Refset update, subset generation, and solution combination methods (Glover, 1998). The whole framework is flexible, and each basic method in the framework can be implemented and replaced by different methods according to the complexity of practical problems. The basic process of SS is shown in Algorithm 1.

Algorithm 1: The framework of SS

```
Input: Population size (PSize), ReferSet size (b)
Output: The best of solutions;
Initialize the population by DGM;
Apply the IM to enhance the population;
Apply the RGM to build the initial RefSet, Refset = {x^1, x^2, ···, x^b};
while the stop condition is not satisfied do
    Create subset NewSubsets from Refset using SGM;
    Apply the SCM to combine each subset into a new solution;
    Apply the IM to optimize each combined solution;
    Update RefSet with RGM;
end while
Update optimal solution.
```

(1) A Diversification Generation Method (DGM) that uses an arbitrary trial solution (or seed solution) as input to generate a set of different trial solutions. Details of the method are as follows. D is the dimension, LB_j and UB_j are the lower bound and upper bound of each dimension j. The range of each dimension $[LB_j, UB_j]$ $(j = 1, 2, \cdots, D)$ is divided into K equal sub-intervals. This process is constructed in two steps. First, randomly select a sub-range. The probability of selecting a sub-range is inversely proportional to its frequency count. A value is then randomly generated in the selected sub-range.

(2) An Improved Method (IM) of converting a trial solution into one or more enhanced trial solutions. Nelder and Mead's simplex method (Herrera et al., 2006) is applied to each solution generated by DGM as a local improvement method.

(3) A Refset Updating Method (RGM) is used to build and maintain a set of reference sets containing high-quality solutions and diverse solutions. The size of the Refset is denoted by $b = b_1 + b_2 = |RefSet|$. The best b_1 solutions are selected from P (P is the initial solution generated by DGM and IM) to construct the initial Refset. These solutions are added to *Refset* and removed from P. Then calculate the minimum Euclidean distance $d_{min}(x)$ between each solution x in $P - RefSet$ to and the solutions y currently in RefSet. The largest of these minimum distances is selected, added to the Refset, and deleted from P, and the minimum distance is updated. This process is repeated b_2 times, where $b_2 = b - b_1$. Then the formula of $d_{min}(x)$ is given as follows.

$$d_{min}(x) = \min_{y \in RefSet} \{d(x, y)\} \tag{1}$$

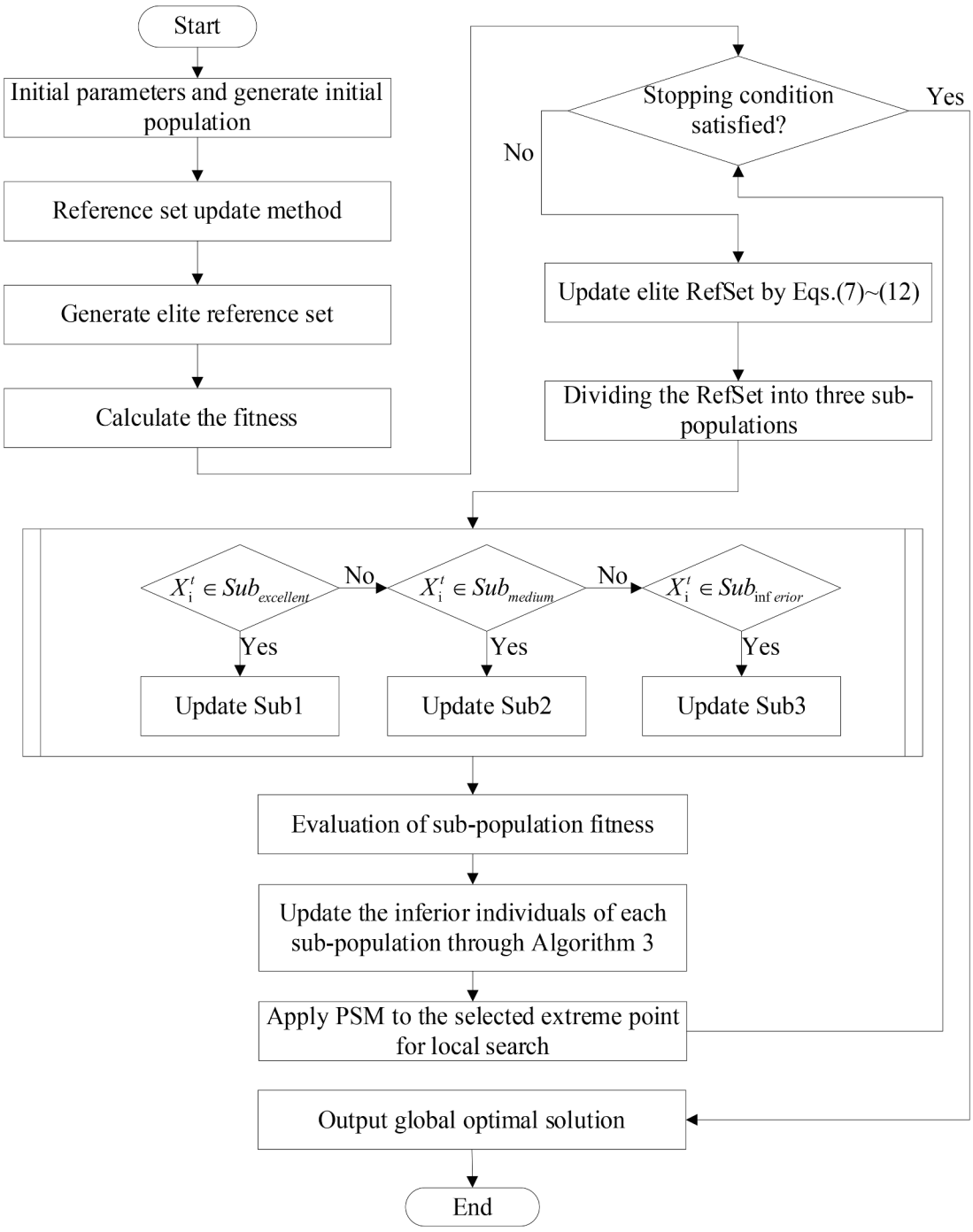


Fig. 1. The flow chart of proposed TCSSMH.

$$d(x,y) = \sqrt{\sum_{i=1}^{n}(x_i - y_i)^2} \tag{2}$$

where $d(x,y)$ is the Euclidean distance between x and y.

(4) A Subset Generation Method (SGM) that operates on a Refset to generate a subset of its solution to provide input for creating a composite solution. The subset size is normally set to 2. The number of reference solutions depends on the strategy implemented in RGM when the SGM is executed.

(5) A Solution Composition Method (SCM) for transforming a given subset of solutions produced by the SGM into one or more combined solutions. The SCM is usually determined by the actual specific problem. The SCM is a problem-specific mechanism because it is directly related to the solution representation. In the classic SS template (Glover, 1998), three trial solutions were created in each combination of two reference solutions represented by $x^{'}$ and $x^{''}$ The specific formula is as follows.

$$C_1 : X_1 = x^{'} - d \tag{3}$$

$$C_2 : X_1 = x^{'} + d \tag{4}$$

$$C_3 : X_1 = x^{''} + d \tag{5}$$

$$d = r^{*}\frac{x^{''} - x^{'}}{2} \tag{6}$$

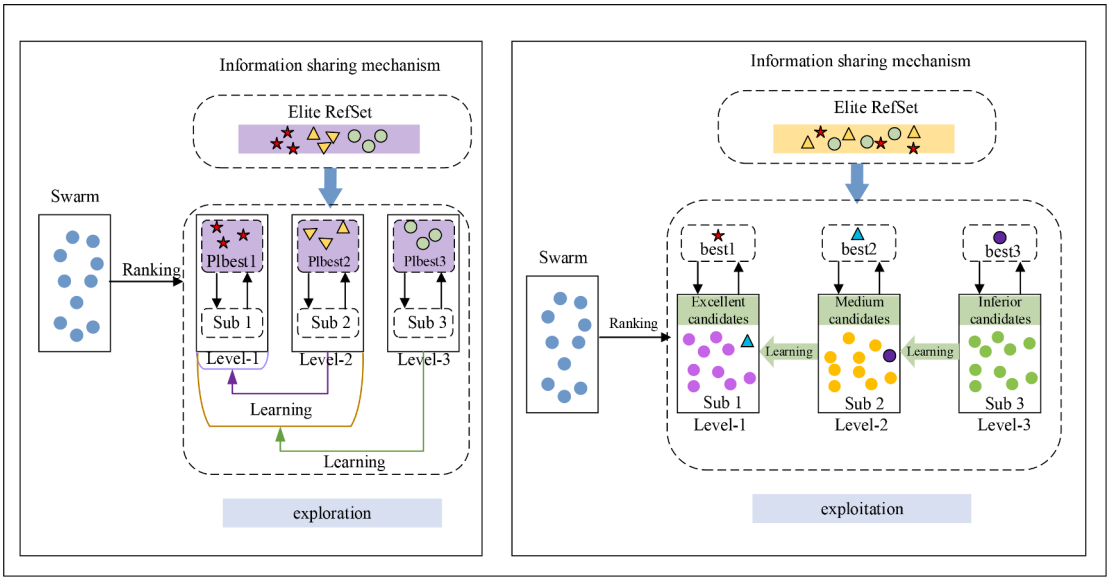


Fig. 2. Multi-population hierarchical learning mechanism of TCSSMH.

Table 1
The parameters of the algorithms and their values.

Algorithms	Parameter settings
SS	$Psize = 10*b, b = 10$
CSS	$Psize = 100, b_1 = 10, b_2 = 10, \alpha = 0.5$
SSDE	$Psize = 100, b = 35, F\ N(0.8, 0.3), Cr\ N(0.3, 0.3), t_{max} = 50$
SSCOL	$Psize = 10*D, b = 36, J_r = 0.3, Balance = 0.5$
CeSSOL	$b = 10, J_r = 0.3$
SS-MCA	$Psize = 100, b_1 = 30, b_2 = 20, k = 5, Fr = 0.2, Cr = 0.25$
BSA	$F = 3.0*randn$
MPEDE	$\lambda_1 = \lambda_2 = \lambda_3 = 0.2, NP = 250, ng = 20$
MAPSO	$N_p = 3, F = 1.0, CR = 0.5, p_m = 0.01$
TAPSO	$w = 0.7298, p_c = 0.5, p_m = 0.02, M = N/4$
XPSO	$\eta = 0.2, Stag_{max} = 5, p = 5$
TCSSMH	$b = 50, e = 15, Pbest\_rate = 0.3, \alpha = 1.4, \beta = 0.2$

Table 2
The parameters for different levels.

Parameters	Levels				
	1	2	3	4	5
b	20	30	50	80	100
e	5	10	15	20	25
$Pbest\_rate$	0.1	0.2	0.3	0.4	0.5
α	1.1	1.2	1.3	1.4	1.5
β	0.1	0.2	0.3	0.4	0.5

where r is a random number in the range (0,1).

4. TCSSMH algorithm

The key to improving the performance of intelligent optimization algorithms is to balance exploitation and exploration reasonably. The exploration refers to the global search in the solution space of an optimization problem by an intelligent optimization algorithm to explore the potential region of the optimal solution. The exploitation capacity is the process of generating new solutions by local searching in the adjacent area of the high-quality solution. In this section, the improved SS is mainly described. The original SS algorithm mainly refers to the individuals in the current *RefSet* to generate new solutions through different combination methods. This combination method is often difficult to find potential areas. In particular, SS has slow convergence speed, low accuracy, and easy premature convergence in solving complex high-dimensional peak optimization problems. The main reason for the low accuracy is that the traditional local improvement method consumes a lot of computing resources, so it is hard to find the potential region of the optimal solution distribution.

In the basic SS algorithm, when the diversity of the *RefSet* becomes worse in the later stage of evolution, the evolutionary ability decreases. This situation limits the ability of individuals to find the global optimal solution. How to improve the diversity and search ability of the population by modifying the update process of the *RefSet* is very important to improve the overall optimization performance of SS. Therefore, the elite *RefSet* is introduced to guide the direction of population evolution. The main framework of TCSSMH follows the basic SS framework. In the process of generating the new solution, the population is divided into multiple subpopulations, which further improves search efficiency. In the final improvement stage, pattern search is applied to deeply explore a better solution. This multi-population cooperation strategy and individual interactive learning mechanism improve the optimization ability and adaptability of the algorithm. The flowchart of TCSSMH is shown in Fig. 1.

4.1. Adaptive bi-directional selection search strategy

The advantages and disadvantages of swarm intelligence optimization algorithms lie in effectively balancing the ability of exploration and exploitation. The adaptive bidirectional selection search mechanism of the elite *RefSet* is proposed to improve the development ability. In the global iteration, multiple elite individuals are selected to form a separate elite group, N individuals in the population are arranged in ascending order of fitness values, the top e individuals are selected as elite individuals, e represents the number of individuals in the elite reference set (The specific value is given through the parameter experiment in the fifth part). An adaptive two-way selection search strategy is introduced, which not only retains the excellent genes of elite individuals but also conducts a deeper and finer search in the field space of elite individuals to generate new individuals with higher quality. The specific mathematical formula for this performance are as follows.

$$\omega = e^{\frac{maxnfes - nfes}{maxnfes}} \tag{7}$$

$$\Delta x_{ij}^{1} = \omega * rand * x_{ij} \tag{8}$$

Table 3
Parameter combinations and average error values.

NO.	Parameter combinations					AVE			TAVE
	b	*e*	*Pbest_rate*	α	β	10*D*	30*D*	50*D*	
1	1	1	1	1	1	1.27E + 02	3.87E + 02	3.95E + 03	1.49E + 03
2	1	2	2	2	2	1.05E + 02	3.04E + 02	3.81E + 03	1.41E + 03
3	1	3	3	3	3	1.12E + 02	3.05E + 02	2.19E + 03	8.70E + 02
4	1	4	4	4	4	1.01E + 02	2.12E + 02	3.21E + 03	1.17E + 03
5	1	5	5	5	5	1.18E + 02	3.10E + 02	4.27E + 03	1.57E + 03
6	2	1	2	3	4	1.11E + 02	3.82E + 02	3.71E + 03	1.40E + 03
7	2	2	3	4	5	9.81E + 01	2.60E + 02	2.03E + 03	7.96E + 02
8	2	3	4	5	1	1.10E + 02	3.22E + 02	3.23E + 03	1.22E + 03
9	2	4	5	1	2	1.01E + 02	2.19E + 02	2.68E + 03	1.00E + 03
10	2	5	1	2	3	1.12E + 02	4.13E + 02	4.63E + 03	1.72E + 03
11	3	1	3	5	2	9.50E + 01	1.78E + 02	2.32E + 03	8.64E + 02
12	3	2	4	1	3	9.13E + 01	2.37E + 02	1.88E + 03	7.35E + 02
13	3	3	5	2	4	9.03E + 01	2.60E + 02	1.80E + 03	7.17E + 02
14	3	4	1	3	5	8.34E + 01	2.19E + 02	2.15E + 03	8.17E + 02
15	3	5	2	4	1	9.27E + 01	2.32E + 02	2.01E + 03	7.79E + 02
16	4	1	4	2	5	1.03E + 02	2.73E + 02	3.59E + 03	1.32E + 03
17	4	2	5	3	1	1.06E + 02	2.76E + 02	4.18E + 03	1.52E + 03
18	4	3	1	4	2	8.51E + 01	2.01E + 02	2.51E + 03	9.33E + 02
19	4	4	2	5	3	1.14E + 02	2.05E + 02	3.28E + 03	1.20E + 03
20	4	5	3	1	4	1.20E + 02	2.51E + 02	3.52E + 03	1.30E + 03
21	5	1	5	4	3	1.12E + 02	2.83E + 02	6.47E + 03	2.29E + 03
22	5	2	1	5	4	1.21E + 02	2.89E + 02	4.77E + 03	1.73E + 03
23	5	3	2	1	5	1.20E + 02	3.33E + 02	5.74E + 03	2.07E + 03
24	5	4	3	2	1	1.30E + 02	3.40E + 02	5.32E + 03	1.93E + 03
25	5	5	4	3	2	1.31E + 02	3.19E + 02	4.84E + 03	1.76E + 03

Table 4
Rank of parameter.

Levels	Parameters				
	b	*e*	*Pbest_rate*	α	β
1	1.30E + 03	1.47E + 03	1.34E + 03	1.32E + 03	1.39E + 03
2	1.23E + 03	1.24E + 03	1.37E + 03	1.42E + 03	**1.19E + 03**
3	**7.82E + 02**	**1.16E + 03**	**1.15E + 03**	1.27E + 03	1.36E + 03
4	1.25E + 03	1.22E + 03	1.24E + 03	**1.19E + 03**	1.26E + 03
5	1.95E + 03	1.42E + 03	1.42E + 03	1.32E + 03	1.31E + 03
Std.	4.20E + 02	1.36E + 02	1.07E + 02	8.14E + 01	7.81E + 01
Rank	1	2	3	4	5

$$\Delta x_{ij}^{2} = -\Delta x_{ij}^{1} \tag{9}$$

$$x_{ij}^{new1} = x_{ij}^{old} + \Delta x_{ij}^{1} \tag{10}$$

$$x_{ij}^{new2} = x_{ij}^{old} + \Delta x_{ij}^{2} \tag{11}$$

$$x_{ij}^{new} = \begin{cases} x_{ij}^{new1}, f(x_{ij}^{old} + \Delta x_{ij}^{1}) \le f(x_{ij}^{old}) \\ x_{ij}^{new2}, f(x_{ij}^{old} + \Delta x_{ij}^{1}) > f(x_{ij}^{old}) \\ x_{ij}^{old}, others \end{cases} \tag{12}$$

where ω is the contraction factor, *maxnfes* is the maximum number of evaluations, *nfes* is the current number of evaluations, x_{ij} is the *j-th* dimension of individual x_i, *rand* is a random value between 0 and 1.

From Eq. (7), it can be seen that the contraction factor ω gradually

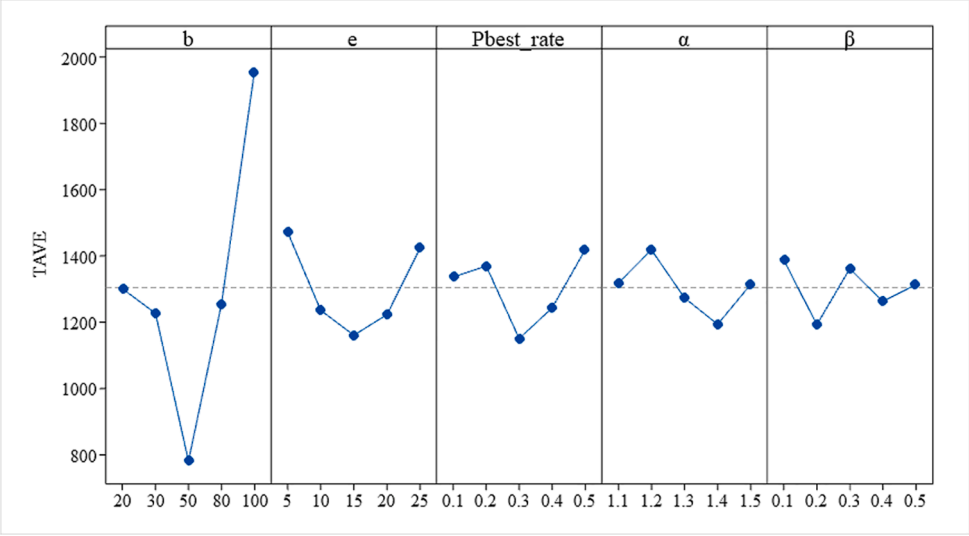


Fig. 3. The tendency of parameters of the TCSSMH.

Table 5
The result of SS, M−SS, ME-SS, TCSSMH.

Type	SS		M−SS		ME-SS		TCSSMH	
	mean	std	mean	std	mean	std	mean	std
Unimodal	2.03E + 05	1.31E + 05	1.40E-08	0.00E + 00	0.00E + 00	0.00E + 00	**0.00E + 00**	**0.00E + 00**
Multimodal	7.94E + 02	2.06E + 02	3.71E + 02	1.17E + 02	2.23E + 02	1.34E + 02	**1.97E + 02**	**1.07E + 02**
Hybrid	2.52E + 04	1.07E + 04	1.28E + 01	3.82E + 00	4.34E + 00	3.33E + 00	**2.77E + 00**	**2.72E + 00**
Composition	9.46E + 04	4.27E + 05	1.92E + 05	9.97E + 03	2.67E + 03	4.83E + 02	**2.42E + 03**	**2.50E + 02**
Average	8.08E + 04	1.42E + 05	4.80E + 04	2.52E + 03	7.25E + 02	9.69E + 01	**6.54E + 02**	**1.51E + 01**

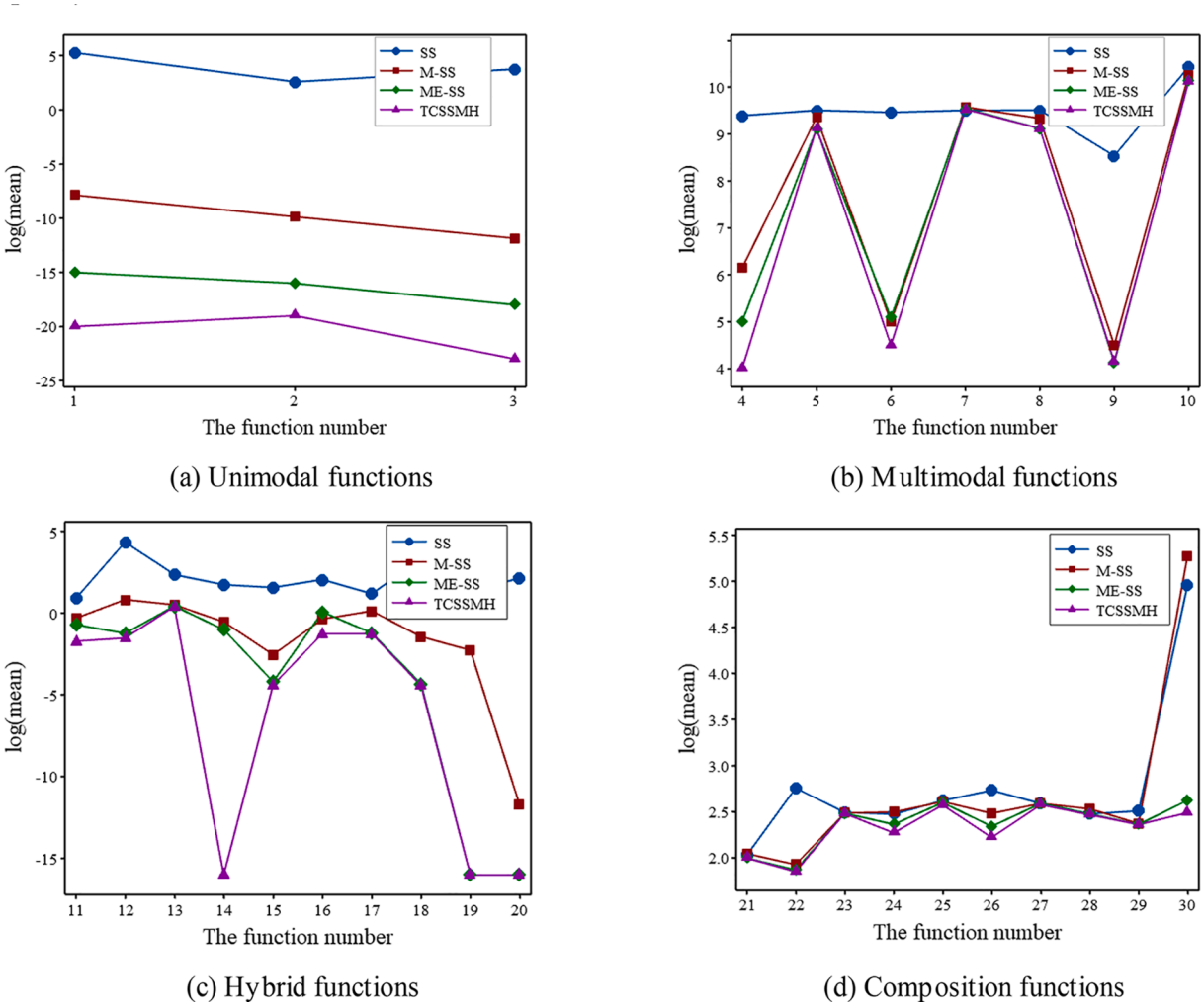


Fig. 4. The scatter diagrams on different functions of SS, M−SS, ME-SS, TCSSMH.

decreases with the increase of evaluation times. The smaller the value of ω, the smaller the effect of Δx_{ij} on the candidate solution. If a better solution is generated by Eq. (10), the original elite individual will be replaced; otherwise, the random disturbance generated by Eq. (9) will be explored in reverse order, and then the better solution will be obtained by Eq. (11). If no better solution is found, keep the original elite individuals unchanged to avoid the deterioration of the generated population individuals. This adaptive bidirectional selection search mechanism enhances the ability of the algorithm to escape from the local optimum, so as to guide the effective search of population and accelerate the convergence rate. The detailed process of elite *RefSet* update is described as follows in Algorithm 2.

Algorithm 2: The adaptive bidirectional selection search strategy

```
1   for each individual i and i from 1 to e do
```

(continued on next column)

(continued)

Algorithm 2: The adaptive bidirectional selection search strategy

```
2       for each dimension d from 1 to D do
3           Generate a random number rand between [−1,1];
4           Update elite RefSet by Eqs (7) – (12);
5       end for
6       if f(x_i^new) < f(x_i^old) then
7           Select better elite individuals to enter the population;
8       end if
9   end for
```

4.2. Multi population hierarchical learning mechanism(MHLM)

In the *RefSet* updating stage of the basic SS algorithm, the combination of two or more solutions is usually used to generate new solutions, and the solution space of the optimization problem is strategically

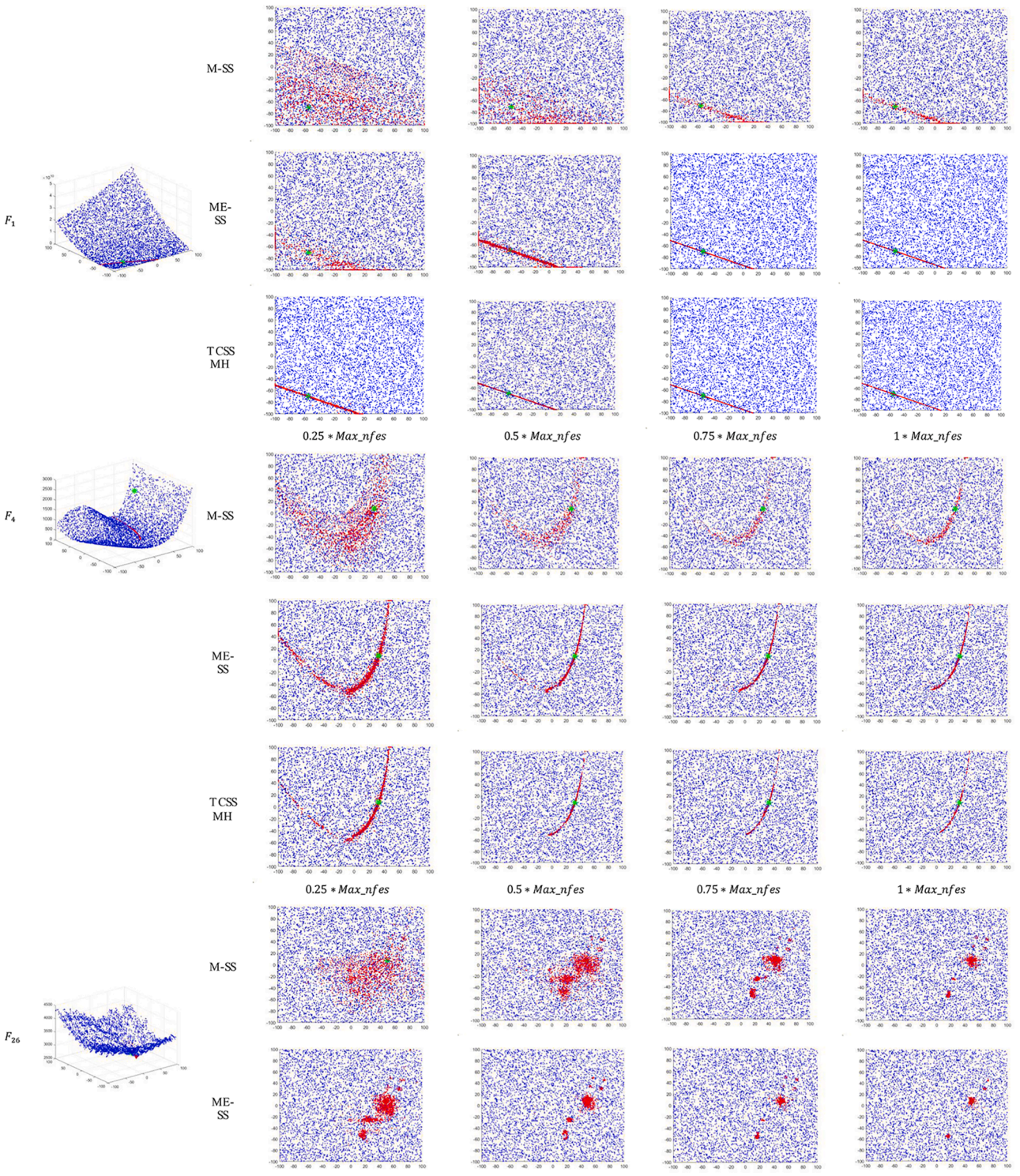


Fig. 5. Three-dimensional location distribution and two-dimensional location distribution of the algorithms.

explored through a series of reference points. These points form a set called a *RefSet*. Because these new solutions are based on specific rules, it greatly limits the search scope and is not conducive to maintaining the diversity of the population. Multi-population manipulation is an effective way to improve population diversity, the information interaction between subgroups can provide more information for individual learning. Therefore, the population is divided into three levels according to the fitness value. Excellent individuals play an important role in guiding the population to the potential high-quality areas. In the early search phase, the optimal number of individuals should be large, which can keep the diversity of the population. In the later stage of evolution, we only set the optimal individual to guide the population to explore deeply. Different elite individuals are selected to guide different subgroups to achieve the purpose of coevolution. At the same time, the elite *RefSet* is invested in steering the direction of population evolution. In this scheme, at the end of each generation, the elite *RefSet* is updated

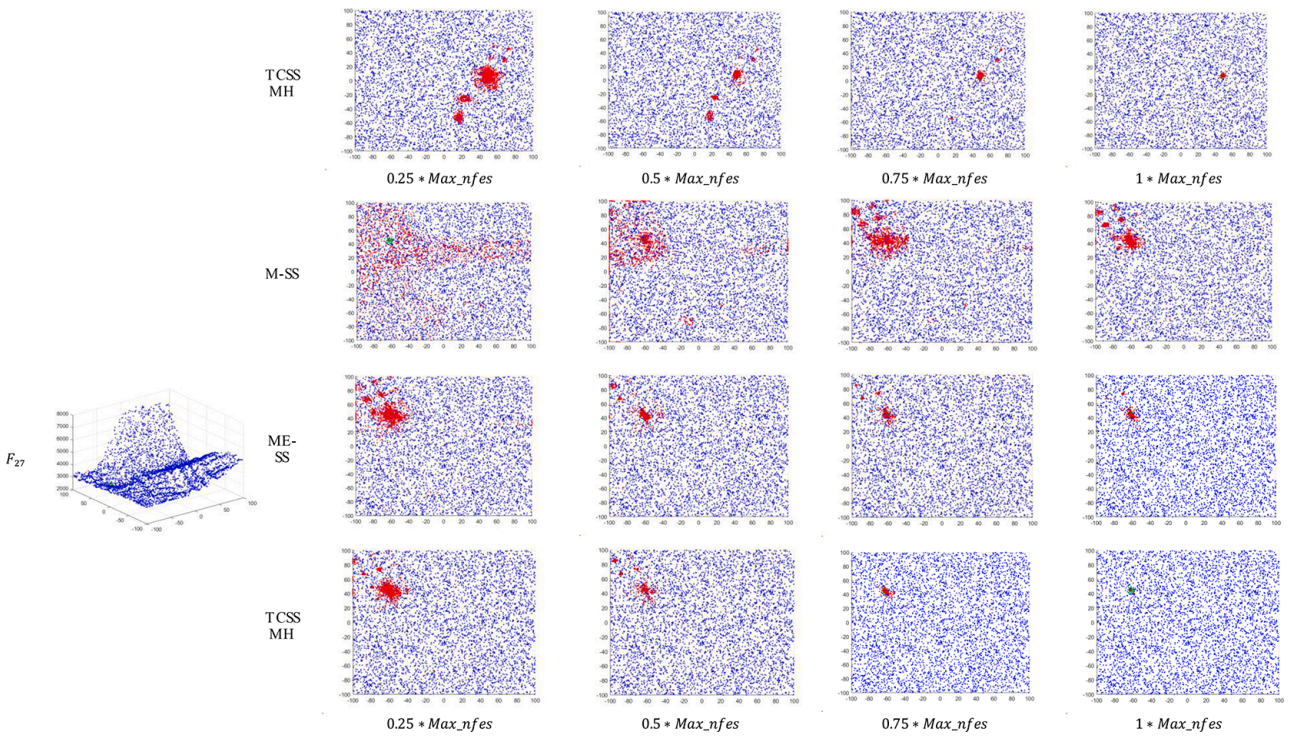


Fig. 5. (*continued*).

according to the ranking of fitness values.

A large number of experimental studies show that the multi-population method is one of the most effective methods to maintain population diversity. In swarm intelligence optimization algorithms, diversity is expressed as the difference between individuals. Population diversity has a great impact on the quality of solutions and convergence speed. Therefore, in this paper, multiple population methods are used to maintain the diversity of solutions in the search space. The population of the candidates is divided into three levels including excellent candidates, medium candidates, and inferior candidates according to the fitness value of the function. The ratios of each sub-population to the total population are 15%, 70%, and 15% respectively. The population is divided into multiple sub-populations, and each subpopulation is located in a different search space to effectively find the promising optimal solution. In the initialization stage, the population is divided into three levels according to the fitness value. The ratio of each subpopulation to elite individuals is also 15%, 70%, and 15%. Each subpopulation is guided by different excellent individuals from the elite *RefSet*. The independent and parallel evolution of each subpopulation helps to maintain the diversity of the population. To avoid the lack of exchange of dominant information among subpopulations. Two strategies are introduced, the first is to re-divide the individuals in the population into subpopulations after each generation of new solutions in the *RefSet*. This strategy can realize the information exchange and share among different subpopulations. The second strategy is to make full use of the optimal individual information of the subgroup and the information of the elite *RefSet* in the evolution process of each subpopulation, the worst individual interactive learning mechanism (this part is introduced in the next section) is introduced to improve the convergence accuracy of the algorithm and reduce the possibility of the algorithm falling into local optimization. The multi-population hierarchical learning mechanism is shown in Fig. 2.

The balance between exploration and exploitation is a key index to improve the optimization performance of the population. Different mutation operations are used in these two stages, which greatly improves the performance of the algorithm. The new mutation strategy is proposed, which not only makes use of the historical information of the population but also increases the learning of elite population individuals, and greatly improves the convergence speed of the algorithm. In the whole process of evolution, different subpopulations are guided by different elite individuals, which prevents the loss of population diversity and increases the diversity of individual learning knowledge. The three subpopulations search different regions, therefore, different mutation strategies lead to different evolutionary directions of subpopulations. The details are as follows.

$$Mutant_n = \begin{cases} P_i^n + F^*\left(Plbest_i^n - P_i^n\right) + F^*\left(P_{r1}^n - P_{r2}^n\right), exploration \\ P_i^n + F^*\left(Pgbest_k^n - P_{r3}^n\right), exploitation \end{cases} \tag{13}$$

$$Pgbest = \begin{cases} best1, k = 1 \\ best2, k = 2 \\ best3, k = 3 \end{cases} \tag{14}$$

$$F = 1 - \frac{1}{e^{\frac{nfes - Maxnfes}{\mathrm{Max}nfes}}} \tag{15}$$

where, F is an amplitude control factor, n represents the subpopulation ($n = 1, 2, 3$). P is the current individual. The indices $r1, r2$ and $r3$ are distinct integers randomly selected from $[1, N_n]$, N is the number of individuals in different subpopulations. *Plbest* represents the individual in the elite *RefSet*. *Pgbest* represents different types of optimal individuals that guide population evolution in the exploitation stage. *M*axnfes is the maximum number of evaluations. In particular, when $k = 1$, *Pgbest* is the optimal individual in the elite *RefSet*, when $k = 2$, *Pgbest* is the optimal individual in the first subpopulation, and when $k = 3$, *Pgbest* is the optimal individual in the second subpopulation. It should be noted that in order to protect the most promising particles from erroneous updates, the individuals in sub1 only learn from the best individuals in the elite *RefSet* during the exploitation phase.

In the early exploration stage, each subpopulation evolved independently and in parallel under the guidance of different elite individuals to maintain the diversity of the population. With the continuous evolution of the population, the rapid decline of population diversity is easy to cause population premature convergence, the

Table 6
Results and comparison of different algorithms on CEC2017 benchmark functions (10-dimension).

Function		SS	CSS	SSDE	SSCOL	CeSSOL	SS-MCA	BSA	MPEDE	MAPSO	TAPSO	XPSO	TCSSMH
F_1	Mean	1.96E + 05	1.53E + 06	2.86E + 02	4.51E + 07	1.96E + 05	6.79E + 03	1.12E-04	4.58E-10	1.79E + 03	1.99E + 03	1.94E + 03	**0.00E + 00**
	Std	1.29E + 05	1.25E + 06	4.98E + 02	1.28E + 07	2.64E + 04	4.67E + 03	3.92E-04	1.07E-09	2.47E + 03	2.46E + 03	2.17E + 03	**0.00E + 00**
F_2	Mean	4.14E + 02	1.04E + 02	3.42E-07	3.88E + 05	3.56E + 00	3.07E + 00	5.16E-07	0.00E + 00	0.00E + 00	7.56E-05	1.74E-05	**0.00E + 00**
	Std	3.57E + 02	3.74E + 01	4.66E-07	3.42E-07	1.94E + 01	1.63E + 01	1.72E-06	0.00E + 00	0.00E + 00	9.67E-05	2.20E-05	**0.00E + 00**
F_3	Mean	6.08E + 03	7.32E + 03	0.00E + 00	2.30E + 03	1.69E + 00	0.00E + 00	7.07E-05	1.67E-14	7.29E-05	0.00E + 00	0.00E + 00	**0.00E + 00**
	Std	2.60E + 03	3.37E + 03	0.00E + 00	2.12E + 03	5.23E-01	0.00E + 00	2.79E-04	5.25E-14	4.06E-04	0.00E + 00	0.00E + 00	**0.00E + 00**
F_4	Mean	6.31E + 00	7.52E + 00	1.03E + 00	1.18E + 01	9.55E-02	2.33E + 00	1.69E + 00	3.26E-07	2.05E + 00	1.12E-05	2.11E-01	**0.00E + 00**
	Std	5.19E-01	1.95E-01	5.58E-01	1.37E + 01	1.59E-02	1.52E + 00	5.30E-01	1.70E-06	5.40E-01	1.17E-05	1.98E-01	**0.00E + 00**
F_5	Mean	1.07E + 01	5.87E + 01	6.97E + 00	3.41E + 01	1.79E + 01	2.29E + 01	5.75E + 00	5.90E + 00	7.63E + 00	1.22E + 01	6.87E + 00	**1.83E + 00**
	Std	3.22E + 00	1.97E + 01	2.29E + 00	1.02E + 01	8.18E + 00	6.75E + 00	2.49E + 00	1.56E + 00	3.48E + 00	5.45E + 00	3.11E + 00	**9.41E-01**
F_6	Mean	8.75E + 00	2.96E + 01	5.80E-08	3.39E + 00	1.74E + 00	1.07E + 01	0.00E + 00	1.81E-05	1.30E-04	6.15E-04	5.91E-07	**0.00E + 00**
	Std	1.62E + 01	2.41E + 01	4.08E-07	2.64E + 00	4.32E + 00	6.97E + 00	0.00E + 00	4.78E-06	5.39E-04	1.46E-03	1.96E-06	**0.00E + 00**
F_7	Mean	**1.06E + 01**	2.34E + 01	1.72E + 01	4.44E + 01	2.00E + 01	2.23E + 01	1.62E + 01	1.73E + 01	2.13E + 01	1.94E + 01	1.77E + 01	1.18E + 01
	Std	5.04E + 00	9.29E + 00	2.83E + 00	7.21E + 00	5.98E + 00	4.74E + 00	2.13E + 00	1.73E + 00	5.44E + 00	4.32E + 00	3.59E + 00	**9.56E-01**
F_8	Mean	1.08E + 01	4.65E + 01	6.38E + 00	3.19E + 01	1.61E + 01	2.39E + 01	4.94E + 00	6.73E + 00	7.96E + 00	1.19E + 01	6.67E + 00	**1.78E + 00**
	Std	5.07E + 00	1.73E + 01	3.10E + 00	8.87E + 00	6.57E + 00	6.74E + 00	1.93E + 00	1.65E + 00	3.65E + 00	5.24E + 00	2.60E + 00	**1.00E + 00**
F_9	Mean	1.21E-01	5.35E + 02	0.00E + 00	8.45E + 00	6.49E-02	1.59E + 02	0.00E + 00	0.00E + 00	0.00E + 00	1.16E-01	0.00E + 00	**0.00E + 00**
	Std	8.46E-02	6.03E + 02	0.00E + 00	3.33E + 00	2.12E-02	1.73E + 02	0.00E + 00	0.00E + 00	0.00E + 00	3.48E-01	0.00E + 00	**0.00E + 00**
F_{10}	Mean	7.47E + 02	1.09E + 03	2.15E + 02	9.58E + 02	5.09E + 02	4.39E + 02	6.34E + 02	2.86E + 02	3.07E + 02	4.14E + 02	2.85E + 02	**1.84E + 02**
	Std	1.76E + 02	1.33E + 02	1.67E + 02	3.06E + 02	2.25E + 02	1.26E + 02	2.24E + 02	**1.12E + 02**	1.92E + 02	1.86E + 02	1.54E + 02	1.14E + 02
F_{11}	Mean	8.10E + 00	2.50E + 01	2.05E + 00	4.76E + 01	4.63E + 00	1.36E + 01	1.26E + 00	2.20E + 00	5.11E + 00	9.71E + 00	2.47E + 00	**1.95E-02**
	Std	3.25E + 00	5.35E + 00	1.45E + 00	3.46E + 01	4.26E + 00	6.90E + 00	9.08E-01	6.55E-01	3.09E + 00	6.94E + 00	1.81E + 00	**1.39E-01**
F_{12}	Mean	2.25E + 04	8.38E + 04	3.53E + 03	2.69E + 06	8.80E + 03	7.54E + 03	7.56E + 02	1.36E + 01	1.14E + 04	9.60E + 03	1.23E + 04	**3.04E-02**
	Std	8.77E + 03	4.96E + 04	4.73E + 03	2.06E + 06	9.60E + 03	5.65E + 03	2.44E + 03	1.51E + 01	7.54E + 03	1.01E + 04	9.34E + 03	**7.24E-02**
F_{13}	Mean	2.29E + 02	1.64E + 03	6.56E + 00	2.95E + 04	1.65E + 02	1.04E + 02	6.81E + 00	5.16E + 00	6.04E + 01	4.63E + 03	6.64E + 03	**2.61E + 00**
	Std	1.91E + 02	1.11E + 03	5.35E + 00	2.06E + 04	4.97E + 02	7.63E + 01	2.38E + 00	2.29E + 00	8.50E + 01	3.84E + 03	5.68E + 03	**2.25E + 00**
F_{14}	Mean	5.56E + 01	6.88E + 03	1.93E + 00	3.85E + 02	4.83E + 01	6.28E + 01	1.90E + 00	4.52E + 00	1.06E + 01	1.41E + 03	3.91E + 01	**0.00E + 00**
	Std	3.31E + 01	4.76E + 03	1.02E + 00	6.89E + 02	3.07E + 01	2.26E + 01	1.44E + 00	1.63E + 00	9.76E+	1.14E + 03	1.58E + 01	**0.00E + 00**
F_{15}	Mean	3.78E + 01	1.16E + 03	1.54E + 00	6.76E + 02	2.46E + 01	3.52E + 01	6.91E-01	6.91E-01	5.54E + 00	2.08E + 03	4.64E + 01	**4.11E-05**
	Std	6.67E + 01	1.08E + 03	1.25E + 00	6.22E + 02	3.97E + 01	3.64E + 01	6.62E-01	2.28E-01	3.59E + 00	3.02E + 03	7.88E + 01	**8.33E-05**
F_{16}	Mean	1.15E + 02	1.69E + 02	6.02E-01	1.31E + 02	8.57E + 01	3.36E + 01	8.35E + 00	2.41E + 00	3.29E + 00	1.47E + 02	7.97E + 01	**5.67E-02**
	Std	5.12E + 01	7.26E + 01	2.62E-01	9.85E + 01	8.78E + 01	4.85E + 01	1.52E + 01	9.13E-01	1.69E + 01	1.12E + 02	9.39E + 01	**8.71E-02**
F_{17}	Mean	1.57E + 01	3.36E + 01	1.73E + 00	8.19E + 01	4.48E + 01	3.31E + 01	1.33E + 01	8.00E + 00	1.03E + 01	3.14E + 01	2.43E + 01	**5.62E-02**
	Std	9.08E + 00	1.23E + 01	1.04E + 00	3.36E + 01	3.11E + 01	1.08E + 01	7.11E + 00	1.96E + 00	1.00E + 01	3.21E + 01	1.51E + 01	**1.68E-01**
F_{18}	Mean	2.08E + 03	3.81E + 03	1.94E + 00	6.94E + 04	2.00E + 02	7.93E + 02	1.11E + 00	2.87E + 00	3.72E + 01	7.67E + 03	7.40E + 03	**3.72E-05**
	Std	1.45E + 03	8.37E + 02	4.00E + 00	5.68E + 04	4.77E + 02	6.42E + 02	1.21E + 00	1.42E + 00	2.75E + 01	1.03E + 04	6.61E + 03	**7.46E-05**
F_{19}	Mean	2.00E + 01	9.83E + 03	1.90E-01	5.87E + 02	1.13E + 01	2.51E + 01	2.43E-01	6.07E-01	1.28E + 00	5.12E + 03	5.15E + 01	**0.00E + 00**

(*continued on next page*)

Table 6 (*continued*)

Function		SS	CSS	SSDE	SSCOL	CeSSOL	SS-MCA	BSA	MPEDE	MAPSO	TAPSO	XPSO	TCSSMH
	Std	2.66E + 01	5.94E + 03	4.06E-01	2.62E + 03	1.46E + 01	2.75E + 01	3.29E-01	1.93E-01	7.10E-01	6.92E + 03	8.01E + 01	**0.00E + 00**
F_{20}	Mean	1.45E + 02	1.59E + 02	3.04E-01	1.17E + 02	7.75E + 01	3.66E + 01	1.22E + 00	1.25E + 00	1.84E + 00	7.10E + 00	3.63E + 01	**0.00E + 00**
	Std	9.29E + 01	6.95E + 01	4.72E-01	6.84E + 01	6.35E + 01	1.75E + 01	1.44E + 00	7.66E-01	4.08E + 00	8.12E + 00	4.81E + 01	**0.00E + 0**
F_{21}	Mean	1.07E + 02	1.16E + 02	1.30E + 02	2.06E + 02	1.86E + 02	1.00E + 02	1.60E + 02	1.13E + 02	1.50E + 02	1.96E + 02	1.83E + 02	**1.00E + 02**
	Std	2.21E + 01	4.49E + 01	4.85E + 01	5.67E + 01	5.36E + 01	1.42E + 01	5.19E + 01	3.49E + 01	5.41E + 01	4.19E + 01	4.68E + 01	**1.09E-13**
F_{22}	Mean	5.73E + 02	1.30E + 02	9.83E + 01	1.14E + 02	1.03E + 02	8.37E + 01	9.98E + 01	9.22E + 01	9.48E + 01	1.01E + 02	1.02E + 02	**7.12E + 01**
	Std	4.48E + 02	1.42E + 02	1.40E + 01	1.21E + 01	1.01E + 01	3.48E + 01	**4.08E + 00**	2.72E + 01	2.38E + 01	1.41E + 00	**8.39E-01**	4.10E + 01
F_{23}	Mean	3.12E + 02	3.25E + 02	3.08E + 02	3.35E + 02	3.19E + 02	**2.72E + 02**	3.13E + 02	3.06E + 02	3.10E + 02	3.17E + 02	3.08E + 02	3.06E + 02
	Std	4.19E + 00	1.07E + 01	4.07E + 00	1.22E + 01	9.71E + 00	1.15E + 02	5.76E + 00	2.54E + 00	3.68E + 00	6.02E + 00	4.85E + 00	**1.89E + 00**
F_{24}	Mean	2.99E + 02	2.58E + 02	2.67E + 02	3.04E + 02	3.13E + 02	**1.27E + 02**	3.45E + 02	2.28E + 02	3.04E + 02	3.40E + 02	3.08E + 02	2.32E + 02
	Std	7.90E + 01	9.89E + 01	1.06E + 02	1.13E + 02	8.56E + 01	6.56E + 01	**5.76E + 00**	1.17E + 02	8.24E + 01	4.95E + 01	7.69E + 01	1.17E + 02
F_{25}	Mean	4.18E + 02	3.99E + 02	4.23E + 02	4.24E + 02	4.22E + 02	**3.61E + 02**	4.20E + 02	4.05E + 02	4.17E + 02	4.30E + 02	4.21E + 02	3.74E + 02
	Std	2.36E + 01	**1.64E-01**	2.27E + 01	2.29E + 01	2.36E + 01	1.04E + 02	2.31E + 01	1.68E + 01	2.34E + 01	2.28E + 01	2.35E + 01	8.08E + 01
F_{26}	Mean	5.42E + 02	4.19E + 02	2.98E + 02	3.40E + 02	3.29E + 02	2.56E + 02	3.35E + 02	3.00E + 02	3.00E + 02	3.86E + 02	2.97E + 02	**1.67E + 02**
	Std	4.58E + 02	3.10E + 02	1.40E + 01	2.39E + 01	1.94E + 02	9.18E + 01	1.31E + 02	4.91E-09	**0.00E + 00**	2.55E + 02	1.43E + 02	1.31E + 02
F_{27}	Mean	3.89E + 02	3.90E + 02	3.94E + 02	4.02E + 02	3.91E + 02	3.92E + 02	3.92E + 02	3.89E + 02	3.90E + 02	3.96E + 02	3.98E + 02	**3.78E + 02**
	Std	4.98E-01	1.11E + 00	2.25E + 00	2.40E + 01	2.71E + 00	2.62E + 00	3.58E + 00	4.19E-01	**2.14E-01**	1.30E + 01	1.39E + 01	4.81E + 00
F_{28}	Mean	3.01E + 02	3.31E + 02	3.18E + 02	4.61E + 02	3.52E + 02	3.12E + 02	4.25E + 02	3.06E + 02	3.93E + 02	4.82E + 02	4.87E + 02	**2.94E + 02**
	Std	**4.26E-01**	5.35E + 01	7.11E + 01	1.45E + 02	1.27E + 02	1.13E + 02	1.38E + 02	3.97E + 01	1.39E + 02	1.32E + 02	1.40E + 02	4.20E + 01
F_{29}	Mean	3.23E + 02	4.25E + 02	2.50E + 02	3.23E + 02	2.93E + 02	2.73E + 02	2.44E + 02	2.49E + 02	2.54E + 02	2.90E + 02	2.60E + 02	**2.30E + 02**
	Std	5.32E + 01	8.81E + 01	1.05E + 01	5.89E + 01	4.98E + 01	2.64E + 01	7.30E + 00	5.67E + 00	1.14E + 01	4.13E + 01	1.93E + 01	**1.72E + 00**
F_{30}	Mean	9.14E + 04	3.33E + 06	1.11E + 03	7.31E + 05	2.47E + 05	5.81E + 03	7.82E + 05	3.96E + 02	1.67E + 05	2.25E + 05	4.08E + 05	**3.10E + 02**
	Std	4.26E + 05	1.80E + 06	5.59E + 02	1.23E + 06	4.57E + 05	4.79E + 03	2.65E + 05	**1.93E + 00**	3.27E + 05	3.54E + 05	7.85E + 05	6.31E + 01

difference between populations is becoming smaller and smaller, forming an aggregation phenomenon. The overall change of the fitness value of all individuals in the population can effectively judge the state of the individual. In this paper, the fitness variance of the population is calculated to judge whether the algorithm has entered the development stage. The specific description process is as follows.

Suppose the fitness of individual i is f_i, the average fitness of the current population is f_{avg} , and σ^2 is the fitness variance of the population, which is defined as:

$$\sigma^2 = \sum_{i=1}^{NP} \left(\frac{f_i - f_{avg}}{f} \right)^2 \tag{16}$$

where, f is the normalized calibration factor, which limits the size of σ^2, the value of f is determined by the following formula:

$$f(x) = \begin{cases} max\{|f_i - f_{avg}|\}, max\{|f_i - f_{avg}|\} \rangle 1 \\ 1, others \end{cases} \tag{17}$$

where,$i = 1, 2, \cdots, NP$, the population fitness variance reflects the aggregation degree of individuals in the population through Eq. (16). The smaller σ^2, the greater the degree of aggregation of individuals in the population, the individuals are in the random search stage, and vice versa. When condition $\sigma^2 < C$ ($C = 0.001$)is satisfied, the algorithm is considered to enter the exploitation stage.

4.3. Cooperative interactive learning mechanism

Inspired by the characteristics of interactive learning among different groups in human society, the information interaction between individuals and their excellent peers can expand the scope of individual perception and improve the speed and accuracy of individual perception of information. The information interaction among members is conducive to individual evolution. Before the population regrouping, due to the lack of cooperative learning among subgroups, even after finding the globally optimal region, individuals will not quickly converge to the globally optimal solution. In order to accelerate the convergence speed and exchange information before subgroup reorganization, a collaborative interactive learning mechanism is introduced. Through this interactive learning mechanism, the quality elements of other subpopulations are acquired, and the quality of the whole population is improved. It also realizes the balance between global exploration and local development. The details of collaborative interactive learning are

Table 7
Results and comparison of different algorithms on CEC2017 benchmark functions (30-dimension).

Function		SS	CSS	SSDE	SSCOL	CeSSOL	SS-MCA	BSA	MPEDE	MAPSO	TAPSO	XPSO	TCSSMH
F_1	Mean	2.29E + 07	1.72E + 08	2.57E + 03	6.70E + 08	3.09E + 06	2.91E + 03	3.93E + 01	0.00E + 00	2.34E + 03	3.09E + 03	3.09E + 03	**0.00E + 00**
	Std	1.15E + 07	2.98E + 07	2.48E + 03	5.65E + 08	6.24E + 05	4.67E + 03	1.68E + 02	0.00E + 00	3.09E + 03	3.25E + 03	3.74E + 03	**0.00E + 00**
F_2	Mean	2.31E + 15	6.93E + 18	1.41E-05	2.54E + 20	7.75E + 06	1.55E + 16	1.96E + 28	0.00E + 00	8.56E + 04	1.91E-04	2.54E + 12	**0.00E + 00**
	Std	4.19E + 15	1.73E + 19	5.62E-05	1.01E + 21	3.39E + 07	6.95E + 16	1.40E + 29	0.00E + 00	3.81E + 05	2.21E-04	9.44E + 12	**0.00E + 00**
F_3	Mean	2.31E + 04	1.90E + 05	2.28E-06	3.86E + 04	4.59E + 04	6.43E + 04	1.51E + 03	0.00E + 00	2.06E + 00	4.63E-07	9.28E-03	**0.00E + 00**
	Std	7.61E + 03	4.55E + 04	1.65E-06	1.56E + 04	1.94E + 04	1.33E + 04	8.77E + 03	0.00E + 00	2.25E + 00	3.30E-06	2.87E-02	**0.00E + 00**
F_4	Mean	1.37E + 02	2.59E + 02	5.79E + 01	1.43E + 02	8.48E + 01	9.30E + 01	7.35E + 01	5.24E + 01	8.97E + 01	1.95E + 01	1.17E + 02	**1.74E + 01**
	Std	3.21E + 01	3.08E + 01	4.22E + 01	2.08E + 01	1.76E + 01	1.21E + 01	3.67E + 01	1.92E + 01	**7.03E + 00**	2.77E + 01	2.58E + 01	9.00E + 00
F_5	Mean	2.28E + 02	4.87E + 02	4.16E + 01	2.09E + 02	1.97E + 02	2.18E + 02	6.36E + 01	**2.86E + 01**	3.39E + 01	5.40E + 01	4.28E + 01	5.29E + 01
	Std	5.78E + 01	1.14E + 02	1.21E + 01	3.85E + 01	5.17E + 01	4.27E + 01	1.90E + 01	**7.91E + 00**	9.80E + 00	1.33E + 01	9.16E + 00	1.15E + 01
F_6	Mean	4.46E + 01	1.06E + 02	4.04E-06	3.43E + 01	3.67E + 01	4.18E + 01	1.47E-02	1.23E-07	2.52E-01	1.41E-02	4.05E-02	**0.00E + 00**
	Std	1.35E + 01	2.50E + 01	1.10E-05	1.26E + 01	1.22E + 01	9.76E + 00	2.93E-02	3.85E-07	4.03E-01	8.44E-03	7.47E-02	**0.00E + 00**
F_7	Mean	9.75E + 01	5.82E + 02	7.38E + 01	2.51E + 02	1.32E + 02	2.13E + 02	1.15E + 02	**5.66E + 01**	9.32E + 01	8.86E + 01	8.25E + 01	7.76E + 01
	Std	2.10E + 01	1.50E + 02	1.18E + 01	3.34E + 01	3.85E + 01	5.27E + 01	1.88E + 01	**7.86E + 00**	1.64E + 01	1.75E + 01	1.54E + 01	1.19E + 01
F_8	Mean	2.09E + 02	4.77E + 02	4.57E + 01	1.89E + 02	1.65E + 02	2.05E + 02	5.69E + 01	**2.92E + 01**	3.76E + 01	5.76E + 01	4.30E + 01	5.53E + 01
	Std	3.94E + 01	8.32E + 01	1.18E + 01	2.65E + 01	4.44E + 01	4.09E + 01	1.70E + 01	**7.95E + 00**	8.62E + 00	1.37E + 01	1.39E + 01	1.11E + 01
F_9	Mean	3.48E + 03	1.72E + 04	2.69E + 00	4.16E + 03	3.70E + 03	5.55E + 03	1.02E + 02	**2.31E-02**	4.92E + 01	1.10E + 02	5.40E + 00	7.79E-01
	Std	1.98E + 03	6.04E + 03	2.77E + 00	1.99E + 03	1.12E + 03	1.41E + 03	7.70E + 01	5.03E + 00	4.29E + 01	1.56E + 02	4.38E + 00	**1.96E + 00**
F_{10}	Mean	3.26E + 03	3.50E + 03	2.67E + 03	5.18E + 03	2.86E + 03	3.20E + 03	5.50E + 03	2.66E + 03	2.73E + 03	2.65E + 03	2.74E + 03	**1.86E + 03**
	Std	4.64E + 02	6.23E + 02	6.28E + 02	5.87E + 02	6.99E + 02	5.23E + 02	1.02E + 03	4.03E + 02	5.80E + 02	5.52E + 02	5.96E + 02	**3.62E + 02**
F_{11}	Mean	1.88E + 02	1.43E + 03	8.05E + 01	3.52E + 02	1.58E + 02	2.18E + 02	6.39E + 01	2.74E + 01	9.52E + 01	8.53E + 01	8.45E + 01	**1.78E + 01**
	Std	9.03E + 01	8.60E + 02	2.56E + 01	1.17E + 02	6.23E + 01	7.37E + 01	2.76E + 01	1.49E + 01	3.46E + 01	3.70E + 01	3.30E + 01	**8.73E + 00**
F_{12}	Mean	8.70E + 06	2.05E + 07	1.09E + 05	5.09E + 07	2.80E + 05	3.32E + 06	2.36E + 04	8.79E + 02	4.22E + 04	3.19E + 04	1.39E + 05	**3.23E + 02**
	Std	6.18E + 06	1.37E + 07	5.88E + 04	2.81E + 07	2.16E + 05	2.84E + 06	1.35E + 04	3.18E + 02	2.50E + 04	5.09E + 04	3.03E + 05	**1.78E + 02**
F_{13}	Mean	4.04E + 05	9.02E + 06	8.05E + 03	6.09E + 07	7.43E + 03	5.01E + 04	1.37E + 04	**2.09E + 01**	1.46E + 04	1.54E + 04	1.45E + 04	2.85E + 01
	Std	4.54E + 05	3.79E + 06	6.66E + 03	3.36E + 07	1.90E + 04	3.29E + 04	1.34E + 04	8.96E + 00	1.38E + 04	1.55E + 04	1.34E + 04	**8.55E + 00**
F_{14}	Mean	1.05E + 05	2.25E + 06	1.04E + 02	1.05E + 05	2.09E + 03	3.15E + 04	5.40E + 01	**1.25E + 01**	2.36E + 03	3.20E + 03	5.79E + 03	1.55E + 01
	Std	6.94E + 04	1.22E + 06	6.19E + 01	1.99E + 05	5.61E + 03	3.41E + 04	1.52E + 01	9.31E + 00	2.41E + 03	4.44E + 03	5.65E + 03	**9.17E + 00**
F_{15}	Mean	1.54E + 04	1.30E + 05	1.11E + 03	9.12E + 06	2.77E + 02	2.66E + 03	1.10E + 03	8.10E + 01	5.74E + 03	5.52E + 03	4.71E + 03	**1.16E + 01**
	Std	1.90E + 04	1.34E + 05	1.64E + 03	9.45E + 06	4.87E + 02	1.85E + 03	2.51E + 03	**3.94E + 00**	7.38E + 03	7.13E + 03	5.58E + 03	4.90E + 00
F_{16}	Mean	1.00E + 03	2.08E + 03	6.43E + 02	1.32E + 03	5.62E + 02	9.23E + 02	5.68E + 02	**3.57E + 02**	5.66E + 02	8.29E + 02	6.01E + 02	3.96E + 02
	Std	1.86E + 02	8.32E + 02	2.74E + 02	2.63E + 02	2.15E + 02	2.33E + 02	2.58E + 02	1.86E + 02	2.73E + 02	2.95E + 02	2.30E + 02	**1.63E + 02**
F_{17}	Mean	5.39E + 02	1.87E + 03	1.19E + 02	4.97E + 02	2.99E + 02	4.33E + 02	1.37E + 02	**5.06E + 01**	1.09E + 02	3.37E + 02	1.64E + 02	6.28E + 01
	Std	2.34E + 02	4.99E + 02	7.25E + 01	1.72E + 02	1.45E + 02	1.55E + 02	9.57E + 01	**1.92E + 01**	5.96E + 01	1.71E + 02	7.63E + 01	4.77E + 01
F_{18}	Mean	5.04E + 05	1.26E + 06	1.20E + 04	9.80E + 05	6.22E + 04	2.97E + 05	8.05E + 03	2.37E + 01	7.15E + 04	3.58E + 04	1.37E + 05	**2.07E + 01**
	Std	4.17E + 05	7.75E + 05	8.13E + 03	6.72E + 05	4.56E + 04	2.56E + 05	3.17E + 04	**5.37E + 00**	4.87E + 04	1.84E + 04	7.37E + 04	1.00E + 01
F_{19}	Mean	5.09E + 04	5.48E + 05	3.67E + 03	2.10E + 07	1.85E + 02	1.85E + 03	8.98E + 01	7.38E + 00	5.00E + 03	4.13E + 03	6.40E + 03	**6.26E + 00**

(*continued on next page*)

Table 7 (*continued*)

Function		SS	CSS	SSDE	SSCOL	CeSSOL	SS-MCA	BSA	MPEDE	MAPSO	TAPSO	XPSO	TCSSMH
	Std	1.14E + 05	3.22E + 05	4.91E + 03	1.19E + 07	6.77E + 02	2.18E + 03	1.56E + 02	2.40E + 00	6.70E + 03	6.40E + 03	6.96E + 03	**2.30E + 00**
F_{20}	Mean	7.05E + 02	8.83E + 02	1.76E + 02	6.57E + 02	3.59E + 02	4.76E + 02	5.01E + 02	**7.03E + 01**	1.37E + 02	3.03E + 02	1.73E + 02	8.96E + 01
	Std	6.31E + 01	7.40E + 01	1.03E + 02	1.91E + 02	1.58E + 02	1.27E + 02	3.15E + 02	5.39E + 01	6.31E + 01	1.29E + 02	**5.08E + 01**	7.21E + 01
F_{21}	Mean	4.54E + 02	7.85E + 02	2.44E + 02	4.29E + 02	3.84E + 02	3.46E + 02	2.50E + 02	2.31E + 02	2.35E + 02	2.61E + 02	2.46E + 02	**2.31E + 02**
	Std	8.97E + 01	1.52E + 02	1.07E + 01	4.18E + 01	4.94E + 01	1.20E + 02	1.36E + 01	**8.46E + 00**	9.51E + 00	1.38E + 01	1.49E + 01	3.83E + 01
F_{22}	Mean	3.73E + 03	4.15E + 03	1.00E + 02	2.45E + 03	1.82E + 03	1.91E + 03	3.91E + 03	1.00E + 02	1.01E + 02	7.70E + 02	4.19E + 02	**1.00E + 02**
	Std	6.93E + 02	6.11E + 02	1.04E + 00	2.71E + 03	1.63E + 03	1.76E + 03	2.76E + 03	3.44E-01	1.18E + 00	1.30E + 03	9.12E + 02	**8.92E-14**
F_{23}	Mean	6.16E + 02	9.93E + 02	3.94E + 02	6.73E + 02	5.99E + 02	5.92E + 02	4.11E + 02	**3.81E + 02**	3.89E + 02	4.38E + 02	3.98E + 02	3.93E + 02
	Std	6.53E + 01	8.35E + 01	1.35E + 01	1.16E + 02	8.58E + 01	6.10E + 01	4.43E + 01	**1.01E + 01**	1.13E + 01	2.18E + 01	1.85E + 01	1.09E + 01
F_{24}	Mean	6.35E + 02	1.12E + 03	4.64E + 02	8.60E + 02	6.77E + 02	6.44E + 02	4.81E + 02	**4.43E + 02**	4.57E + 02	5.17E + 02	4.73E + 02	4.66E + 02
	Std	5.50E + 01	7.41E + 01	1.32E + 01	1.37E + 02	8.93E + 01	5.03E + 01	2.02E + 01	**7.46E + 00**	1.23E + 01	2.72E + 01	3.37E + 01	1.23E + 01
F_{25}	Mean	4.08E + 02	5.51E + 02	3.90E + 02	4.40E + 02	3.87E + 02	3.88E + 02	3.91E + 02	3.87E + 02	3.88E + 02	3.94E + 02	3.99E + 02	**3.77E + 02**
	Std	2.08E + 01	3.25E + 01	7.24E + 00	3.07E + 01	1.75E + 00	5.59E + 00	1.26E + 01	**6.94E-02**	3.73E-01	1.55E + 01	1.49E + 01	3.14E + 00
F_{26}	Mean	2.99E + 03	6.36E + 03	1.43E + 03	3.17E + 03	2.34E + 03	1.43E + 03	1.79E + 03	1.21E + 03	1.38E + 03	1.50E + 03	9.07E + 02	**3.13E + 02**
	Std	9.01E + 02	5.66E + 02	4.43E + 02	1.18E + 03	1.48E + 03	1.59E + 03	3.62E + 02	1.20E + 02	1.13E + 02	7.44E + 02	6.33E + 02	**1.06E + 02**
F_{27}	Mean	5.46E + 02	6.77E + 02	5.28E + 02	6.21E + 02	5.25E + 02	5.33E + 02	5.61E + 02	5.03E + 02	5.07E + 02	5.20E + 02	5.36E + 02	**4.45E + 02**
	Std	3.08E + 01	8.65E + 01	1.15E + 01	9.37E + 01	2.45E + 01	1.83E + 01	6.54E + 01	6.73E + 00	1.07E + 01	1.61E + 01	1.78E + 01	**4.88E + 00**
F_{28}	Mean	4.52E + 02	4.85E + 02	3.43E + 02	4.96E + 02	3.91E + 02	4.35E + 02	9.96E + 02	3.29E + 02	3.80E + 02	3.44E + 02	3.73E + 02	**3.06E + 02**
	Std	**1.74E + 01**	2.83E + 01	5.34E + 01	3.49E + 01	5.77E + 01	2.07E + 01	1.29E + 03	4.80E + 01	5.11E + 01	6.18E + 01	6.85E + 01	2.53E + 01
F_{29}	Mean	9.38E + 02	2.77E + 03	5.26E + 02	1.07E + 03	7.82E + 02	1.12E + 03	1.02E + 03	4.47E + 02	5.23E + 02	6.85E + 02	5.66E + 02	**3.96E + 02**
	Std	2.09E + 02	5.76E + 02	7.58E + 01	2.02E + 02	1.89E + 02	1.57E + 02	5.66E + 02	**2.85E + 01**	7.22E + 01	1.81E + 02	9.14E + 01	6.27E + 01
F_{30}	Mean	9.90E + 04	4.72E + 05	4.62E + 03	1.15E + 07	5.18E + 04	5.08E + 04	4.03E + 03	2.02E + 03	4.27E + 03	4.76E + 03	7.88E + 03	**4.95E + 02**
	Std	1.21E + 05	2.62E + 05	1.48E + 03	7.64E + 06	3.12E + 05	2.61E + 04	1.80E + 03	**9.16E + 01**	1.88E + 03	2.69E + 03	3.70E + 03	2.62E + 02

described as follows. The pseudo-code of the cooperative interactive learning mechanism is given in Algorithm 3.

(1) Firstly, each subgroup is arranged in ascending order according to the fitness value, and the inferior individuals to be updated are selected. The number selected for each subgroup is 0.9**spopsize*, *spopsize* is the number of each subpopulation.
(2) Considering the limitations of individual learning ability and the diversity of learning methods, these inferior individuals not only learn from the best individuals in the subpopulation but also learn from the best individuals in other subpopulations and from the elite *RefSet* to improve the diversity of individual learning methods. Randomly select an individual from the elite *RefSet* and select the optimal individual of the subpopulation.
(3) To ensure the effectiveness of interactive learning, different subpopulations have different learning objects. Specifically, the inferior individuals in the first subpopulation learn from the elite *RefSet* and the best individuals in the current subpopulation. In addition to learning from the individuals in the elite *RefSet*, the second subpopulation and the third subpopulation learn from the optimal individuals of the previous subpopulation respectively.

Algorithm 3. The cooperative interactive learning mechanism

```
for iter = 0.9*spopsize to spopsize −1 do
    randomly select P_i(m) and P_i(n) from current subpopulation;
    if f(P_i(m)) )(f(P_i(n)) then
        Xnew_ij = X_ij + r*(gbest_j − P_ij(m)) + r*(sbest_j − P_ij(m));
    else
        Xnew_ij = X_ij + r*(gbest_j − P_ij(n)) + r*(sbest_j − P_ij(n));
    end if
end for
```

where, $P_i(m)$ and $P_i(n)$ are two individuals randomly selected except for the optimal individuals of the current subpopulation. *spopsize* is the number of each subpopulation. r is the uniformly distributed random number in the range of (0,1), *gbest* represents individuals randomly selected from the elite *RefSet*, *sbest* represents the optimal individual in

Table 8
Results and comparison of different algorithms on CEC2017 benchmark functions (50-dimension).

Function		SS	CSS	SSDE	SSCOL	CeSSOL	SS-MCA	BSA	MPEDE	MAPSO	TAPSO	XPSO	TCSSMH
F_1	Mean	2.40E + 08	7.30E + 08	2.59E + 03	2.48E + 10	9.52E + 07	4.83E + 06	1.04E + 03	2.20E-14	2.39E + 03	3.94E + 03	3.66E + 03	**0.00E + 00**
	Std	6.90E + 07	2.24E + 08	2.75E + 03	2.08E + 10	3.06E + 07	3.37E + 06	1.41E + 03	7.686E-15	3.70E + 03	4.37E + 03	4.66E + 03	**0.00E + 00**
F_2	Mean	8.38E + 29	1.00E + 30	1.72E + 02	1.00E + 30	2.03E + 28	9.27E + 29	7.03E + 25	4.58E-08	1.01E + 11	7.72E-05	4.19E + 22	**0.00E + 00**
	Std	3.20E + 29	2.84E + 14	2.85E + 02	2.84E + 14	1.40E + 29	2.52E + 29	5.02E + 26	3.27E-07	6.91E + 11	1.22E-04	2.39E + 23	**0.00E + 00**
F_3	Mean	8.24E + 04	6.04E + 05	1.59E-04	1.57E + 05	1.50E + 05	2.01E + 05	8.52E + 03	4.71E-04	1.12E + 02	2.23E + 02	2.70E + 02	**0.00E + 00**
	Std	2.08E + 04	1.13E + 05	1.29E-04	3.72E + 04	3.56E + 04	2.33E + 04	3.91E + 03	2.43E-03	9.07E + 01	4.32E + 02	1.81E + 02	**0.00E + 00**
F_4	Mean	2.68E + 02	3.95E + 02	1.23E + 02	4.81E + 02	1.64E + 02	2.31E + 02	1.06E + 02	4.88E + 01	1.31E + 02	4.64E + 01	2.38E + 02	**2.89E + 01**
	Std	5.87E + 01	6.05E + 01	5.13E + 01	3.91E + 02	5.15E + 01	4.28E + 01	4.81E + 01	4.51E + 01	**2.94E + 01**	4.54E + 01	4.73E + 01	2.95E + 01
F_5	Mean	5.15E + 02	9.43E + 02	1.04E + 02	4.07E + 02	3.31E + 02	4.62E + 02	1.76E + 02	**5.51E + 01**	7.09E + 01	1.29E + 02	9.13E + 01	1.57E + 02
	Std	1.01E + 02	1.53E + 02	1.64E + 01	5.68E + 01	5.67E + 01	7.90E + 01	3.59E + 01	**1.08E + 01**	1.33E + 01	2.63E + 01	2.21E + 01	2.84E + 01
F_6	Mean	5.32E + 01	1.12E + 02	6.88E-05	4.67E + 01	4.14E + 01	5.45E + 01	1.09E + 00	1.77E-03	1.15E + 00	2.21E-02	8.23E-01	**0.00E + 00**
	Std	1.42E + 01	2.22E + 01	1.86E-04	1.01E + 01	9.08E + 00	8.28E + 00	9.21E-01	5.53E-03	5.67E-01	1.47E-02	7.05E-01	**0.00E + 00**
F_7	Mean	2.10E + 02	9.30E + 02	1.65E + 02	5.26E + 02	3.17E + 02	4.89E + 02	3.02E + 02	**1.07E + 02**	1.99E + 02	1.70E + 02	1.76E + 02	1.82E + 02
	Std	3.96E + 01	1.73E + 02	2.08E + 01	6.66E + 01	5.24E + 01	1.34E + 02	5.42E + 01	**1.18E + 01**	3.13E + 01	2.65E + 01	3.22E + 01	2.31E + 01
F_8	Mean	4.44E + 02	8.58E + 02	1.07E + 02	4.22E + 02	3.62E + 02	4.60E + 02	1.79E + 02	**5.61E + 01**	6.84E + 01	1.25E + 02	9.24E + 01	1.53E + 02
	Std	5.92E + 01	1.51E + 02	2.11E + 01	4.97E + 01	6.00E + 01	8.35E + 01	3.09E + 01	**1.12E + 01**	1.41E + 01	2.49E + 01	2.20E + 01	3.35E + 01
F_9	Mean	1.74E + 04	5.46E + 04	1.35E + 02	1.89E + 04	1.64E + 04	1.81E + 04	1.67E + 03	**1.51E + 00**	2.52E + 02	6.79E + 02	1.24E + 02	2.63E + 02
	Std	6.03E + 03	1.22E + 04	1.61E + 02	4.34E + 03	3.76E + 03	2.85E + 03	8.58E + 02	**1.41E + 00**	1.33E + 02	4.20E + 02	1.11E + 02	2.45E + 02
F_{10}	Mean	6.38E + 03	8.06E + 03	4.90E + 03	9.29E + 03	7.00E + 03	5.73E + 03	1.09E + 04	4.73E + 03	4.73E + 03	4.35E + 03	5.16E + 03	**4.18E + 03**
	Std	7.72E + 02	1.02E + 03	7.58E + 02	7.94E + 02	1.14E + 03	8.38E + 02	1.60E + 03	7.02E + 02	8.63E + 02	7.17E + 02	8.34E + 02	**5.52E + 02**
F_{11}	Mean	2.26E + 03	1.94E + 04	1.48E + 02	6.63E + 03	3.88E + 02	3.67E + 02	1.34E + 02	9.98E + 01	1.70E + 02	1.42E + 02	1.79E + 02	**6.37E + 01**
	Std	1.02E + 03	5.75E + 03	3.98E + 01	4.70E + 03	9.13E + 01	7.40E + 01	4.06E + 01	2.35E + 01	4.52E + 01	4.23E + 01	4.54E + 01	**2.10E + 01**
F_{12}	Mean	7.55E + 07	4.00E + 08	1.16E + 06	5.83E + 08	1.54E + 07	6.04E + 07	4.45E + 05	**1.24E + 04**	5.64E + 05	7.22E + 04	2.77E + 06	1.72E + 04
	Std	3.99E + 07	1.77E + 08	4.95E + 05	2.44E + 08	6.86E + 06	5.02E + 07	3.15E + 05	1.30E + 04	4.62E + 05	5.06E + 04	6.52E + 06	**1.26E + 04**
F_{13}	Mean	1.08E + 07	1.16E + 08	2.14E + 03	1.57E + 08	1.33E + 06	3.65E + 04	4.29E + 03	**8.84E + 01**	6.48E + 03	4.26E + 03	4.92E + 03	2.72E + 02
	Std	6.98E + 06	3.99E + 07	1.62E + 03	5.83E + 07	3.05E + 05	2.76E + 04	4.69E + 03	**4.19E + 01**	7.86E + 03	6.79E + 03	6.01E + 03	4.37E + 02
F_{14}	Mean	1.58E + 06	1.15E + 07	4.71E + 03	8.80E + 05	5.92E + 04	4.06E + 05	1.80E + 03	6.47E + 01	2.45E + 04	1.35E + 04	4.97E + 04	**5.38E + 01**
	Std	1.03E + 06	5.80E + 06	5.71E + 03	6.01E + 05	7.02E + 04	3.44E + 05	1.59E + 03	1.52E + 01	2.40E + 04	1.94E + 04	1.12E + 05	**1.13E + 01**
F_{15}	Mean	2.40E + 07	2.30E + 07	4.17E + 03	7.10E + 07	9.33E + 04	1.44E + 04	5.95E + 03	**7.23E + 01**	3.58E + 03	6.48E + 03	4.29E + 03	9.05E + 01
	Std	1.62E + 08	9.13E + 06	2.91E + 03	3.85E + 07	2.61E + 04	1.12E + 04	4.76E + 03	**2.64E + 01**	4.14E + 03	1.28E + 03	4.56E + 03	5.60E + 01
F_{16}	Mean	2.15E + 03	2.71E + 03	1.10E + 03	2.63E + 03	1.71E + 03	1.89E + 03	1.11E + 03	9.68E + 02	9.98E + 02	3.98E + 02	9.39E + 02	**9.16E + 02**
	Std	4.23E + 02	4.07E + 02	3.35E + 02	4.22E + 02	4.15E + 02	3.18E + 02	3.30E + 02	2.80E + 02	3.44E + 02	9.75E + 02	2.89E + 02	**2.21E + 02**
F_{17}	Mean	1.43E + 03	3.34E + 03	9.34E + 02	1.67E + 03	1.28E + 03	1.65E + 03	7.87E + 02	**5.47E + 02**	9.58E + 02	2.88E + 02	8.12E + 02	6.50E + 02
	Std	3.04E + 02	7.35E + 02	2.82E + 02	3.08E + 02	3.67E + 02	3.05E + 02	2.49E + 02	1.79E + 02	2.72E + 02	3.50E + 04	2.31E + 02	**1.64E + 02**
F_{18}	Mean	1.72E + 06	9.53E + 06	4.15E + 04	2.60E + 06	6.06E + 05	2.30E + 06	3.29E + 04	1.23E + 02	1.25E + 05	2.37E + 04	3.39E + 05	**1.18E + 02**
	Std	1.43E + 06	8.35E + 06	2.24E + 04	2.26E + 06	7.80E + 05	1.78E + 06	3.58E + 04	8.88E + 01	8.20E + 04	6.62E + 03	5.63E + 05	**7.56E + 01**
F_{19}	Mean	2.67E + 05	8.20E + 06	1.31E + 04	2.91E + 07	3.90E + 04	2.38E + 04	1.01E + 04	4.48E + 01	1.09E + 04	1.39E + 04	1.38E + 04	**2.40E + 01**

(*continued on next page*)

Table 8 (*continued*)

Function		SS	CSS	SSDE	SSCOL	CeSSOL	SS-MCA	BSA	MPEDE	MAPSO	TAPSO	XPSO	TCSSMH
	Std	2.93E + 05	2.22E + 06	4.96E + 03	1.25E + 07	1.91E + 04	3.62E + 04	7.22E + 03	2.50E + 01	6.34E + 03	8.13E + 03	8.71E + 03	**8.30E + 00**
F_{20}	Mean	8.55E + 02	1.50E + 03	7.02E + 02	1.41E + 03	1.00E + 03	1.09E + 03	7.63E + 02	**3.78E + 02**	5.03E + 02	5.87E + 02	5.19E + 02	4.08E + 02
	Std	2.24E + 02	2.84E + 02	2.46E + 02	3.08E + 02	3.59E + 02	2.22E + 02	4.25E + 02	1.81E + 02	2.10E + 02	2.93E + 02	2.46E + 02	**1.28E + 02**
F_{21}	Mean	8.11E + 02	1.37E + 03	2.96E + 02	6.48E + 02	5.69E + 02	6.06E + 02	3.34E + 02	**2.56E + 02**	2.70E + 02	3.28E + 02	2.90E + 02	3.28E + 02
	Std	1.44E + 02	2.21E + 02	2.09E + 01	7.02E + 01	7.68E + 01	5.77E + 01	3.16E + 01	**1.42E + 01**	1.51E + 01	2.89E + 01	1.87E + 01	2.80E + 01
F_{22}	Mean	7.41E + 03	8.95E + 03	4.99E + 03	9.81E + 03	7.62E + 03	6.45E + 03	1.13E + 04	3.30E + 03	4.85E + 03	5.19E + 03	3.95E + 03	**3.23E + 03**
	Std	7.80E + 02	**7.53E + 02**	2.07E + 03	9.70E + 02	9.19E + 02	8.52E + 02	1.41E + 03	2.63E + 03	1.74E + 03	1.51E + 03	2.90E + 03	2.20E + 03
F_{23}	Mean	1.22E + 03	1.77E + 03	5.38E + 02	1.04E + 03	1.11E + 03	8.97E + 02	5.84E + 02	4.84E + 02	4.97E + 02	6.04E + 02	5.35E + 02	**4.82E + 02**
	Std	2.38E + 02	1.31E + 02	2.42E + 01	1.56E + 02	1.97E + 02	9.38E + 01	3.76E + 01	**1.33E + 01**	1.39E + 01	3.57E + 01	5.10E + 01	1.89E + 01
F_{24}	Mean	1.22E + 03	2.08E + 03	6.15E + 02	1.60E + 03	1.42E + 03	1.06E + 03	6.51E + 02	**5.44E + 02**	5.62E + 02	7.05E + 02	6.55E + 02	5.97E + 02
	Std	1.67E + 02	1.46E + 02	2.89E + 01	2.36E + 02	2.25E + 02	9.12E + 01	4.21E + 01	**1.53E + 01**	1.76E + 01	4.77E + 01	9.21E + 01	2.26E + 01
F_{25}	Mean	6.22E + 02	6.80E + 02	5.67E + 02	7.28E + 02	5.46E + 02	5.86E + 02	5.79E + 02	5.23E + 02	5.24E + 02	5.42E + 02	5.98E + 02	**4.50E + 02**
	Std	2.89E + 01	3.53E + 01	4.50E + 01	7.43E + 01	2.65E + 01	2.71E + 01	2.80E + 01	3.33E + 01	2.24E + 01	3.72E + 01	3.01E + 01	**1.87E + 01**
F_{26}	Mean	5.37E + 03	1.12E + 04	2.87E + 03	5.49E + 03	4.27E + 03	5.22E + 03	3.48E + 03	1.59E + 03	1.89E + 03	2.49E + 03	1.18E + 03	**5.81E + 02**
	Std	1.41E + 03	1.50E + 03	4.33E + 02	1.60E + 03	1.97E + 03	1.84E + 03	7.61E + 02	**1.38E + 02**	1.70E + 02	1.04E + 03	9.54E + 02	5.06E + 02
F_{27}	Mean	8.37E + 02	1.51E + 03	7.65E + 02	1.12E + 03	7.57E + 02	7.83E + 02	9.94E + 02	5.47E + 02	5.70E + 02	6.24E + 02	7.16E + 02	**4.83E + 02**
	Std	2.60E + 02	5.63E + 02	6.02E + 01	4.57E + 02	1.96E + 02	1.17E + 02	1.84E + 02	2.68E + 01	4.63E + 01	4.74E + 01	9.21E + 01	**2.10E + 01**
F_{28}	Mean	5.72E + 02	6.77E + 02	5.13E + 02	7.25E + 02	4.95E + 02	5.42E + 02	2.37E + 03	4.91E + 02	4.83E + 02	4.95E + 02	5.44E + 02	**4.47E + 02**
	Std	3.51E + 01	7.56E + 01	2.42E + 01	3.48E + 02	1.88E + 01	4.20E + 01	2.80E + 03	2.34E + 01	2.42E + 01	2.12E + 01	3.41E + 01	**1.28E + 01**
F_{29}	Mean	2.11E + 03	2.95E + 03	8.34E + 02	2.11E + 03	1.80E + 03	2.17E + 03	1.03E + 03	**4.35E + 02**	7.71E + 02	1.02E + 03	9.11E + 02	5.98E + 02
	Std	4.23E + 02	4.39E + 02	2.48E + 02	5.62E + 02	6.22E + 02	4.40E + 02	3.77E + 02	**1.07E + 02**	2.57E + 02	2.89E + 02	2.38E + 02	1.42E + 02
F_{30}	Mean	4.00E + 06	1.05E + 08	9.26E + 05	6.17E + 07	2.64E + 06	4.93E + 06	6.96E + 05	7.02E + 05	8.43E + 05	7.52E + 05	1.99E + 06	**9.17E + 02**
	Std	1.19E + 06	8.83E + 07	9.27E + 04	2.30E + 07	8.98E + 05	6.96E + 06	1.12E + 05	9.29E + 04	1.14E + 05	7.39E + 04	5.28E + 05	**7.23E + 02**

the subpopulation.

4.4. Pattern search

The PS (Sahu, Panda, & Padhan, 2015) is a direct local search algorithm. Its core idea is to find out the optimal descending direction of function by calculating and comparing function values, so as to solve the objective optimization problem. PS is mainly composed of exploratory search and pattern search. The exploratory search is detected in the direction of the coordinate axis near a reference point to detect the favorable direction of the descent. A point (base point) that obtains a better value by detecting movement. Starting from the base point, the process of moving from the reference point to the base point is the pattern search. The PS does not need to solve the derivative of the objective function, so the PS has strong applicability, but it is difficult to quickly determine the falling direction of the objective function. The main steps of the pattern search method are as follows.

(1) **Initialization**: select an initial point $x(1)$, let $e_1, e_2, \cdots e_n$ be the coordinate directions, given initial step δ, and acceleration factor $\alpha = 1.4$, reduction rate $\beta = 0.2$, (α and β are calculated by the Taguchi method in Section 5) select a precision value $\varepsilon(\varepsilon = 1.0E-06)$ as the termination condition of the algorithm, let $y^{(1)} = x^{(1)}, k = 1, j = 1$.

(2) **Exploratory search**: if $f(y^{(j)} + \delta e_j) < f(y^{(j)})$, let $y^{(j+1)} = y^{(j)} + \delta e_j$, and go to step (4), otherwise, go to step (3).

(3) If $f(y^{(j)} - \delta e_j) < f(y^{(j)})$, let $y^{(j+1)} = y^{(j)} - \delta e_j$, and go to step (4), otherwise, $y^{(j+1)} = y^{(j)}$, and go to step (4).

(4) If $j < n$, set $j = j + 1$ and go to step (2), otherwise, go to step (5).

(5) If $f(y^{n+1}) < f(x^k)$, go to step (6), otherwise, go to step (7).

(6) **Pattern search**: let $x^{k+1} = y^{n+1}$, and let $y^{(1)} = x^{k+1} + \alpha(x^{k+1} - x^k)$, let $k = k + 1, j = 1$, go to step (2).

(7) If $\delta \le \varepsilon$, stop the iteration and get that point $x^{(k)}$ is the current optimal solution, otherwise, let $\delta = \beta\delta, y^{(1)} = x^{(k)}, x^{(k+1)} = x^{(k)}$, replace k by $k + 1$, let $j = 1$, and go to step (2).

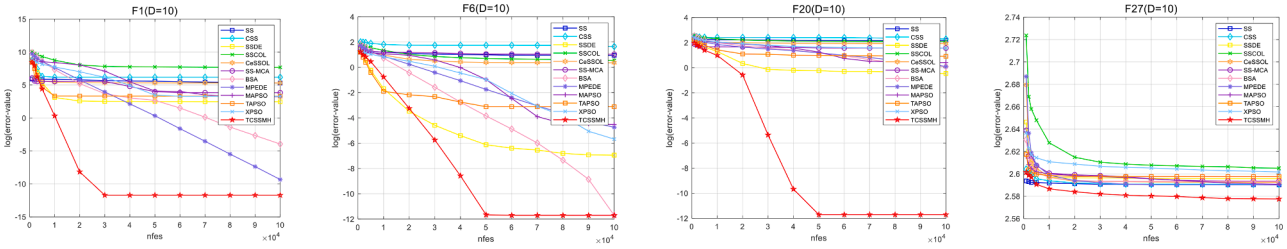
Table 9
Results and comparison of different algorithms on CEC2017 benchmark functions (100-dimension).

Function		SS	CSS	SSDE	SSCOL	CeSSOL	SS-MCA	BSA	MPEDE	MAPSO	TAPSO	XPSO	TCSSMH
F_1	Mean	6.52E + 08	2.57E + 09	4.43E + 03	1.98E + 11	1.12E + 09	1.66E + 09	3.78E + 03	1.70E-13	4.72E + 03	6.59E + 03	5.95E + 04	**0.00E + 00**
	Std	1.02E + 08	4.30E + 08	3.58E + 03	7.39E + 10	1.78E + 08	7.58E + 08	3.25E + 03	5.97E-13	5.74E + 03	6.90E + 03	3.66E + 05	**0.00E + 00**
F_2	Mean	1.00E + 30	1.00E + 30	8.34E + 11	1.00E + 30	1.00E + 30	1.00E + 30	1.00E + 30	6.31E + 16	3.28E + 51	2.16E + 15	2.89E + 86	**1.90E-06**
	Std	2.84E + 14	2.84E + 14	3.98E + 12	2.84E + 14	2.84E + 14	2.84E + 14	2.84E + 14	4.51E + 17	2.34E + 52	9.01E + 15	2.05E + 87	**7.93E-06**
F_3	Mean	1.19E + 05	1.04E + 06	7.55E-03	3.42E + 05	3.40E + 05	5.90E + 05	4.13E + 04	2.30E + 01	4.45E + 03	3.40E + 04	2.65E + 04	**0.00E + 00**
	Std	2.59E + 04	1.97E + 05	4.36E-03	7.29E + 04	6.67E + 04	4.21E + 04	9.36E + 03	6.93E + 01	1.58E + 03	1.26E + 04	6.80E + 03	**0.00E + 00**
F_4	Mean	5.98E + 02	1.05E + 03	2.58E + 02	1.55E + 04	3.71E + 02	5.80E + 02	2.88E + 02	8.53E + 01	2.67E + 02	1.79E + 02	4.95E + 02	1.08E + 02
	Std	7.88E + 01	2.85E + 02	4.92E + 01	1.64E + 04	4.71E + 01	8.68E + 01	5.54E + 01	6.67E + 01	2.36E + 01	5.15E + 01	5.63E + 01	7.00E + 01
F_5	Mean	1.34E + 03	2.25E + 03	3.45E + 02	1.08E + 03	9.34E + 02	1.24E + 03	5.71E + 02	1.47E + 02	1.55E + 02	3.62E + 02	2.34E + 02	5.07E + 02
	Std	1.72E + 02	2.34E + 02	3.66E + 01	9.57E + 01	1.08E + 02	1.74E + 02	8.23E + 01	2.53E + 01	2.52E + 01	4.90E + 01	4.28E + 01	1.00E + 02
F_6	Mean	5.14E + 01	1.07E + 02	4.57E-02	4.59E + 01	4.65E + 01	7.05E + 01	1.60E + 01	1.50E-01	1.90E + 00	7.63E-02	5.88E + 00	**0.00E + 00**
	Std	7.90E + 00	1.59E + 01	6.84E-02	7.04E + 00	6.50E + 00	6.20E + 00	6.02E + 00	1.13E-01	4.51E-01	2.65E-02	2.14E + 00	**0.00E + 00**
F_7	Mean	6.81E + 02	2.26E + 03	4.94E + 02	2.02E + 03	8.66E + 02	1.28E + 03	1.15E + 03	**3.07E + 02**	4.82E + 02	5.49E + 02	4.68E + 02	5.56E + 02
	Std	1.05E + 02	4.24E + 02	6.99E + 01	7.25E + 02	7.27E + 01	2.08E + 02	1.50E + 02	**3.54E + 01**	7.95E + 01	7.09E + 01	8.03E + 01	1.00E + 02
F_8	Mean	1.20E + 03	2.32E + 03	3.47E + 02	1.13E + 03	9.17E + 02	1.24E + 03	5.92E + 02	**1.45E + 02**	1.60E + 02	3.79E + 02	2.18E + 02	4.51E + 02
	Std	1.20E + 02	2.13E + 02	4.11E + 01	9.43E + 01	1.01E + 02	1.74E + 02	7.85E + 01	**2.23E + 01**	2.86E + 01	6.48E + 01	4.19E + 01	9.37E + 01
F_9	Mean	4.85E + 04	1.32E + 05	4.27E + 03	4.76E + 04	4.52E + 04	5.15E + 04	1.35E + 04	**3.92E + 01**	8.35E + 02	6.26E + 03	7.74E + 02	6.11E + 03
	Std	1.17E + 04	1.90E + 04	2.06E + 03	6.74E + 03	7.55E + 03	5.61E + 03	1.98E + 03	**5.14E + 01**	2.66E + 02	2.35E + 03	3.10E + 02	2.32E + 03
F_{10}	Mean	1.58E + 04	2.09E + 04	1.28E + 04	2.24E + 04	1.72E + 04	1.50E + 04	2.52E + 04	1.12E + 04	1.15E + 04	1.20E + 04	1.25E + 04	**1.11E + 04**
	Std	1.36E + 03	1.90E + 03	1.16E + 03	1.64E + 03	1.82E + 03	1.31E + 03	4.10E + 03	1.29E + 03	1.10E + 03	1.17E + 03	1.17E + 03	**1.04E + 03**
F_{11}	Mean	3.65E + 03	2.93E + 05	9.93E + 02	3.93E + 04	1.61E + 04	7.88E + 04	6.40E + 02	7.88E + 02	7.80E + 02	3.06E + 02	1.30E + 03	**2.47E + 02**
	Std	1.27E + 03	3.08E + 05	1.39E + 02	2.68E + 04	7.63E + 03	2.69E + 04	1.55E + 02	2.29E + 02	1.71E + 02	8.81E + 01	2.44E + 02	**5.61E + 01**
F_{12}	Mean	4.80E + 08	1.75E + 09	4.07E + 06	1.43E + 10	1.33E + 08	2.38E + 08	1.95E + 06	**3.32E + 04**	2.51E + 06	4.26E + 05	3.19E + 07	8.65E + 04
	Std	1.13E + 08	3.02E + 08	1.67E + 06	1.65E + 10	4.95E + 07	1.06E + 08	7.30E + 05	**1.72E + 04**	1.76E + 06	3.25E + 05	2.72E + 07	4.71E + 04
F_{13}	Mean	5.41E + 07	3.73E + 08	4.69E + 03	3.28E + 08	7.41E + 06	1.43E + 05	4.33E + 03	7.33E + 03	5.13E + 03	4.28E + 03	6.61E + 03	**2.67E + 03**
	Std	1.47E + 07	5.82E + 07	1.77E + 03	9.67E + 07	4.81E + 06	4.38E + 05	2.66E + 03	4.06E + 03	5.21E + 03	5.14E + 03	6.28E + 03	**9.09E + 02**
F_{14}	Mean	6.63E + 06	3.55E + 07	5.22E + 04	5.85E + 06	1.53E + 06	4.76E + 06	3.14E + 04	4.83E + 02	1.22E + 05	3.68E + 04	4.66E + 05	**4.38E + 02**
	Std	2.74E + 06	6.90E + 06	2.68E + 04	2.46E + 06	1.01E + 06	2.53E + 06	1.74E + 04	3.18E + 02	6.55E + 04	3.60E + 04	7.76E + 05	**1.33E + 02**
F_{15}	Mean	1.34E + 07	1.36E + 08	**9.99E + 02**	1.23E + 08	2.25E + 06	1.53E + 05	1.66E + 03	3.73E + 03	2.36E + 03	1.68E + 03	4.04E + 03	1.02E + 03
	Std	5.03E + 06	4.98E + 07	1.11E + 03	3.65E + 07	6.45E + 05	7.37E + 05	2.11E + 03	**2.03E + 02**	2.69E + 03	2.30E + 03	6.34E + 03	1.34E + 03
F_{16}	Mean	5.51E + 03	7.43E + 03	3.21E + 03	6.93E + 03	5.38E + 03	5.26E + 03	3.68E + 03	**2.74E + 03**	2.77E + 03	3.28E + 03	2.94E + 03	3.05E + 03
	Std	9.32E + 02	7.72E + 02	**4.98E + 02**	8.92E + 02	8.30E + 02	6.36E + 02	5.98E + 02	5.08E + 02	6.20E + 02	6.56E + 02	6.78E + 02	5.01E + 02
F_{17}	Mean	4.20E + 03	6.67E + 03	2.44E + 03	4.47E + 03	3.66E + 03	4.22E + 03	2.65E + 03	2.02E + 03	2.53E + 03	2.55E + 03	2.52E + 03	**1.97E + 03**
	Std	7.86E + 02	7.93E + 02	5.05E + 02	6.02E + 02	4.86E + 02	5.51E + 02	4.63E + 02	3.81E + 02	4.69E + 02	5.67E + 02	5.10E + 02	**2.83E + 02**
F_{18}	Mean	1.09E + 07	4.06E + 07	1.23E + 05	7.06E + 06	2.72E + 06	7.01E + 06	1.49E + 05	**1.09E + 03**	2.59E + 05	1.05E + 05	3.46E + 05	2.66E + 03
	Std	3.70E + 06	2.71E + 07	3.89E + 04	3.86E + 06	1.49E + 06	3.38E + 06	6.92E + 04	**1.13E + 03**	9.51E + 04	5.83E + 04	2.01E + 05	1.89E + 03
F_{19}	Mean	9.32E + 06	1.23E + 08	1.20E + 03	1.18E + 08	2.60E + 06	8.14E + 05	2.18E + 03	**2.28E + 02**	2.65E + 03	2.32E + 03	4.16E + 03	4.75E + 02

(*continued on next page*)

Table 9 (*continued*)

Function		SS	CSS	SSDE	SSCOL	CeSSOL	SS-MCA	BSA	MPEDE	MAPSO	TAPSO	XPSO	TCSSMH
	Std	5.06E + 06	4.45E + 07	1.18E + 03	3.35E + 07	1.84E + 06	1.10E + 06	2.39E + 03	**5.23E + 01**	2.67E + 03	2.84E + 03	4.82E + 03	4.75E + 02
F_{20}	Mean	3.46E + 03	5.63E + 03	2.46E + 03	4.36E + 03	3.36E + 03	3.18E + 03	2.54E + 03	1.85E + 03	2.08E + 03	2.25E + 03	2.26E + 03	**1.82E + 03**
	Std	5.26E + 02	5.04E + 02	4.90E + 02	6.29E + 02	5.63E + 02	**3.12E + 02**	7.62E + 02	4.34E + 02	4.00E + 02	5.45E + 02	5.30E + 02	1.88E + 03
F_{21}	Mean	2.18E + 03	3.38E + 03	5.28E + 02	1.48E + 03	1.25E + 03	1.58E + 03	6.64E + 02	**3.66E + 02**	3.73E + 02	6.11E + 02	4.66E + 02	5.67E + 02
	Std	3.38E + 02	5.95E + 02	4.25E + 01	1.47E + 02	1.76E + 02	1.37E + 02	5.57E + 01	2.56E + 01	**2.02E + 01**	5.70E + 01	4.60E + 01	5.77E + 01
F_{22}	Mean	1.69E + 04	2.20E + 04	1.41E + 04	2.37E + 04	1.90E + 04	1.88E + 04	2.71E + 04	1.22E + 04	1.32E + 04	1.29E + 04	**1.20E + 04**	1.22E + 04
	Std	1.45E + 03	1.32E + 03	1.20E + 03	1.63E + 03	1.59E + 03	**8.53E + 02**	3.06E + 03	1.07E + 03	1.27E + 03	1.27E + 03	5.37E + 03	1.06E + 03
F_{23}	Mean	2.60E + 03	3.27E + 03	7.96E + 02	2.12E + 03	1.99E + 03	1.89E + 03	9.26E + 02	7.01E + 02	**6.60E + 02**	8.19E + 02	8.29E + 02	6.99E + 02
	Std	5.42E + 02	3.69E + 02	4.42E + 01	3.25E + 02	4.09E + 02	1.47E + 02	7.46E + 01	3.56E + 01	**2.38E + 01**	4.42E + 01	5.95E + 01	2.50E + 01
F_{24}	Mean	3.29E + 03	3.24E + 03	1.34E + 03	2.83E + 03	2.14E + 03	2.57E + 03	1.59E + 03	1.21E + 03	1.05E + 03	1.34E + 03	1.26E + 03	**1.03E + 03**
	Std	5.02E + 02	7.05E + 02	8.00E + 01	1.33E + 03	2.98E + 02	2.01E + 02	2.22E + 02	4.83E + 01	**2.51E + 01**	6.06E + 01	9.70E + 01	2.70E + 01
F_{25}	Mean	1.17E + 03	1.33E + 03	8.17E + 02	4.39E + 03	9.60E + 02	1.61E + 03	8.24E + 02	7.49E + 02	8.18E + 02	7.64E + 02	1.08E + 03	**7.40E + 02**
	Std	4.12E + 01	1.36E + 02	5.23E + 01	4.52E + 03	5.28E + 01	9.71E + 01	5.40E + 01	5.32E + 01	**3.88E + 01**	7.26E + 01	7.91E + 01	4.35E + 01
F_{26}	Mean	1.76E + 04	2.84E + 04	1.03E + 04	1.92E + 04	1.34E + 04	2.02E + 04	1.14E + 04	**4.49E + 03**	4.79E + 03	8.07E + 03	4.63E + 03	7.47E + 03
	Std	1.85E + 03	2.96E + 03	1.97E + 03	2.84E + 03	1.79E + 03	1.63E + 03	2.27E + 03	**2.77E + 02**	3.09E + 02	9.32E + 02	2.29E + 03	6.98E + 02
F_{27}	Mean	1.34E + 03	2.80E + 03	1.11E + 03	1.47E + 03	8.29E + 02	1.47E + 03	1.20E + 03	6.95E + 02	6.94E + 02	8.31E + 02	8.71E + 02	**4.85E + 02**
	Std	3.40E + 02	1.27E + 03	1.02E + 02	3.69E + 02	1.10E + 02	1.47E + 02	6.50E + 02	3.10E + 01	3.27E + 01	8.08E + 01	6.34E + 01	**9.92E + 00**
F_{28}	Mean	9.15E + 02	9.95E + 02	6.10E + 02	5.68E + 03	6.59E + 02	2.28E + 03	4.36E + 03	5.34E + 02	6.05E + 02	5.52E + 02	8.28E + 02	**4.83E + 02**
	Std	5.99E + 01	1.06E + 02	3.71E + 01	5.18E + 03	3.90E + 01	3.34E + 02	8.68E + 03	4.15E + 01	3.13E + 01	**2.67E + 01**	4.38E + 01	8.55E + 01
F_{29}	Mean	5.50E + 03	7.46E + 03	3.28E + 03	5.38E + 03	4.59E + 03	7.20E + 03	3.55E + 03	2.43E + 03	2.84E + 03	3.13E + 03	3.05E + 03	**2.27E + 03**
	Std	6.00E + 02	1.82E + 03	4.94E + 02	7.85E + 02	5.99E + 02	6.29E + 02	4.74E + 02	4.21E + 02	4.88E + 02	3.85E + 02	5.15E + 02	**3.61E + 02**
F_{30}	Mean	7.35E + 07	1.59E + 08	2.28E + 04	3.81E + 08	1.54E + 07	3.82E + 08	7.98E + 03	**2.62E + 03**	6.43E + 03	6.59E + 03	4.27E + 04	3.62E + 03
	Std	7.87E + 07	4.16E + 07	1.69E + 04	1.45E + 08	1.02E + 07	6.89E + 07	3.47E + 03	**2.25E + 02**	2.88E + 03	4.42E + 03	3.44E + 04	3.12E + 03



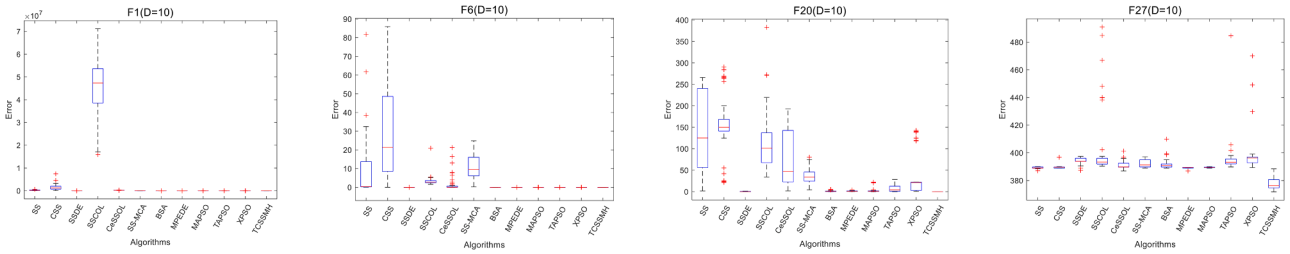
(a) F_1 (b) F_6 (c) F_{20} (d) F_{27}

Fig. 6. Convergence curves of the algorithms on 10-dimension.

The search efficiency of PS is greatly affected by the position of the initial point. For different initial points, the accuracy and speed of PS optimization will have a great impact, so ensuring a better initial position is one of the keys to ensuring efficient optimization. In this paper, multiple sub-populations are used for global search. After the search of each sub-population, the optimal individual of each sub-population and *Pbest_rate*e* (these parameters need to be determined through calibration) individuals from the elite *RefSet* as the initial point for pattern search. A small number of individuals are selected for fine exploration, which not only saves computational resources but also improves the ability of the algorithm to jump out of the local optimal solution.

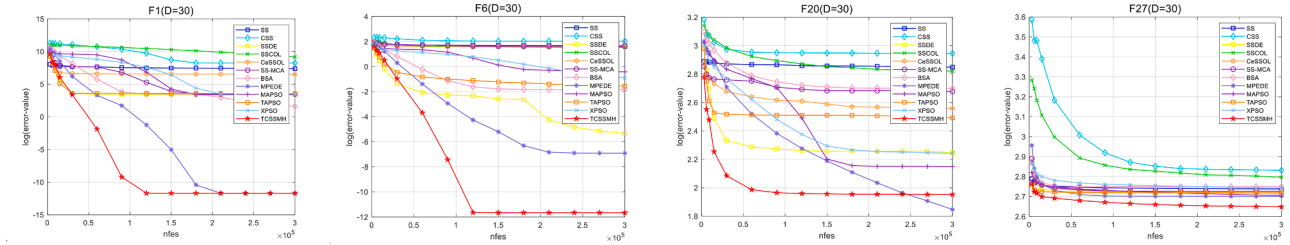
4.5. The framework of TCSSMH

Based on the above analysis, the complete pseudo code process of TCSSMH is shown in Algorithm 4, the code of this article can be



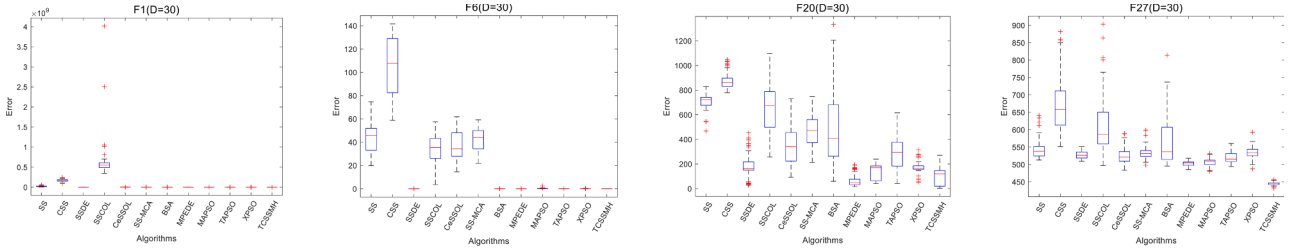
(a) F_1 (b) F_6 (c) F_{20} (d) F_{27}

Fig. 7. Box plots of the algorithms on 10-dimension.



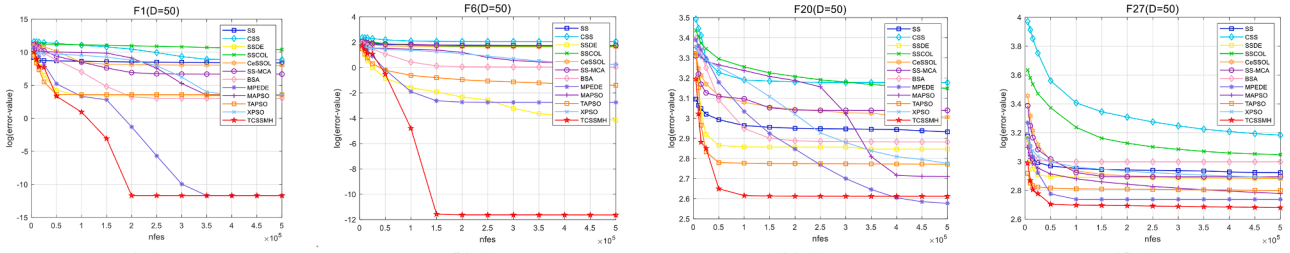
(a) F_1 (b) F_6 (c) F_{20} (d) F_{27}

Fig. 8. Convergence curves of the algorithms on 30-dimension.



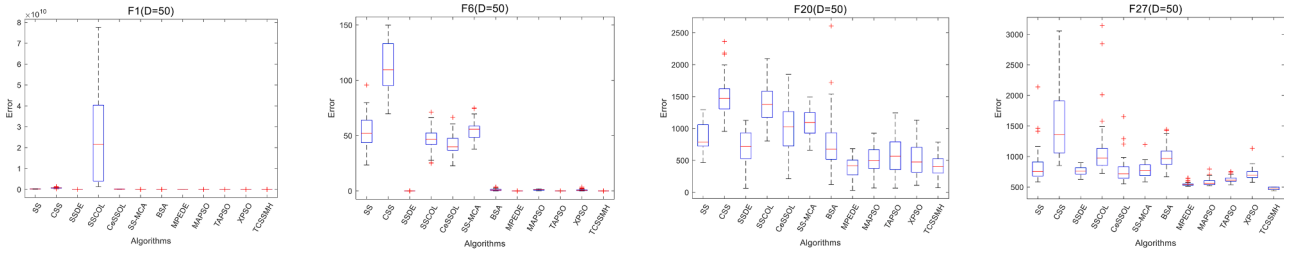
(a) F_1 (b) F_6 (c) F_{20} (d) F_{27}

Fig. 9. Box plots of the algorithms on 30-dimension.



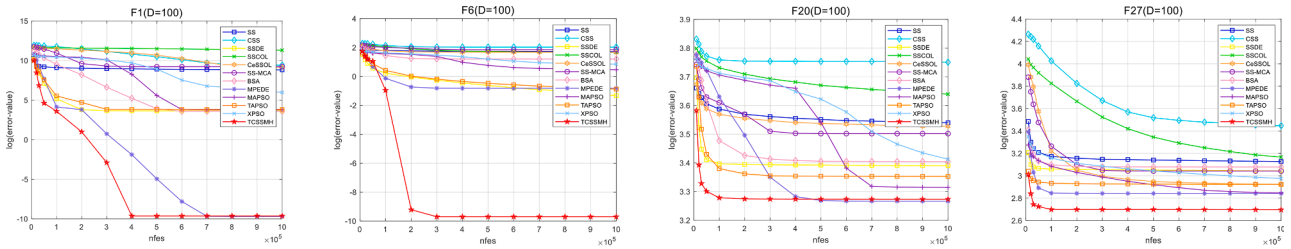
(a) F_1 (b) F_6 (c) F_{20} (d) F_{27}

Fig. 10. Convergence curves of the algorithms on 50-dimension.



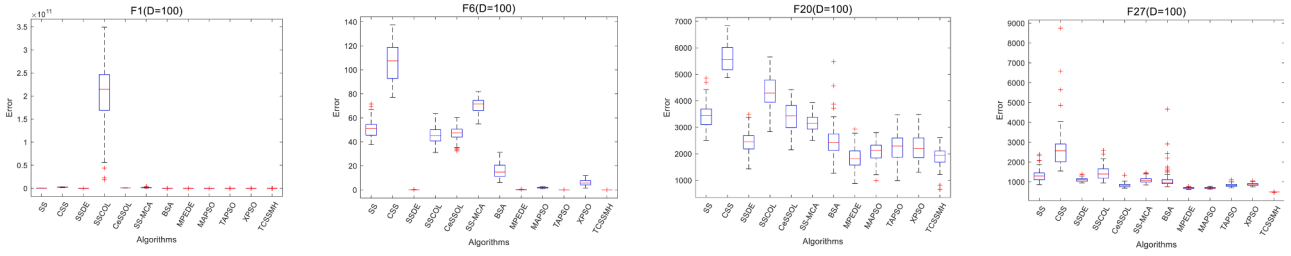
(a) F_1 (b) F_6 (c) F_{20} (d) F_{27}

Fig. 11. Box plots of the algorithms on 50-dimension.



(a) F_1 (b) F_6 (c) F_{20} (d) F_{27}

Fig. 12. Convergence curves of the algorithms on 100-dimension.



(a) F_1 (b) F_6 (c) F_{20} (d) F_{27}

Fig. 13. Box plots of the algorithms on 100-dimension.

downloaded from the Code Ocean platform https://codeocean.com/, and its DOI is https://doi.org/10.24433/CO.1182393.v1.

Algorithm 4: The framework of TCSSMH

```
Input: Max_nfes, RefSet size b, D, Subpopulation size (n);
/* the elite RefSet size e; n is the number of subpopulations */
Output: The optimal individual and fitness.
Initialize the population using the Diversification Generation Method;
Refset Update Method;
Individuals in the RefSet were arranged in ascending order of fitness value and
divided into n subpopulations;
while the stop condition is not satisfied do
  Select the best e elite individuals to form the elite Refset;
  Update elite Refset according to Algorithm 2;
  Divide elite individuals and assign them to each subgroup;
  for i = 1 to n do
    Each subpopulation is updated according to Eqs (13)–(15);
    Select the inferior individual of each subgroup to update according to
Algorithm 3;
  end for
  Calculate population fitness variance σ^2;
  if σ^2 < C then
    Calculate the optimal individuals in each subgroup and elite Refset;
    Apply PS to the selected extreme point for local search;
    Regroup subpopulation;
  end if
end while
```

4.6. Time complexity analysis of TCSSMH

The computational complexity of each operation in TCSSMH is discussed and compared with that of the corresponding operation in fundamental SS in this section. According to the CEC2017 test set standard, the maximum number of evaluations of the fitness value is set to D*10000. Let the initial population size be N, the dimension of the problem to be D. The time complexity of each component of the original SS is analyzed as follows.

(1) The complexity of the diversified initial population method is $O(N)$.
(2) Evaluating the fitness of the initial population costs $O(N)$.
(3) Improving the initial solution by the solution improvement method cost $O(D*N^2)$.
(4) Generate initial Refset costs $O(D*b)$.
(5) Subset generation method costs $O(D*b)$.
(6) Subset combination method costs $O(D*b^2)$.

The overall time complexity of SS is $2O(N) + O(D*N^2) + 2O(D*b) + O(D*b^2)$.

The detailed calculation steps of the time complexity of the proposed TCSSMH are as follows. The Refset in TCSSMH is divided into three parts N_1, N_2, N_3, and $N_1 + N_2 + N_3 = b$.

(1) The initialization of population needs $O(N)$.
(2) The complexity of calculating the initial population fitness value is $O(N)$.
(3) The time consumed for dividing the Refset into subpopulations is $O(N_1) + O(N_2) + O(N_3) = O(b)$.
(4) The cost of generating new individuals for each subpopulation is $O(D*N_1) + O(D*N_2) + O(D*N_3) = O(D*b)$.
(5) The cost of evaluating the fitness value of each subpopulation is $O(D*N_1) + O(D*N_2) + O(D*N_3) = O(D*b)$.
(6) Using cooperative interactive learning mechanism to update individual needs $O(D*b)$.
(7) The time complexity of updating individuals using pattern search is $O(D*b^2)$.

The overall time complexity of TCSSMH is $2O(N) + O(b) + 3O(D*b) + O(D*b^2)$.

The proposed TCSSMH removes the operation of improving the initial solution, so the time complexity is relatively reduced by $O(D*N^2)$. However, the time complexity of pattern search increases $O(D*b^2)$ in the process of updating the solution. Overall, TCSSMH does not increase computational complexity over the original SS.

5. Experimental results and analysis

In this section, all the gained results will be presented, and then the observation results will be comprehensively and deeply analyzed.

5.1. Experimental environment and parameter settings

In this research, the CEC2017 benchmark test suite is employed to

Table 10
p-value of Wilcoxon's rank-sum test for $10D, 30D, 50D$, and..$100D$

Dimension	Algorithms	VS.	$R+$	$R-$	$+$	$\approx$	$-$	Z	*P-value*	$\alpha = 0.05$	$\alpha = 0.1$
10	TCSSMH	SS	463	2	29	0	1	−4.74	2.00E-06	**Yes**	**Yes**
		CSS	465	0	30	0	0	−4.78	2.00E-06	**Yes**	**Yes**
		SSDE	406	0	28	2	0	−4.78	4.00E-06	**Yes**	**Yes**
		SSCOL	465	0	30	0	0	−4.78	2.00E-06	**Yes**	**Yes**
		CeSSOL	465	0	30	0	0	−4.78	2.00E-06	**Yes**	**Yes**
		SS-MCA	389	46	26	1	3	−3.71	2.09E-04	**Yes**	**Yes**
		BSA	406	0	28	2	0	−4.62	4.00E-06	**Yes**	**Yes**
		MPEDE	393	13	27	2	1	−4.33	1.50E-05	**Yes**	**Yes**
		MAPSO	406	0	30	0	0	−4.62	4.00E-06	**Yes**	**Yes**
		TAPSO	465	0	30	0	0	−4.78	2.00E-06	**Yes**	**Yes**
		XPSO	465	0	30	0	0	−4.78	2.00E-06	**Yes**	**Yes**
30	TCSSMH	SS	465	0	30	0	0	−4.74	2.00E-06	**Yes**	**Yes**
		CSS	465	0	30	0	0	−4.78	2.00E-06	**Yes**	**Yes**
		SSDE	405	30	25	1	4	−4.05	5.00E-05	**Yes**	**Yes**
		SSCOL	465	0	30	0	0	−4.78	2.00E-06	**Yes**	**Yes**
		CeSSOL	465	0	30	0	0	−4.78	2.00E-06	**Yes**	**Yes**
		SS-MCA	465	0	30	0	0	−4.78	2.00E-06	**Yes**	**Yes**
		BSA	465	0	30	0	0	−4.78	2.00E-06	**Yes**	**Yes**
		MPEDE	253	153	15	2	13	−1.14	2.55E-01	NO	NO
		MAPSO	435	30	26	0	4	−4.18	3.00E-05	**Yes**	**Yes**
		TAPSO	465	0	30	0	0	−4.78	2.00E-06	**Yes**	**Yes**
		XPSO	450	15	28	0	2	−4.47	8.00E-06	**Yes**	**Yes**
50	TCSSMH	SS	465	0	30	0	0	−4.78	2.00E-06	**Yes**	**Yes**
		CSS	465	0	30	0	0	−4.78	2.00E-06	**Yes**	**Yes**
		SSDE	431	34	25	0	5	−4.08	4.40E-05	**Yes**	**Yes**
		SSCOL	465	0	30	0	0	−4.78	2.00E-06	**Yes**	**Yes**
		CeSSOL	465	0	30	0	0	−4.78	2.00E-06	**Yes**	**Yes**
		SS-MCA	465	0	30	0	0	−4.78	2.00E-06	**Yes**	**Yes**
		BSA	465	0	30	0	0	−4.78	2.00E-06	**Yes**	**Yes**
		MPEDE	222	243	17	0	13	−0.22	8.29E-01	No	No
		MAPSO	430	35	25	0	5	−4.06	4.90E-05	**Yes**	**Yes**
		TAPSO	421	14	26	1	3	−4.40	1.10E-05	**Yes**	**Yes**
		XPSO	432	33	25	0	5	−4.10	4.10E-05	**Yes**	**Yes**
100	TCSSMH	SS	465	0	30	0	0	−4.78	2.00E-06	**Yes**	**Yes**
		CSS	465	0	30	0	0	−4.78	2.00E-06	**Yes**	**Yes**
		SSDE	412	53	24	0	6	−3.69	2.00E-06	**Yes**	**Yes**
		SSCOL	465	0	30	0	0	−4.78	2.00E-06	**Yes**	**Yes**
		CeSSOL	465	0	30	0	0	−4.78	2.00E-06	**Yes**	**Yes**
		SS-MCA	465	0	30	0	0	−4.78	2.00E-06	**Yes**	**Yes**
		BSA	465	0	30	0	0	−4.78	2.00E-06	**Yes**	**Yes**
		MPEDE	200	235	16	1	13	−0.38	7.05E-01	No	No
		MAPSO	336	99	22	0	8	−2.56	1.04E-02	**Yes**	**Yes**
		TAPSO	445	20	27	0	3	−4.37	1.20E-05	**Yes**	**Yes**
		XPSO	361	74	21	0	8	−3.10	1.92E-03	**Yes**	**Yes**

verify the performance of the proposed TCSSMH. The CEC2017 is the latest set of benchmark functions involving Unimodal Functions(F_1 F_3), Simple Multimodal Functions(F_4 F_{10}), Hybrid Functions(F_{11} F_{20}), and Composition Functions(F_{21} F_{30}). For more detailed information, please see Ref (Awad, Ali, Liang, Qu, & Suganthan, 2016). All experiments are executed by MATLAB 2020a and run on a PC with a 3.4 GHz Intel, Core I i7-6700 CPU, 8 GB of RAM, and 64-bit Operating System. In order to prevent the error caused by the randomness of the algorithm, all experiments carry out 51 independent operations and then use the average value of 51 times as the data for evaluating the performance. Evaluating the performance refers to comparing the final optimal solution and stability of each algorithm after the specified evaluation times. For a given dimension ($D = 10,30,50,100$), each test function runs for a fixed number of times. The maximum number of fitness evaluations is set to 10000*D (D is the dimension of the problem), and the error value of mean and standard deviation smaller than 10^{-8} will be taken as zero.

In order to verify the advantages of TCSSMH, a series of testing experiments are described here. The original SS and Backtracking search algorithm (BSA) (Civicioglu, 2013). Five state-of-the-art SS variants, including CSS (Herrera et al., 2006), SSDE (Li & Tian, 2015), SSCOL (Remli et al., 2017), CeSSOL (Remli et al., 2019), SS-MCA (Abd Alradha Abd Alradha Alsaidi et al., 2020). Meanwhile, four other algorithms are mentioned above (MPEDE (Wu, Mallipeddi, Suganthan, Wang, & Chen, 2016), MAPSO (Wei et al., 2020), TAPSO (Wei et al., 2020), XPSO (Xia, Gui, He, et al., 2020)) were selected for comparative study. The algorithms are tested on the CEC2017 benchmark test suite. The experimental results of the above twelve algorithms are compared under $D = 10, D = 30, D = 50$, and $D = 100$ respectively. Parameter settings of the selected peer algorithms summarized in Table 1 are the same as that in the corresponding papers.

5.2. Sensitivity analysis of user-defined parameters

The reasonable parameters of the algorithm have an important impact on its performance. To more extensively analyze the performance of our algorithm, the user-defined parameters are worth investigating. We proceed by the sensitivity analysis of the main parameters affecting the TCSSMH optimal performance. There are five user-defined parameters in TCSSMH: b (the size of the *RefSet*), e (the number of individuals in elite *RefSet*), *Pbest_rate* (the ratio of fine search individuals selected from the elite *RefSet* in the pattern search algorithm), α is the acceleration factor, β is the reduction factor.

In this experiment, Taguchi's method (Gao et al., 2019) is used to obtain the best combination of five TCSSMH parameters. The Taguchi method has targeted detection of some possible combinations of factors, rather than the entire combination, so as to attain the best estimate of the influencing experimental parameters with the fewest number of experimental runs. The specific level settings of the five parameters are

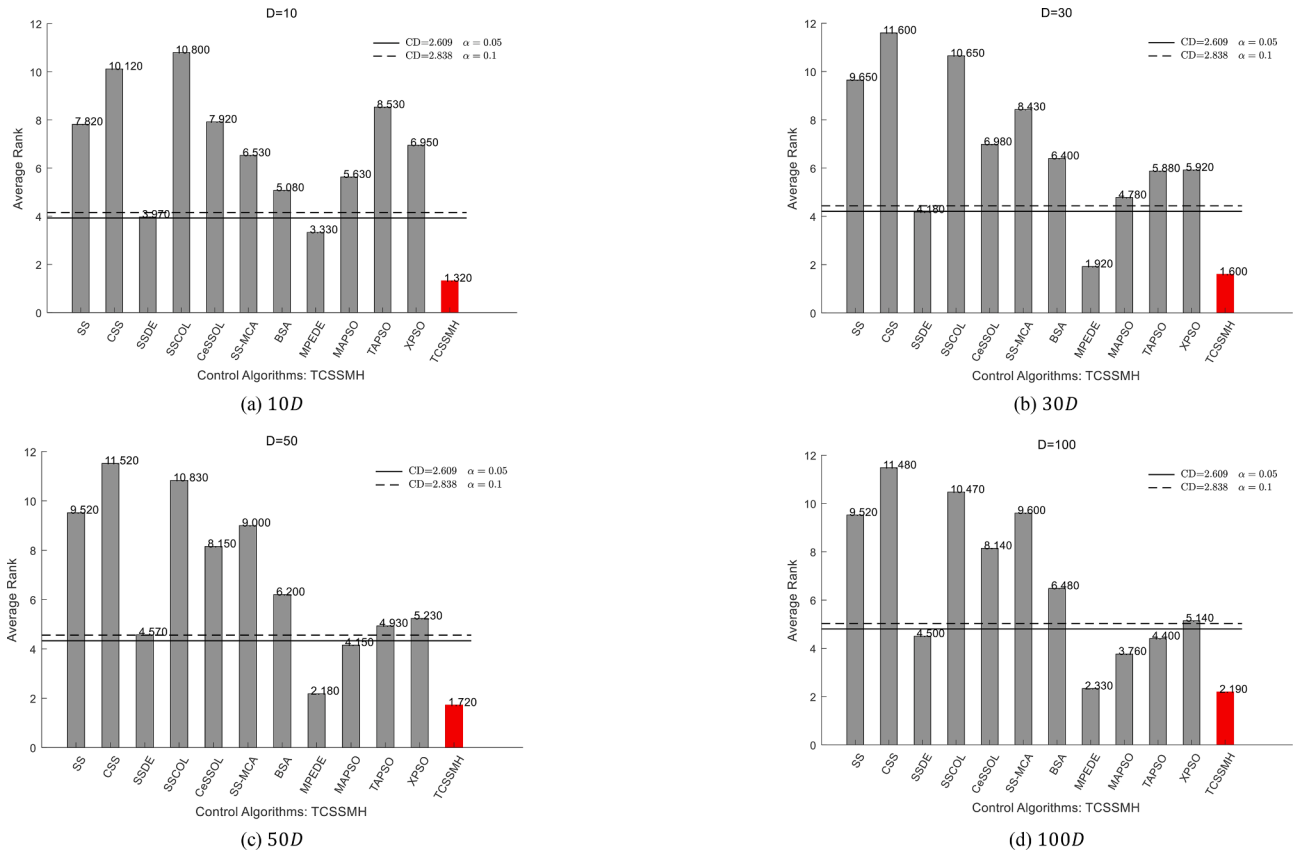


Fig. 14. Rankings obtained through Friedman test.

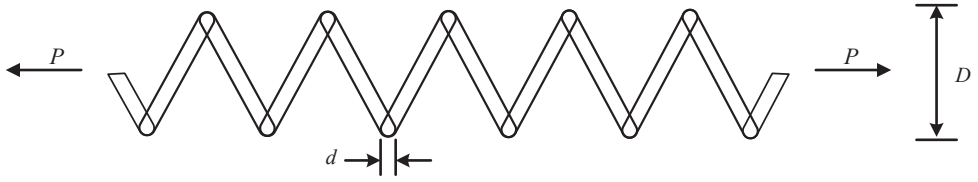


Fig. 15. Tension/compression string design problem.

Table 11
Comparison of the best solution by different algorithms.

Algorithms	DELC	DEDS	CPSO	HPSO	HEAA	TCSSMH
x_1	0.051689	0.051689	0.051728	0.051706	0.051689	0.051656
x_2	0.356717	0.356717	0.357644	0.357126	0.356729	0.355000
x_3	11.288965	11.288965	11.244543	11.265083	11.288293	11.344665
$g_1(x)$	1.56E-06	1.56E-06	−8.25E-04	−3.07E-06	−3.98E-05	6.98E-03
$g_2(x)$	1.65E-06	1.65E-06	−2.53E-05	1.39E-06	2.87E-05	−2.09E-03
$g_3(x)$	−4.053801	−4.053801	−4.051307	−4.054583	−4.053762	−4.074510
$g_4(x)$	−0.727729	−0.727729	−0.727085	−0.727445	−0.727721	−0.728896
$f(x)$	0.012665	0.012665	0.012675	0.012665	0.012665	**0.012641**

Table 12
Statistical results of different algorithms.

	DELC	DEDS	CPSO	HPSO	HEAA	TCSSMH
Best	0.126652	0.126652	0.012675	0.012665	0.012665	**0.012641**
Worst	0.012666	0.012738	0.012924	0.012719	0.012665	**0.012648**
Mean	0.012665	0.012669	0.012730	0.012707	0.012665	**0.012645**
Std	1.30E-07	1.30E-05	5.20E-04	1.58E-05	1.40E-09	**4.70E-15**

as follows: five levels for $b \in \{20, 30, 50, 80, 100\}$; five levels for $e \in \{5, 10, 15, 20, 25\}$; five levels for $Pbest\_\text{rate} \in \{0.1, 0.2, 0.3, 0.4, 0.5\}$; five levels for $\alpha \in \{1.1, 1.2, 1.3, 1.4, 1.5\}$; and five levels for $\beta \in \{0.1, 0.2, 0.3, 0.4, 0.5\}$. Through different parameter combinations, there are $5^5 = 3125$ experimental results. For the all-factor experimental method, all possible combinations must be studied deeply. Taguchi method uses an orthogonal table, which effectively reduces the number of experiments and saves time. Therefore, an orthogonal table $L_{25}(5^5)$ that contains only 25 experiments is adopted in our experiment. In order to make the experiment statistically significant, the functions of different dimensions (D = 10, D = 30, and D = 50) are selected as test cases. At the same time, in order to avoid the error caused by the randomness of the algorithm, each function runs each parameter group 30 times independently.

The different combinations of parameters were shown in Table 2. The parameter combinations and average error values yielded by TCSSMH are listed in Table 3. According to Table 3, the average error values (AVE) and total average error values (TAVE) show the performance of each combination. The levels of each parameter are shown in Table 4. Meanwhile, the tendency of the parameters is described in Fig. 3.

The impact of the five parameters on the performance of the algorithm is shown in Table 4. Note that, among these five parameters, b has the most significant impact on the algorithm. The Refset is divided into several subpopulations. The individual of each sub-population plays a key role in maintaining the diversity of the population in the process of searching different fields. It can be seen that the performance is best when b value is 50. The second-ranked parameter e illustrated that it is still an important factor affecting the performance of the algorithm. The parameter e plays a pivotal role in leading the direction of population evolution and balancing the exploration and exploitation capabilities of the algorithm. The horizontal axis of Fig. 3 represents the value level of different parameters, and the vertical axis represents the total average error value. It can be seen from the parameter change trend diagram in Fig. 3 that $e = 15$ is the most reasonable choice. The individual ratio value *Pbest_rate* selected for refined search in the elite Refset ranks third. *Pbest_rate* has a great influence on the convergence speed of the algorithm, and a reasonable value can accelerate the speed of TCSSMH to find the optimal solution. The fourth parameter of the ranking is α and the fifth parameter is β. These two parameters are used to control the search step size in the pattern search method, and reasonable values can effectively improve the quality of the solution. In summary, according to the results of the Taguchi method, the parameters in TCSSMH were suggested as follows: $b = 50, e = 15, Pbest\_rate = 0.3, \alpha = 1.4, \beta = 0.2$.

5.3. Effectiveness analysis of strategy composition

This section mainly verifies the contribution of each strategy to TCSSMH performance. There are three main mechanisms in TCSSMH, which are the multi-population evolution mechanism, elite individual-dominated two-stage coevolution mechanism, and pattern search. To test the influence of three strategies in TCSSMH on the performance of the algorithm, the classical SS, SS with only multi-population mechanism (M−SS), SS with multi-population mechanism, and elite *RefSet* guide strategy (ME-SS), and TCSSMH with all three strategies were tested respectively. The three algorithms are compared with the classical SS on the 10-dimensional CEC2017 benchmark test suite. All the algorithms run 51 times independently. The error value of mean and standard deviation smaller than 10^{-8} will be taken as zero. Table 5 shows the experimental results.

Firstly, the mean and variance of four types of functions are calculated (the optimal results are shown in bold). It is obvious from Table 3 that the mean and variance of TCSSMH are better than M−SS and ME-SS. Through the comparison of each strategy, it shows that each improvement strategy is effective. The line chart is illustrated in Fig. 4. The abscissa represents the test function and the ordinate represents the average of each function(All function values are logarithmically normalized). The purple line at the bottom of the line chart shows that TCSSMH has a stronger exploration and exploitation capacity.

From Fig. 4, for unimodal functions, after adding different improvement strategies, the theoretical optimal value zero is found on F_1 and F_3, each strategy improves the performance of the algorithm. The results show that the integration of multi-group strategy and elite individual guidance mechanism plays a key role in global optimization. For multimodal functions, as shown in Fig. 4(b), it can be seen from F_4 that the search range of the population is expanded after adding the multi-population mechanism, and the search accuracy of the algorithm is improved. The ME-SS makes full use of the optimal information of the subgroup and the optimal information of the whole population to speed up the convergence while maintaining the diversity of the population. For hybrid functions, it is clear from Fig. 4(c) that each strategy plays a key role in improving the performance of the algorithm. In particular, for F_{14}, when the population is in a state of stagnation, pattern search enhances the ability of searching for the optimal solution. For composition functions, the search accuracy of the algorithm is further enhanced with each strategy added.

In order to further clearly describe the influence of the three strategies of M−SS, ME-SS, and TCSSMH on the algorithm in the global iterative process. The three-dimensional topographic maps and two-dimensional plans of F_1, F_4, F_{26} and F_{27} are illustrated in Fig. 5. Both the horizontal and vertical axes represent the value range of the function variable. There are three types of functions. F_1 is unimodal function, F_4 is multimodal function, F_{26} and F_{27} are composition function. The three-dimensional topographic map of each function represents the final optimization result. The red dots in the two-dimensional plan represent the search position of each individual under the condition of different evaluation times. The green point represents the actual optimal solution of the function. In the evolution of the algorithm, the green points are gradually covered by the red ones. Under the condition of four different evaluation times, the M−SS converges relatively slowly without the guidance of excellent individuals. The convergence speed of the ME-SS population is greatly improved, and the global optimal solution is found faster. The pattern search further mines the individuals in the stagnation state, which makes the algorithm jump out of the local optimum and enhances the ability of global optimization.

To sum up, the performance of the TCSSMH is better than other incomplete algorithms. This indicates that all the improvements in the exploration and exploitation capability of the balancing algorithm are effective and indispensable.

5.4. Comparison between TCSSMH and other algorithms

In this section, numerical simulation results are provided to verify the performance of the proposed TCSSMH algorithm. In this paper, the mean value (Mean) and the standard deviation value (Std) are used as the evaluation criteria of TCSSMH and other comparison algorithms. The mean value indicates the optimization ability of the algorithm, and the standard deviation indicates the stability of the algorithm. The experimental results of TCSSMH and the other eleven algorithms are summarized in Tables 6–9 for $10D, 30D, 50D$, and $100D$ cases, respectively. The best performance results of the average and standard values are highlighted in bold font. To avoid the error caused by the randomness of the algorithm, under the same experimental environment, each function runs 51 tests independently in four dimensions: $D = 10, D = 30, D = 50, D = 100$. The average value of these 51 times is used as evaluation data. The final optimal solution of each algorithm and the stability under the specified evaluation times are the key indexes to evaluate the performance of the algorithm. In order to observe the figure more obviously, the experimental results with the theoretical optimal solution of zero are treated logarithmically.

The CEC2017 benchmark test suite was selected to evaluate the performance of the algorithm. In all the following experiments, the

control parameters of each algorithm refer to the recommended values mentioned in the corresponding literature. The specific parameters are given in Table 1. Concerning the TCSSMH, the parameter combination used in this experiment ($b = 50, e = 15, Pbest\_rate = 0.3, \alpha = 1.4, \beta = 0.2$.). It is worth noting that this parameter set may not be the best parameter set to solve all optimization problems. Hence, it is still recommended to use parameter tuning to deal with any problems for better performance.

The original SS, the classic SS variant CSS, the other four state-of-arts SS variants, a metaheuristic algorithm BSA, and an advanced multi-population algorithm (MPEDE) and three PSO variants are selected to compare with the proposed TCSSMH. Here, four representative functions F_1, F_6, F_{20} and F_{27} are selected to explain the experimental results. From Table 4, under the given evaluation times, the optimization performance of TCSSMH with $D = 10$ is better than other comparison algorithms, and the theoretical optimal solution is found stably on functions $F_1, F_2, F_3, F_6, F_9, F_{14}, F_{19}$ and F_{20}. The standard deviation of these functions is zero, which shows that the algorithm has high stability. Although the optimal solution of the MPEDE is also found on F_1, F_2, F_3 and F_6, as shown in Fig. 6, the number of evaluations taken by TCSSMH to calculate the optimal value is much less than that of MPEDE. This is because the strategy guided by elite individuals accelerates the convergence of TCSSMH in the process of population evolution. What's more, under the condition of $D = 10, D = 30, D = 50$, and $D = 100$. The optimal solution is found stable on the functions F_1, F_3, and F_6. This further shows that the algorithm has strong exploitation ability. In conclusion, TCSSMH shows high-quality performance in the comparative test, which verifies the effectiveness of the added strategy.

To make a clear comparison of the experimental results, the specific average convergence curve and box diagram are shown in Figs. 6–13.F_1, F_6, F_{20} and F_{27} are four different types of functions, F_1 is a unimodal function with only one optimal value. In four different dimensions, The proposed TCSSMH search solution has higher accuracy and faster convergence speed. According to the average value in Tables 6–9, the theoretical optimal value of the F_1 function is found. For the $F_1(a)$ convergence graph of each dimension, the final curve tends to be horizontal due to the setting of specific convergence accuracy conditions. If the convergence accuracy is more flexible, the TCSSMH still has the ability to develop and mine new solutions, which shows that the added pattern search strategy has a strong exploitation ability. The box diagram of F_1 in four dimensions is shown in Fig. 7(a), Fig. 9(a), Fig. 11 (a) and Fig. 13(a). From the box diagram, the stability of TCSSMH is better than other comparison algorithms. Therefore, our algorithm is a better choice to solve the problem that there is no local minimum in real life.

The multimodal functions have infinite local extremum information, which is usually more challenging. There is no gradient information in this type of function in the optimization stage, the traditional method can not solve this kind of problem, and it is hard to gain high precision. Consequently, this type of function is used to test the exploration performance of the algorithm. The multi-population mechanism and elite guided search mechanism designed in this paper enhance the possibility of population escaping from local optimization to produce new solutions. For the selected F_6, which represents the multimodal function, the performance of the algorithm will not deteriorate with the increase of dimensions, and the optimal solution is found in different dimensions. Most other algorithms fall into local optimization and converge slowly. Thus, it can be confirmed that TCSSMH has good exploration ability.

In the hybrid function, for function F_{20}, it can be seen from Fig. 6 (a), Fig. 8 (a), Fig. 10(a), and Fig. 12(a), TCSSMH converges faster than other comparison algorithms and can find the optimal solution in $D = 10$. The performance of TCSSMH is slightly worse than that of the MPEDE on 30, 50, and 100 dimensions. However, the convergence speed is faster than any of the seven comparison algorithms, indicating that the TCSSMH has significant differences in convergence performance.

In the composition function, the performance of the TCSSMH algorithm is not outstanding compared with other comparison algorithms, but it can find a better solution. The effectiveness of the improved strategy on SS has been verified again. In summary, the convergence of TCSSMH on four types of functions is better than that of any comparison algorithm. According to the results of Tables 6–9, except for the competitiveness of MPEDE, the proposed TCSSMH is superior to other comparison algorithms in solving the CEC2017 benchmark test suite.

5.5. The result of the Friedman test and the Wilcoxon symbolic rank test

In this paper, the experimental results of eleven comparison algorithms and 30 optimization functions will be used to illustrate the application of nonparametric test statistical methods. The nonparametric test will be used to show significant statistical differences between the different algorithms studied. The Wilcoxon symbolic rank test was selected to analyze whether there were significant differences between the two pairing algorithms. The R^+ is the rank sum that the TCSSMH is better than the current comparison algorithm, and R^- is the rank sum that the current comparison algorithm is better than TCSSMH. The experimental results in this test comparing TCSSMH and other algorithms for all benchmark functions are shown in Table 10. As shown in Table 10, TCSSMH shows a significant improvement over SS, BSA, CSS, SSDE, SSCOL, CeSSOL, SS-MCA, MAPSO,TAPSO and XPSO, with a level of significant $\alpha = 0.05$. The rank-sum of R^+ in 30 and 50 dimensions is higher than that of R^- except for the MPEDE. In all other results, the R^+ of TCSSMH is higher than that of R^-. We utilize the results of the Wilcoxon symbolic rank test to calculate the statistically significant difference between the two algorithms at 5% significance level. The *p-value* can be considered as strong evidence against the null hypothesis. In addition, the 'No' means that there is no significant statistical difference between the values of the two samples, whereas the 'Yes' indicates the opposite. Specifically, if the resulting *p -value* is less than 5%, the algorithm has a statistical advantage in solving the problem. The TCSSMH is compared with other algorithms, all the results have significant statistical differences in $D = 10$. In 30,50,100 dimension problems, except for MPEDE, there are significant differences in other algorithms.

The Friedman test is a nonparametric test to test whether there are significant differences between multiple comparison algorithms by using rank. The statistical results of the Friedman test of 8 algorithms with $D = 10, 30, 50$, and 100 are shown in Fig. 14. The horizontal axis represents the comparison algorithm, the vertical axis represents the average ranking of the results of each algorithm. The average rank of TCSSMH in different dimensions is the smallest of the eight algorithms with $\alpha = 0.05$, and $\alpha = 0.1$. The TCSSMH and MPEDE have no significant difference under the critical difference of confidence interval of 90% and 95%, and the TCSSMH has significant differences with the other eleven comparison algorithms.

6. TCSSMH for engineering problems

In this section, to verify the effectiveness of TCSSMH in dealing real problems, we test TCSSMH on an engineering design problem called tension/compression spring design problem. Note that the optimal solution to be gained has many constraints that should not be violated.

The tension/compression spring design problem, as shown in Fig. 15. The purpose of this engineering problem is to minimize the weight $f(x)$ of the tension/compression spring and is constrained by the minimum deflection, shear stress, surge frequency, outer diameter limit, and design variables. They are four design variables in this problem: wire diameter $d(x1)$, mean coil diameter $D(x2)$, and the number of active coils $P(x3)$.

This problem has been studied by DELC (Li, 2010), DEDS (Zhang, Luo, & Wang, 2008), CPSO (He & Wang, 2007b), HPSO (He & Wang, 2007a), and HEAA (Wang, Cai, Zhou, & Fan, 2009). The details of the mathematical model for this problem and more details about DELC,

DEDS, CPSO, HPSO, and HEAA can be found in (Sadollah, Bahreininejad, Eskandar, & Hamdi, 2012). The problem contains three continuous variables and four inequality constraints, and its mathematical model is expressed as:

$$\textbf{Decision variable}: X = \{d, D, P\} = \{x_1, x_2, x_3\} \tag{18}$$

$$\textbf{Objective}: minf(x) = (x_3 + 2)x_2{x_1}^2 \tag{19}$$

$$\textbf{Subject to}: g_1(x) = 1 - \frac{{x_2}^3 x_3}{71785{x_1}^4} \leq 0 \tag{20}$$

$$g_2(x) = \frac{4{x_2}^2 - x_1x_2}{12566({x_2}^3{x_1}^3 - {x_1}^4)} + \frac{1}{5108{x_1}^2} - 1 \leq 0 \tag{21}$$

$$g_3(x) = 1 - \frac{140.45x_1}{{x_2}^2 x_3} \leq 0 \tag{22}$$

$$g_4(x) = \frac{x_1 + x_2}{1.5} - 1 \leq 0 \tag{23}$$

$$\textbf{Variable range}: 0.05 \leq x_1 \leq 2 \tag{24}$$

$$0.25 \leq x_2 \leq 1.3 \tag{25}$$

$$2 \leq x_3 \leq 15 \tag{26}$$

In this experiment, TCSSMH is compared with the above five comparison algorithms. Each algorithm runs independently 31 times. After performing the same number of fitness value evaluations (50000 Nfes), the average value, the best value, worst value, and variance are calculated. The comparison results and statistical results of the tension/compression spring design problem are shown in Table 11 and Table 12.

As can be seen from Fig. 8, given the same number of evaluations of the fitness value, TCSSMH has superiority in fitness value, and the optimal function value is 0.012641. Note that we are minimizing the objective function, so the minimum fitness value is regarded as the best fitness value. At the same time, according to the statistical analysis in Fig. 9, TCSSMH is also better than the other five comparison algorithms in terms of mean and standard deviation. Thus, TCSSMH is also very competitive in dealing with real-world problems.

7. Conclusions and future work

In this study, a two-stage multi-group hierarchical cooperative learning mechanism for SS is proposed to address problems of slow convergence of the original SS, low solution accuracy, and easy to fall into local optimum when solving high-dimensional problems. Under the guidance of the elite reference set, multiple sub-population operations search different regions at the same time, maintaining the diversity of the population. The information exchange between subpopulations accelerates the convergence speed and achieves a better balance between exploration and development. In addition, pattern search is used to develop the domain space of excellent individuals to effectively avoid falling into local optimization. T is compared with five state-of-the-art SS variants and three PSO variant algorithms on the CEC2017 benchmark test suite. The results of the convergence curve and boxplot indicate that TCSSMH outperforms the compared algorithms in both convergence speed and stability. This study also applies TCSSMH to tensile and compression spring design problems. The experimental results demonstrate that TCSSMH can be successfully applied to practical engineering design problems. Thus, the proposed TCSSMH is an efficient algorithm. In the future, it is an important task to use SS to solve various practical engineering optimization problems and expand the application field of SS. We will combine data-driven mechanisms to build surrogate models to reduce computational complexity and research the application of TCSSMH in solving various complicated optimization problems and large-scale problems.

CRediT authorship contribution statement

Fuqing Zhao: Funding acquisition, Investigation, Supervision. **Gang Zhou:** Investigation, Software, Writing – original draft. **Ling Wang:** Methodology, Resources. **Tianpeng Xu:** Project administration, Writing – review & editing. **Ningning Zhu:** Conceptualization, Formal analysis. **Jonrinaldi:** Visualization.

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

This work was financially supported by the National Natural Science Foundation of China under grant 62063021. It was also supported by the Key talent project of Gansu Province (ZZ2021G50700016), the Key Research Programs of Science and Technology Commission Foundation of Gansu Province (21YF5WA086), Lanzhou Science Bureau project (2018-rc-98), and Project of Gansu Natural Science Foundation (21JR7RA204), respectively.

References

Abd Alradha Alsaidi, S. A., Muhsen, D. K., & Ali, S. M. (2020). Improved Scatter Search Algorithm based on Meerkat Clan Algorithm to Solve NP-Hard Problems. *Periodicals of Engineering and Natural Sciences, 8*(3). https://doi.org/10.21533/pen.v8i3.1563

Angeline, P. J. (1994). Genetic programming: On the programming of computers by means of natural selection,: John R. Koza, A Bradford Book, MIT Press, Cambridge MA, 1992, ISBN 0-262-11170-5, xiv + 819pp., US$55.00. *Biosystems*, *33*(1), 69–73. doi:10.1016/0303-2647(94)90062-0.

Awad, N. H., Ali, M. Z., Liang, J., Qu, B. Y., & Suganthan, P. N. (2016). Problem definitions and evaluation criteria for the CEC 2017 special session and competition on real-parameter optimization. *Nanyang Technol.* Univ Singapore, Tech. Rep (Issue August).

Bajestani, M. A., Rabbani, M., Rahimi-Vahed, A. R., & Baharian Khoshkhou, G. (2009). A multi-objective scatter search for a dynamic cell formation problem. *Computers and Operations Research, 36*(3), 777–794. https://doi.org/10.1016/j.cor.2007.10.026

Çelik, E. (2020). Improved stochastic fractal search algorithm and modified cost function for automatic generation control of interconnected electric power systems. *Engineering Applications of Artificial Intelligence, 88*, Article 103407. https://doi.org/10.1016/J.ENGAPPAI.2019.103407

Chen, M. R., Huang, Y. Y., Zeng, G. Q., Lu, K. Di, & Yang, L. Q. (2021). An improved bat algorithm hybridized with extremal optimization and Boltzmann selection. *Expert Systems with Applications*, *175*(June 2019), 114812. doi:10.1016/j.eswa.2021.114812.

Chen, Y., He, F., Zeng, X., Li, H., & Liang, Y. (2021). The explosion operation of fireworks algorithm boosts the coral reef optimization for multimodal medical image registration. *Engineering Applications of Artificial Intelligence*, *102*(October 2020), 104252. doi:10.1016/j.engappai.2021.104252.

Chen, H., Li, W., & Yang, X. (2020). A whale optimization algorithm with chaos mechanism based on quasi-opposition for global optimization problems. *Expert Systems with Applications, 158*, Article 113612. https://doi.org/10.1016/j.eswa.2020.113612

Chen, C., Wang, P., Dong, H., & Wang, X. (2020). Hierarchical Learning Water Cycle Algorithm. *Applied Soft Computing Journal, 86*, Article 105935. https://doi.org/10.1016/j.asoc.2019.105935

Civicioglu, P. (2013). Backtracking Search Optimization Algorithm for numerical optimization problems. *Applied Mathematics and Computation, 219*(15), 8121–8144. https://doi.org/10.1016/j.amc.2013.02.017

Cuong-Le, T., Minh, H. L., Khatir, S., Wahab, M. A., Tran, M. T., & Mirjalili, S. (2021). A novel version of Cuckoo search algorithm for solving optimization problems. *Expert Systems with Applications, 186*(July), Article 115669. https://doi.org/10.1016/j.eswa.2021.115669

D'Ambrosio, A., Spiller, D., & Curti, F. (2020). Improved magnetic charged system search optimization algorithm with application to satellite formation flying. *Engineering Applications of Artificial Intelligence*, *89*(June 2019), 103473. doi:10.1016/j.engappai.2020.103473.

Das, S., Mullick, S. S., & Suganthan, P. N. (2016). Recent advances in differential evolution-An updated survey. *Swarm and Evolutionary Computation, 27*(February), 1–30. https://doi.org/10.1016/j.swevo.2016.01.004

Del Ser, J., Osaba, E., Molina, D., Yang, X. S., Salcedo-Sanz, S., Camacho, D., … Herrera, F. (2019). Bio-inspired computation: Where we stand and what's next. *Swarm and Evolutionary Computation, 48*(December 2018), 220–250. https://doi.org/10.1016/j.swevo.2019.04.008

Espitia, H. E., & Sofrony, J. I. (2018). Statistical analysis for vortex particle swarm optimization. *Applied Soft Computing Journal, 67*, 370–386. https://doi.org/10.1016/j.asoc.2018.03.002

Gao, S., Zhou, M., Wang, Y., Cheng, J., Yachi, H., & Wang, J. (2019). Dendritic Neuron Model with Effective Learning Algorithms for Classification, Approximation, and Prediction. *IEEE Transactions on Neural Networks and Learning Systems, 30*(2), 601–614. https://doi.org/10.1109/TNNLS.2018.2846646

Glover, F. (1977). Heuristics for integer programming using-surrogate constraints. *Decision Sciences, 8*(1), 156–166. https://doi.org/10.1111/J.1540-5915.1977.TB01074.X

Glover, F. (1998). A template for scatter search and path relinking. *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), 1363*, 3–51. https://doi.org/10.1007/BFB0026589

Hakli, H., & Ortacay, Z. (2019). An improved scatter search algorithm for the uncapacitated facility location problem. *Computers and Industrial Engineering, 135*(May), 855–867. https://doi.org/10.1016/j.cie.2019.06.060

He, Q., & Wang, L. (2007a). A hybrid particle swarm optimization with a feasibility-based rule for constrained optimization. *Applied Mathematics and Computation, 186*(2), 1407–1422. https://doi.org/10.1016/j.amc.2006.07.134

He, Q., & Wang, L. (2007b). An effective co-evolutionary particle swarm optimization for constrained engineering design problems. *Engineering Applications of Artificial Intelligence, 20*(1), 89–99. https://doi.org/10.1016/J.ENGAPPAI.2006.03.003

Herrera, F., Lozano, M., & Molina, D. (2006). Continuous scatter search: An analysis of the integration of some combination methods and improvement strategies. *European Journal of Operational Research, 169*(2), 450–476. https://doi.org/10.1016/J.EJOR.2004.08.009

Holland, J. H. (1992). Genetic algorithms. *Scientific American, 267*(1), 66–72. https://doi.org/10.1038/scientificamerican0792-66

Hvattum, L. M., Duarte, A., Glover, F., & Martí, R. (2013). Designing effective improvement methods for scatter search: An experimental study on global optimization. *Soft Computing, 17*(1), 49–62. https://doi.org/10.1007/s00500-012-0902-9

Kalra, M., Tyagi, S., Kumar, V., Kaur, M., Mashwani, W. K., Shah, H., & Shah, K. (2021). A Comprehensive Review on Scatter Search: Techniques, Applications, and Challenges. *Mathematical Problems in Engineering, 2021*(June). https://doi.org/10.1155/2021/5588486

Kaveh, A., & Dadras, A. (2017). A novel meta-heuristic optimization algorithm: Thermal exchange optimization. *Advances in Engineering Software, 110*, 69–84. https://doi.org/10.1016/j.advengsoft.2017.03.014

Kerkhove, L. P., & Vanhoucke, M. (2017). A parallel multi-objective scatter search for optimising incentive contract design in projects. *European Journal of Operational Research, 261*(3), 1066–1084. https://doi.org/10.1016/j.ejor.2017.02.043

Kumar, M., Kulkarni, A. J., & Satapathy, S. C. (2018). Socio evolution & learning optimization algorithm: A socio-inspired optimization methodology. *Future Generation Computer Systems, 81*, 252–272. https://doi.org/10.1016/j.future.2017.10.052

Laguna, M., & Marti, R. (2003). Scatter search: Methodology and implementations in C. *Operations Research/ Computer Science Interfaces Series, 24*, 1–283. https://doi.org/10.1007/978-1-4615-0337-8

Laguna, M., & Martí, R. (2005). Experimental testing of advanced scatter search designs for global optimization of multimodal functions. *Journal of Global Optimization, 33*(2), 235–255. https://doi.org/10.1007/s10898-004-1936-z

Li, K., & Tian, H. (2015). A DE-based scatter search for global optimization problems. *Discrete Dynamics in Nature and Society, 2015*. https://doi.org/10.1155/2015/303125

Li, Z., & Zhang, Q. (2018). A simple yet efficient evolution strategy for large-scale black-box optimization. *IEEE Transactions on Evolutionary Computation, 22*(5), 637–646. https://doi.org/10.1109/TEVC.2017.2765682

Li, L. W. L. (2010). *An effective differential evolution with level comparison for constrained engineering design*. 947–963. doi:10.1007/s00158-009-0454-5.

Liaw, R. (2021). *A cooperative coevolution framework for evolutionary learning and instance selection*. *62*(February 2020). doi:10.1016/j.swevo.2021.100840.

Liu, F., Huang, H., Yang, Z., Hao, Z., & Wang, J. (2019). Search-based algorithm with scatter search strategy for automated test case generation of NLP toolkit. *IEEE Transactions on Emerging Topics in Computational Intelligence, 5*(3), 491–503. https://doi.org/10.1109/tetci.2019.2914280

Liu, Y., Shi, Y., Chen, H., Heidari, A. A., Gui, W., Wang, M., … Li, C. (2021). Chaos-assisted multi-population salp swarm algorithms: Framework and case studies. *Expert Systems with Applications, 168*, Article 114369. https://doi.org/10.1016/j.eswa.2020.114369

Ma, H., Shen, S., Yu, M., Yang, Z., Fei, M., & Zhou, H. (2019). Multi-population techniques in nature inspired optimization algorithms: A comprehensive survey. *Swarm and Evolutionary Computation, 44*, 365–387. https://doi.org/10.1016/j.swevo.2018.04.011

Ma, Y., Zhang, X., Song, J., & Chen, L. (2021). A modified teaching–learning-based optimization algorithm for solving optimization problem. *Knowledge-Based Systems, 212*, Article 106599. https://doi.org/10.1016/j.knosys.2020.106599

Maheri, A., Jalili, S., Hosseinzadeh, Y., Khani, R., & Miryahyavi, M. (2021). A comprehensive survey on cultural algorithms. *Swarm and Evolutionary Computation, 62*(November 2020), 100846. doi:10.1016/j.swevo.2021.100846.

Nebro, A. J., Luna, F., Alba, E., Dorronsoro, B., Durillo, J. J., & Beham, A. (2008). AbYSS: Adapting scatter search to multiobjective optimization. *IEEE Transactions on Evolutionary Computation, 12*(4), 439–457. https://doi.org/10.1109/TEVC.2007.913109

Paithankar, A., & Chatterjee, S. (2019). Open pit mine production schedule optimization using a hybrid of maximum-flow and genetic algorithms. *Applied Soft Computing Journal, 81*. https://doi.org/10.1016/J.ASOC.2019.105507

Pardo, X. C., Argüeso-Alejandro, P., González, P., Banga, J. R., & Doallo, R. (2020). Spark implementation of the enhanced Scatter Search metaheuristic: Methodology and assessment. *Swarm and Evolutionary Computation, 59*. https://doi.org/10.1016/j.swevo.2020.100748

Pashaei, E., & Aydin, N. (2017). Binary black hole algorithm for feature selection and classification on biological data. *Applied Soft Computing, 56*, 94–106. https://doi.org/10.1016/j.asoc.2017.03.002

Remli, M. A., Deris, S., Mohamad, M. S., Omatu, S., & Corchado, J. M. (2017). An enhanced scatter search with combined opposition-based learning for parameter estimation in large-scale kinetic models of biochemical systems. *Engineering Applications of Artificial Intelligence, 62*(February), 164–180. https://doi.org/10.1016/j.engappai.2017.04.004

Remli, M. A., Mohamad, M. S., Deris, S., A Samah, A., Omatu, S., & Corchado, J. M. (2019). Cooperative enhanced scatter search with opposition-based learning schemes for parameter estimation in high dimensional kinetic models of biological systems. *Expert Systems with Applications*, *116*, 131–146. doi:10.1016/j.eswa.2018.09.020..

Sadiq, A. T. (2013). Improved Scatter Search Using Cuckoo Search. *International Journal of Advanced Research in Artificial Intelligence, 2*(2), 61–67. https://doi.org/10.14569/ijarai.2013.020210

Sadollah, A., Bahreininejad, A., Eskandar, H., & Hamdi, M. (2012). Mine blast algorithm: A new population based algorithm for solving constrained engineering optimization problems. *Applied Soft Computing Journal*. https://doi.org/10.1016/j.asoc.2012.11.026

Sagheer, A. M., Sadiq, A. T., & Ibrahim, M. S. (2012). Improvement of scatter search using Bees Algorithm. In *6th International Conference on Signal Processing and Communication Systems*. https://doi.org/10.1109/ICSPCS.2012.6507943

Sahu, R. K., Panda, S., & Padhan, S. (2015). A novel hybrid gravitational search and pattern search algorithm for load frequency control of nonlinear power system. *Applied Soft Computing Journal, 29*, 310–327. https://doi.org/10.1016/j.asoc.2015.01.020

Sarakhsi, M. K., Fatemi Ghomi, S. M. T., & Karimi, B. (2016). A new hybrid algorithm of scatter search and Nelder-Mead algorithms to optimize joint economic lot sizing problem. *Journal of Computational and Applied Mathematics, 292*, 387–401. https://doi.org/10.1016/j.cam.2015.07.027

Singh, G., & Singh, A. (2021). Extension of particle swarm optimization algorithm for solving transportation problem in fuzzy environment. *Applied Soft Computing, 110*, Article 107619. https://doi.org/10.1016/J.ASOC.2021.107619

Soman, J. T., & Patil, R. J. (2020). A scatter search method for heterogeneous fleet vehicle routing problem with release dates under lateness dependent tardiness costs. *Expert Systems with Applications, 150*, Article 113302. https://doi.org/10.1016/j.eswa.2020.113302

Sreelaja, N. K. (2021). Ant Colony Optimization based Light weight Binary Search for efficient signature matching to filter Ransomware. *Applied Soft Computing, 111*, Article 107635. https://doi.org/10.1016/j.asoc.2021.107635

Sun, Y., & Chen, Y. (2021). Multi-population improved whale optimization algorithm for high dimensional optimization. *Applied Soft Computing, 112*, Article 107854. https://doi.org/10.1016/j.asoc.2021.107854

Talaei, K., Rahati, A., & Idoumghar, L. (2020). A novel harmony search algorithm and its application to data clustering. *Applied Soft Computing Journal, 92*, Article 106273. https://doi.org/10.1016/j.asoc.2020.106273

Wang, Y., Cai, Z., Zhou, Y., & Fan, Z. (2009). Constrained optimization based on hybrid evolutionary algorithm and adaptive constraint-handling technique. *Structural and Multidisciplinary Optimization, 37*(4), 395–413. https://doi.org/10.1007/s00158-008-0238-3

Wang, Yirui, Gao, S., Member, S., Zhou, M., & Yu, Y. (2021). *A Multi-Layered Gravitational Search Algorithm for Function Optimization and Real-World Problems*. *8*(1), 94–109.

Wang, Y., Gao, S., Yu, Y., Cai, Z., & Wang, Z. (2021a). A gravitational search algorithm with hierarchy and distributed framework. *Knowledge-Based Systems, 218*, Article 106877. https://doi.org/10.1016/j.knosys.2021.106877

Wang, Y., Gao, S., Yu, Y., Cai, Z., & Wang, Z. (2021b). Knowledge-Based Systems A gravitational search algorithm with hierarchy and distributed framework. *Knowledge-Based Systems, 218*, Article 106877. https://doi.org/10.1016/j.knosys.2021.106877

Wei, B., Xia, X., Yu, F., Zhang, Y., Xu, X., Wu, H., … He, G. (2020). Multiple adaptive strategies based particle swarm optimization algorithm. *Swarm and Evolutionary Computation, 57*, Article 100731. https://doi.org/10.1016/j.swevo.2020.100731

Wu, G., Mallipeddi, R., Suganthan, P. N., Wang, R., & Chen, H. (2016). Differential evolution with multi-population based ensemble of mutation strategies. *Information Sciences, 329*(September), 329–345. https://doi.org/10.1016/j.ins.2015.09.009

Xia, X., Gui, L., He, G., Wei, B., Zhang, Y., Yu, F., … Zhan, Z. H. (2020). An expanded particle swarm optimization based on multi-exemplar and forgetting ability. *Information Sciences, 508*, 105–120. https://doi.org/10.1016/j.ins.2019.08.065

Xia, X., Gui, L., Yu, F., Wu, H., Wei, B., Zhang, Y. L., & Zhan, Z. H. (2020). Triple archives particle swarm optimization. *IEEE Transactions on Cybernetics, 50*(12), 4862–4875. https://doi.org/10.1109/TCYB.2019.2943928

Xue, Y., Jiang, J., Zhao, B., & Ma, T. (2018). A self-adaptive artificial bee colony algorithm based on global best for global optimization. *Soft Computing, 22*(9), 2935–2952. https://doi.org/10.1007/s00500-017-2547-1

Yang, Q., Member, S., Chen, W., & Deng, J. Da. (2017). *A Level - based Learning Swarm Optimizer for Large Scale Optimization*. *c*. doi:10.1109/TEVC.2017.2743016.

Yang, P., Zhang, H., Yu, Y., Li, M., & Tang, K. (2022). Evolutionary reinforcement learning via cooperative coevolutionary negatively correlated search. *Swarm and Evolutionary Computation*, *68*(October 2020), 100974. doi:10.1016/j.swevo.2021.100974.

Yao, X., Liu, Y., & Lin, G. (1999). Evolutionary programming made faster. *IEEE Transactions on Evolutionary Computation, 3*(2), 82–102. https://doi.org/10.1109/4235.771163

Yu, X., Xu, W. Y., & Li, C. L. (2021). Opposition-based learning grey wolf optimizer for global optimization. *Knowledge-Based Systems, 226*, Article 107139. https://doi.org/10.1016/j.knosys.2021.107139

Zhang, M., Luo, W., & Wang, X. (2008). Differential evolution with dynamic stochastic selection for constrained optimization. *Information Sciences, 178*(15), 3043–3074. https://doi.org/10.1016/J.INS.2008.02.014

Zhao, F., Ding, R., Wang, L., Cao, J., & Tang, J. (2021). A hierarchical guidance strategy assisted fruit fly optimization algorithm with cooperative learning mechanism. *Expert Systems with Applications, 183*(June), Article 115342. https://doi.org/10.1016/j.eswa.2021.115342

Zhao, F., Zhang, L., Zhang, Y., Ma, W., Zhang, C., & Song, H. (2020). A hybrid discrete water wave optimization algorithm for the no-idle flowshop scheduling problem with total tardiness criterion. *Expert Systems with Applications, 146*, Article 113166. https://doi.org/10.1016/J.ESWA.2019.113166

Zhao, F., Zhao, L., Wang, L., & Song, H. (2020). An ensemble discrete differential evolution for the distributed blocking flowshop scheduling with minimizing makespan criterion. *Expert Systems with Applications, 160*, Article 113678. https://doi.org/10.1016/J.ESWA.2020.113678