Success intensity-based chaotic local search in metaheuristic algorithms

by

Lin Yang

A dissertation

submitted to the Graduate School of Science and Engineering

in Partial Fulfillment of the Requirements

for the Degree of

Doctor of Engineering

University of Toyama
Gofuku 3190, Toyama-shi, Toyama 930-8555 Japan

2021

(Submitted December 17, 2021)

Abstract

Optimization refers to the problem of finding the best solution under constraints. It is widely used in industrial production, aerospace technology, biomedicine and many other fields. Since traditional methods are difficult to find the best solution when faced with complex optimization problems with high dimensions, researchers have proposed methods to find approximate solutions. The advantage of this method is that it can find an acceptable solution under the constraints and save resources. Metaheuristic algorithms, as a class of approximation algorithms, have been widely studied due to their simplicity, generality, and robustness.

The spherical evolutionary algorithm and the equilibrium optimizer are two new metaheuristic algorithms proposed in recent years. The spherical evolutionary algorithm generalises the hypercube search style of most algorithms and innovatively proposes a spherical search style. The spherical search style provides a larger search space and increases the probability of finding the optimal solution. The equilibrium optimizer is inspired by physical laws. It introduces the concept of an equilibrium pool, using the four best individuals of each generation to improve the ability to jump out of the local optimal. These two algorithms performed well on the benchmark test sets, but they are still considered to have room for improvement. The ability of an algorithm to find the global optimal solution depends on the balance of exploration and exploitation. The aim of exploration is to find the region where the optimal solution exists; the aim of exploitation is to find the optimal solution in the current region. Therefore, our focus is on providing algorithms with a strategy that balances exploration and exploitation.

This paper innovatively proposes an adaptive chaotic local search strategy based on differential radius to improve the performance of the spherical evolutionary algorithm and the equilibrium optimizer. We propose a concept called success intensity, which records the magnitude of the improvement brought by each chaotic sequence. The success intensity is used to guide the algorithm in the selection of chaotic sequences, thus better balancing exploration and exploitation. The improved algorithms, namely the chaotic spherical evolution algorithm and the chaotic equilibrium optimizer, showed better performance on the benchmark test sets.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 An outline of metaheuristic algorithm

Optimization can be seen everywhere, whether it's planning your route to work, the order of chores or engineering, biomedicine. The purpose of optimization is to choose the most profitable solution. Optimization is usually accompanied by some constraints. It is also necessary to consider the fare or the time of arrival. The traditional methods for solving such problems include enumeration, gradient descent, and so on. However, the traditional methods have their own limitations, some of them require the problem to satisfy linearity conditions, and some of them can only solve small-scale problems. When faced with complex engineering problems with high dimensions, it is often difficult to find the optimal solution in a tractable time.

To solve this problem, in the 1940s, researchers proposed heuristic algorithms that sacrifice accuracy to reduce computational complexity. It is an intuitively or empirically constructed algorithm that gives a feasible solution to the optimization problem at an acceptable cost (computational time, space occupied, etc.). It is not guaranteed to find the global optimal solution, but it can give a superior solution in an acceptable time and the algorithm is simple and easy to modify. The greedy algorithm is one of the classical heuristics, and its solution logic is to take the current optimal choice each time. This approach is simple and efficient, but when the problem becomes complex, it will fall easily into a local optimal.

In the 1960s, metaheuristic algorithms inspired by random phenomena in nature emerged under the influence of bionics. Metaheuristic algorithms are improvements of heuristic algorithms, but they are also not guaranteed to find the global optimal solution. The main difference between the two in problem solving is that the solution found by heuristic algorithms are fixed, but metaheuristic algorithms find different solutions each time, which gives it the ability to jump out of the local optimal. In addition heuristic algorithms are suitable for solving specific problems, metaheuristic algorithms are not limited to specific problems and are more widely used. After decades of development, four types of metaheuristic algorithms have emerged: (1) evolutionary algorithms, (2) population intelligence, (3) physics-based, and (4) human-based. Genetic algorithm [1] is one of the representatives of evolutionary algorithm. It was first proposed by Holland in 1975 and was inspired by Darwin's theory of evolution and Mendel's genetics. The particle swarm optimization [2] is one of the representatives of population intelligence, which finds the optimal solution by simulating the foraging behavior of a flock of birds. The gravitational search algorithm [3] is a member of the physics-based class, which uses the gravitational laws between masses and interactions to update the location of points to find the optimal solution. Brain storm optimization [4] is a member of human-based algorithms, which imitates human brainstorming behavior to continuously generate new ideas to reach the optimal.

1.2 Typical metaheuristic algorithms

1.2.1 Genetic algorithm

The biological populations on the earth have been evolving since they were born. They have gone through a process from low to high, from simple to complex, and from defect to perfection. This process follows the rules of ”natural selection” and ”survival of the fittest”. On the other hand, in the reproduction process of the population, the offspring will inherit the genes of the parent and will generally be more adaptable to the environment. In addition, the evolution of populations is accompanied by uncertain genetic mutations, which bring unpredictable changes to the population and can make individuals stronger or weaker. The mutated population goes through the above process again, and in such a spiraling cycle, the population is continuously enhanced. A simple depiction of this process is given in Fig. 1.1.

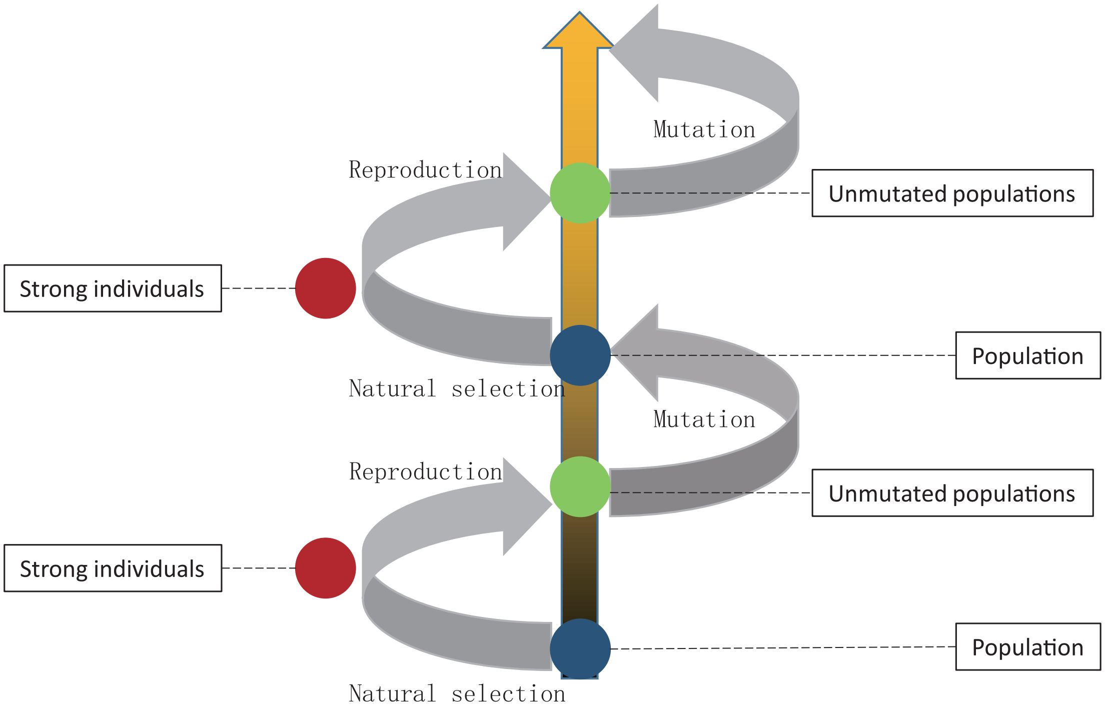


Figure 1.1: Evolutionary spiral.

The genetic algorithm (GA) takes inspiration from the above process. It treats each valid solution as a chromosome and each coding unit of the chromosome as a gene. The fitness value is used to describe the quality of the population and the function to calculate the fitness value is called the evaluation function. Natural selection, reproduction and mutation are then expressed as the following three operators:

(1) Selection operator: A number of chromosomes are selected from the population as the parent population according to certain rules. Selection is usually done using roulette, where the better chromosomes have a higher probability of being selected.

(2) Crossover operator: The probability of crossover is pre-set and a random value is assigned to each chromosome; if the random value is less than the crossover probability, the chromosome will participate in the crossover. The way of crossover is to exchange part of the genes in the chromosomes in pairs to generate the offspring.

The offspring will replace the parent in the new population, while the chromosomes not involved in the crossover will enter the new population directly.

(3) Mutation operator: A small probability of mutation is pre-set and a random value is assigned to each gene of each chromosome. If this value is less than the mutation probability the gene will be mutated.

The population that has not undergone natural selection is called the initial population, which is the first step of GA. Normally, each gene of each chromosome is a random value. The range of the random value depends on the problem. If the initial population has a good fitness, the algorithm's ability to find the global optimum will also be improved. It has been found that the performance of GA can be improved by certain strategies to increase the initial population while ensuring the completeness of the search space. The termination of the algorithm means exhaustion of computing resources. Usually, the maximum number of iterations or function evaluations is set as the termination condition. Fig. 1.2 shows the flowchart of GA.

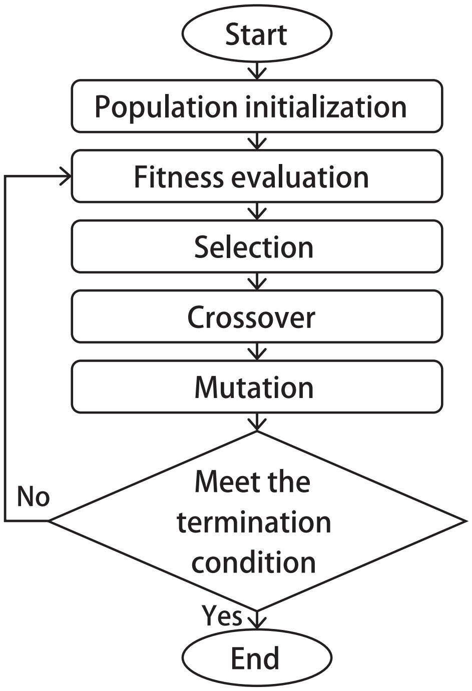


Figure 1.2: Flowchart of GA.

1.2.2 Particle swarm optimization

Researchers have found that flocks of animals in nature, such as bird flocks and fish flocks, exhibit a high degree of organization and regularity in their foraging and migration processes. There are underlying patterns in the aggregation or direction-change behavior of animal flocks. It has been suggested that, at least in theory, in group foraging behavior, each individual benefits from the discoveries and experiences of the other members. Suppose there is a flock of birds in the sky looking for food, and they do not know the location of the food in advance. But by chance, some birds know the distance between the food and themselves, such as the intensity of the smell of the food. The birds record the location of the strongest smell of food they pass, and share this information with other birds. After comparison, the best foraging location in the group can be known. Thus, all birds adjust their direction towards the food location based on their experience and speed, which eventually makes the flock gather to the food location.

The key to guiding birds to forage can be summarized in two points, their own experience and the experience of other birds. The particle swarm optimization (PSO) is designed with this in mind. First, the particles in the PSO represent the birds, and a number of particles are randomly generated as initial solutions. Through iteration, the quality of the solution is improved to achieve optimization. The particles have two properties, velocity and position. During the iteration, the best position the particle experienced will be recorded. The best position of the flock will also recorded. The particle will update to the new position according to its position and speed, affected by the above two.

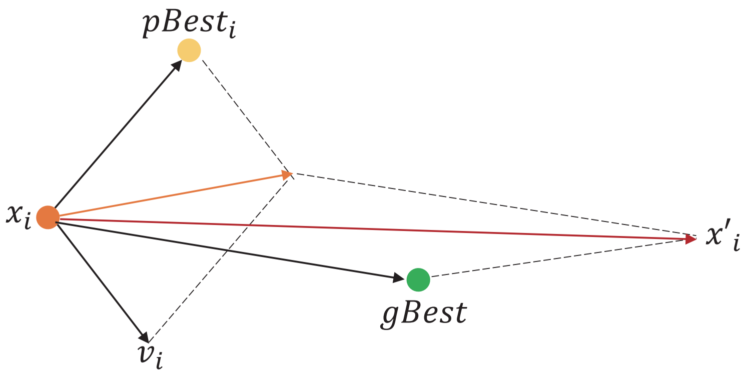


Figure 1.3: Position update in PSO.

This position update process is shown as Fig. 1.3. x represents the position of the flock, x_i denotes the position of the ith bird, the best position the bird experienced is denoted as $pBest_i$, and its own velocity is recorded as v_i. The best position of the flock is recorded as $gBest$. x_i' denotes the updated position. The iteration process of PSO is demonstrated as follows:

step 1) Initialize all individuals and each particle gets a random velocity and position. Record the historical best position of each particle. Record the best position of the particle swarm.

step 2) The fitness value of each particle is calculated by the evaluation function. The evaluation function depends on the problem.

step 3) If the current fitness value of the particle is better than the historical best value, the current position will replace the historical best position.

step 4) If the current particle swarm best fitness is better than the historical swarm best fitness, this particle position will replace the historical swarm best position.

step 5) Update the velocity and position of the particles by Eqs. (1.1,1.2),where N is the population size, k and K denotes the current number of iterations and the maximum number of iterations..

step 6) Determine whether the termination condition is reached, if not, go to step 2, otherwise output the best position of particle swarm and end.

$$v_i(k+1) = \omega \times v_i(k) + c_1 \times rand_1 \times (pBest_i(k) - x_i(k)) + c_2 \times rand_2 \times (gBest - x_i(k)) \\ i = 1, 2, 3, ..., N \quad k = 1, 2, 3, ..., K \tag{1.1}$$

$$x_i(k+1) = x_i(k) + v_i(k) \quad i = 1, 2, 3, ..., N \quad k = 1, 2, 3, ..., K \tag{1.2}$$

1.2.3 Gravitational search algorithm

Newton's law of universal gravitation states that the force of attraction between two particle is proportional to their mass product and inversely proportional to the square of the distance between them. It can be expressed as Eq. (1.3)

$$F = g\frac{m_1 m_2}{r^2} \tag{1.3}$$

where F denotes the gravitational force between two particles, G is the gravitational constant, m_1 and m_2 denote the masses of the two particles, and R represents the distance between them. In 2009, E. Rashedi. et al. simulated this physical phenomenon and proposed a gravitational search algorithm (GSA) [3].

In GSA, the position of each particle corresponds to a solution of the problem. The inertial mass of the particle is related to the quality of the solution, and the quality of the solution is calculated by the evaluation function. The better the quality of the solution, the greater the mass of the particle. When subjected to gravitational forces, particles with larger masses obtain smaller accelerations and smaller movements, which ensure the exploitation capability; particles with smaller masses obtain larger accelerations and larger movements, which ensure the exploration capability. Fig. 1.4 briefly describes the force of particles in GSA. In the figure, m_1 to m_4 represent four particles with different masses, and a force analysis is performed for m_1. The gravitational force on m_1 depends on the masses of m_2, m_3, and m_4 and their distance from m_1. $F_{1,2}$ represents the gravitational force on m_1 from m_2, and $F_{1,3}$ and $F_{1,4}$ are similar. The red arrow F indicates the final force on m_1.

The inertial mass of the particle is related to the fitness of the current population, and it is calculated as shown in Eq. (1.4 and 1.5). f_i denotes the fitness of the ith particle in the population, and f_w and f_b represent the worst and best fitness in the current population, respectively. m_i is the mass of the ith particle.

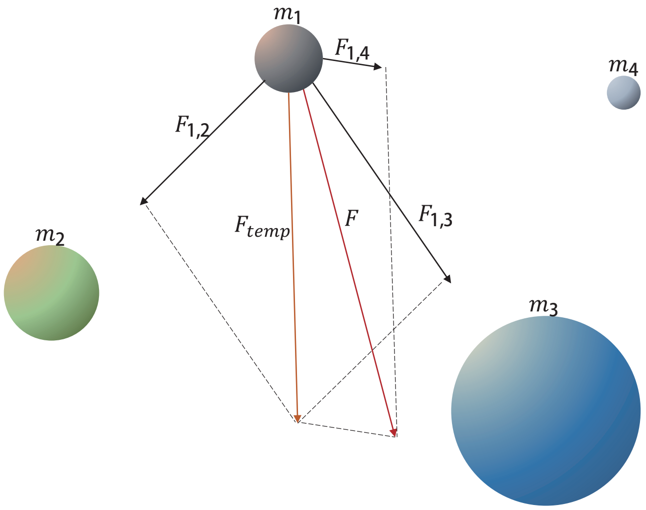


Figure 1.4: Force analysis in GSA.

$$s_i = \frac{f_i - f_w}{f_b - f_w} \tag{1.4}$$

$$m_i = \frac{s_i}{\sum_{j=1}^{N} s_j} \tag{1.5}$$

Each particle is subject to the gravitational force from other particles, and the gravitational force on particle x_i from x_j is calculated as Eq. (1.6).

$$F_{i,j} = g\frac{m_i \times m_j}{r_{i,j} - \theta}(x_j - x_i) \quad i = 1, 2, 3, ..., N \quad j = 1, 2, 3, ..., N \tag{1.6}$$

where $r_{i,j}$ denotes the Euclidean distance between x_i and x_j, θ is a small constant. α represents a constant parameter that affects the decreasing trend of the gravitational coefficient. g is a gravitational constant that decreases with time, defined as Eq. (1.7).

$$g(k) = g_0 \times e^{-\alpha k/K} \quad k = 1, 2, 3, ..., K \tag{1.7}$$

where g_0 means the initial value of g, k denotes the current number of iterations, and K denotes the maximum number of iterations. Thus the total gravitational force on the particle x_i is expressed as:

$$F_i = \sum_{j=1, j \neq i}^{N} rand_j F_{i,j} \tag{1.8}$$

where $rand_j$ is a random value in the interval (0,1). After that GSA will calculate the acceleration β for each particle, β is proportional to F_i and inversely proportional to m_i, defined as:

$$\beta_i = \frac{F_i}{m_i} \tag{1.9}$$

Finally, the velocity v_i and position update of particle x_i is calculated as:

$$v_i(k+1) = rand_i \times v_i(k) + \beta_i(k) \tag{1.10}$$

$$x_i(k+1) = x_i(k) + v_i(k+1) \tag{1.11}$$

A complete GSA process should include the following steps:

step 1) Initialize the particle position and set the initial velocity to 0.

step 2) Evaluate the fitness of each particle and pick out the best and worst fitness in the current iteration.

step 3) Calculate the inertial mass of the particle and update the gravitational constant.

step 4) Calculate the gravitational force between particles.

step 5) Calculate the acceleration and velocity of the particles.

step 6) Update the position of the particle.

step 7) Determine whether the termination condition is reached, if not go to step 2, if yes then output the result.

1.2.4 Brain storm optimization

In group decision-making, members' thinking is often affected by the authority of others or the opinions of the majority. This situation weakens the creativity of group and impairs the quality of decision-making. In order to break this situation, the founder of an advertising company proposed a brainstorming method. Brainstorming allows decision-making members to speak freely without being restricted by the atmosphere, thereby improving the quality of decision-making. The success of brainstorming comes from the following four basic rules:

1 Make as many suggestions as possible. The more suggestions there are, the greater the chance of excellent suggestions.

2 No criticism. Any suggestion in this session is not allowed to be criticized. Members' minds are thus unrestrained to come up with unusual ideas.

3 Encourage wild ideas. These ideas usually come from a different perspective and can bring unexpected rewards.

4 Combine thinking and improve ideas. New ideas can come from existing ideas, and the combination of thinking can come up with better ideas.

The core thinking of the above rules can be summarized in two points: 1. Wild new ideas can offer more possibilities and provide a new way out when the discussion gets stuck. 2. Digging or combining existing ideas can lead to better ideas. Brain storm optimization (BSO) is inspired by these two points. In BSO, each individual corresponds to a suggestion or idea in the brainstorming. Similar individuals are clustered. The best individual in each cluster is called the cluster center. Individual updates mimic the brainstorming process and are generated either randomly, by a single individual, or by a combination of two individuals. Fig. 1.5 and Fig. 1.6 describe these three types of updates. The light bulbs represent the ideas, which are individuals, and the red bulbs represent the cluster centers. In Fig. 1.5, a randomly generated individual replaces a randomly selected cluster center. In Fig. 1.6, there are two ways to update individuals. If only one cluster is selected, one of the common individuals or the cluster center will be selected for updating. If two clusters are selected, common individuals or cluster centers located in each cluster will be combined for update.

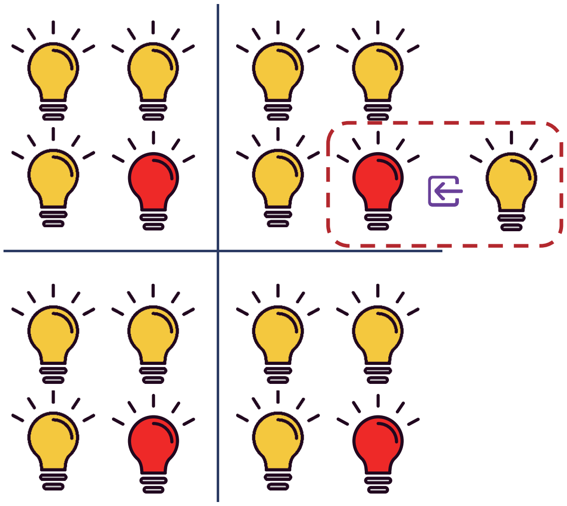


Figure 1.5: Schematic diagram of random individual update in BSO.

Based on the above key information, the complete process of BSO is described as follows:

step 1) Randomly generate *N* individuals.

step 2) Cluster individuals using a clustering algorithm.

step 3) Evaluate the fitness of each individual.

step 4) Record the best individual in each cluster, which is also called the cluster center.

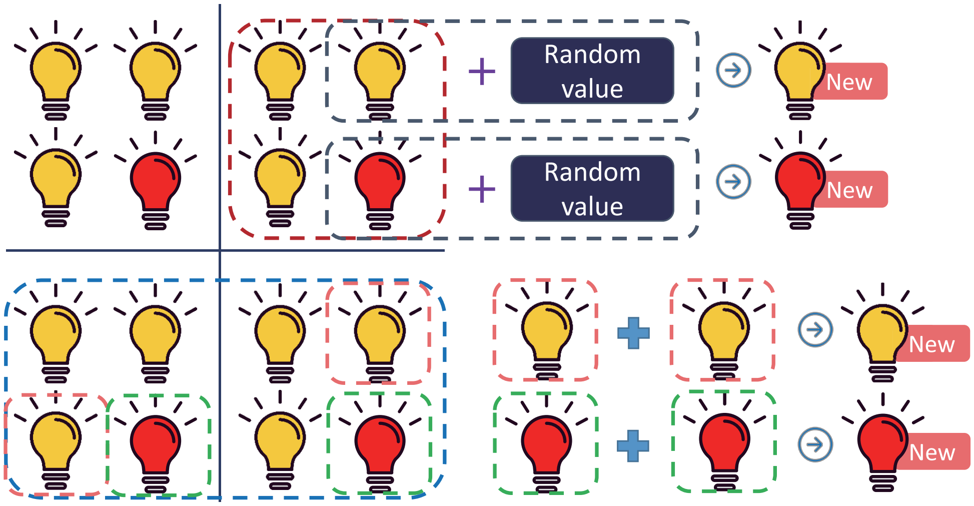


Figure 1.6: Schematic diagram of single or double individual update in BSO.

step 5) Decide whether to execute it or not with probability p_r. If yes, randomly generate an individual to replace a randomly selected cluster center.

step 6) Generate a random value between 0 and 1. If it is less than probability p_1, execute step 7, otherwise execute step 11.

step 7) Pick one cluster out at random.

step 8) Generate a random value between 0 and 1. If it is less than probability p_{1i}, execute step 9, otherwise execute step 10.

step 9) Use the cluster center in step 7 to generate a new individual. Go to step 15.

step 10) Use one common individual in step 7 to generate a new individual. Go to step 15.

step 11) Pick two cluster out at random.

step 12) Generate a random value between 0 and 1. If it is less than probability p_{2i}, execute step 13, otherwise execute step 14.

step 13) Combine the two cluster centers in step 11 and generate a new individual. Go to step 15.

step 14) Combine two common individuals in each cluster in step 11 and generate a new individual. Go to step 15.

step 15) Determine whether the new individual is better than the original individual, if so, replace the original individual, if not, discard.

step 16) Determine if N new individuals have been generated, if so, go to step 17, if not, go to step 6.

step 17) If the termination condition is reached, it ends, if not, go to step 2.

The individuals selected in the above step are updated using Ep. 1.12. x^s represents the selected individual, and x^n denotes the generated individual. $E(\delta, \eta)$ is a random value generated by the Gaussian distribution, and δ and η are the mean and variance of this Gaussian distribution.

$$x^n = x^s + \gamma \times E(\delta, \eta) \tag{1.12}$$

where E is a parameter used to adjust the Gaussian random value, which is related to the number of iterations, defined as:

$$E = logsig(\frac{0.5 \times K - k}{\varphi}) \times rand_E \quad k = 1, 2, 3, ..., K \tag{1.13}$$

where k and K denote the number of current iterations and the maximum number of iterations. $logsig()$ is a sigmoid function, and φ is used to adjust its slope. $rand_E$ means a random value in the interval (0, 1).

Chapter 2

Adaptive chaotic spherical evolution algorithm

2.1 Introduction

Nature-inspired metaheuristic algorithms (NMH) have garnered extensive attention in recent decades because of their excellent ability to solve complex problems [5, 6]. Inspired by nature, these algorithms can achieve a balance between the time required to arrive at a solution and the accuracy of the solution derived when faced with complex problems. For example, the genetic algorithm (GA) [7, 8], ant colony optimization (ACO) [9, 10], and brainstorming optimization (BSO) [11–13] are inspired by evolution and genetics, the foraging behavior of ants, and the brainstorming behavior of humans, respectively. Currently, NMHs are being used in several fields, such as the assignment problem, flow shop problem, traveling salesman problem, group-shop scheduling problem, and voltage reactive power control [14–17].

Although NMHs can achieve a balance between solution time and solution accuracy, the pursuit of higher accuracy is still a significant task for researchers. The quality of the solution depends on the balance of exploitation and exploration in the algorithm. Excessive exploration will cause the algorithm to converge slowly; on the contrary, it may cause the algorithm to fall into a local optimum and converge prematurely [18]. The improvement of algorithm performance can be implemented in the exploration and exploitation of the search space. Under this view, an algorithm with excellent exploration capabilities, i.e., spherical evolution algorithm (SE) [19], has received much attention.

The SE algorithm was proposed in 2019. It summarizes most search patterns of NMHs and proposes a spherical search style based on the mathematical form instead of the hypercube approach. As spherical search has a larger search range than hypercube search in most cases, it can explore better. This makes the SE stand out among several heuristic algorithms. Nevertheless, there is still room for improvement; for example, the full utilization of the larger search space of SE may improve the speed of convergence.

This idea leads the focus of the study to the memetic method. Memes refers to local search strategies, and their combination with evolutionary algorithms is known as memetic algorithms (MA) [20]. Certain studies have reported that adding local search operators to the global search algorithm is an effective method to improve its search performance [21–23]. One of them, chaotic local search (CLS), has been shown to be effective in improving the convergence speed and the quality of solutions [24, 25].

CLS is one of the methods to improve the performance of algorithms using chaos. Chaotic systems [26, 27], due to their ergodicity and randomicity, are incorporated in many optimizers to improve the search performance. Chaotic systems are commonly applied in two ways.

A common way is to replace random values with chaotic maps and adjust parameters. In [28], a chaotic number generator is introduced to generate all the random values needed in the particle swarm optimization algorithm (PSO), which makes this algorithm competitive with other improved PSO. In [29], the PSO is improved because multiple chaotic maps are used to adjust the important attraction parameters. In addition to the PSO, the properties of chaos are also applied to the differential evolution algorithm (DE) [30] to improve their performance [31]. Moreover, chaos is also seen to play a role in some newly proposed algorithms, such as firefly algorithm with chaos [32], chaotic bat algorithm [33], etc [34, 35].

Using chaotic search as a local search operator is another effective way to improve the convergence speed and the quality of the solution, i.e., CLS [24]. CLS [36] has garnered attention because of its ability to accelerate the convergence of the GSA. The ergodicity of chaos makes it more capable of finding the optimum solution in a nearby area [24].

Moreover, the unpredictability caused by its randomicity also accords the algorithm the ability to jump out of a local optimum. In the chaotic gravitational search algorithm (CGSA) [37], two chaotic participation schemes are proposed—one replaces different parameters of the GSA with random numbers using chaotic sequences and the other adds chaotic search as a local search to the GSA. In the chaotic brain swarm optimization algorithm (CBSO) [12], when the population is stagnant, CLS is used to break the existing equilibrium. In CJADE [38], which is an improved variant of DE [39–42], several chaotic maps are used, and the algorithm selects random values generated by the different chaotic maps at various stages. It is noteworthy that a chaotic bee colony algorithm research [43] shows that the artificial bee colony algorithm using CLS performs better than using chaotic variables. The reason lies on that the introduction of CLS adds a new operator to the algorithm, which changes its search dynamics and interacts more directly with the search space. This search strategy is more effective than just replacing random numbers.

Based on the above considerations that CLS can further improve the performance of an NMH, this paper for the first time proposes an adaptive chaotic spherical evolution algorithm (CSE) based on the SE. The CSE uses CLS along with several chaotic maps; CLS employs one chaotic map at a time and acts on the best individual within the population. When CLS finds a better solution, the success information is recorded. The historical success information guides the algorithm to select an appropriate chaotic map in the next iteration. This approach combines well with the global search mechanism of the SE, which helps increase the global convergence speed.

The novelty of our proposed study is that we focus more on the extent of improvement brought about by CLS in the iterations; we refer to it as success intensity. Since several chaotic maps are employed in CLS, a key challenge that arises is the choice of which map should be used and when it should be used. Previous usage of CLS [12,36–38] only records the success rate of each map, and the ones with a higher success rate are more likely to be selected. We believe that the extent of improvement is sometimes more significant than the success rate. Therefore, the CSE records the intensity of success brought about by each map in several iterations. The higher the intensity, the higher the probability of being selected in the next generation.

The remainder of this chapter is structured as follows: In Section 2.2, we describe the original SE and its contributions. In Section 2.3, we illustrate CLS based on the historical recorded success intensity and the CSE process. In Section 2.4, we present the results of experiments and compare those obtained using CSE with those obtained using SE and other algorithms. Finally, in Section 2.5, we present a summary and future research directions.

2.2 Spherical evolution algorithm

The search operator is the core of the algorithm, which determines the ability of each individual in the population to find a better solution. The search operator of the nature-inspired metaheuristic (NMH) algorithm has certain common patterns. For example, the initial solution is updated through one or more update units. The SE employs the concept of search pattern and search style based on a mathematical form. A search pattern can be expressed as Eq. (2.1):

$$\begin{aligned} A_{i,d}^{new} = B_{i,d}^{0} + \sum_{k=1}^{n} SS\left(M_{i,d}^{k}, N_{i,d}^{k}\right), i = 1, 2, 3, \ldots popsize, \\ d = 1, 2, 3, \ldots, D \end{aligned} \tag{2.1}$$

Here, A, B, M, and N are four solution sets, which are represented as matrices consisting of *popsize* rows and D columns. Each individual in the population corresponds to a solution, and the size of population represents the number of individuals. The dimension of the solution is represented by D. $SS\left(M_{i,d}^{k}, N_{i,d}^{k}\right)$ represents an updating unit based on M and N that acts on the initial individual B^{0} to generate a new individual A^{new}, and n represents the number of updating units.

The first-order difference is a common updating unit in NMH algorithms. In high-dimensional space, this unit will appear as a search process within the scope of the hypercube. The updating unit is expressed as Eq. (2.2):

$$SS\left(M_{i,d}^{k}, N_{i,d}^{k}\right) = ScaleFun00_{i,d}() \cdot (ScaleFun01_{i,d}() \cdot M_{i,d}^{k} - ScaleFun02_{i,d}() \cdot N_{i,d}^{k}) \\ i = 1, 2, 3, \ldots, popsize;\ d = 1, 2, 3, \ldots, D;\ k = 1, 2, 3, \ldots, n. \tag{2.2}$$

Here, each $ScaleFun$ represents a scale adjustment function. $ScaleFun01$ and $ScaleFun02$ act on M and N, respectively. $ScaleFun00$ adjusts the difference between M and N. The common forms of $ScaleFun$ are constant functions, simple linear functions, probability functions, or any other complex nonlinear functions.

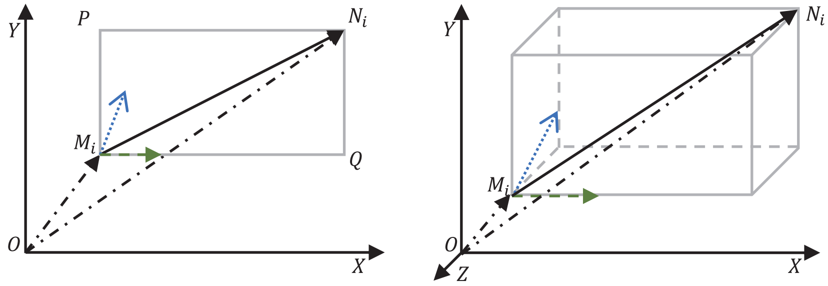


Figure 2.1: Hypercube search style.

The search styles used by most algorithms are based on first-order differences, and the form of expression is hypercube. Fig. 2.1 presents examples of hypercube search styles in two- and three-dimensional spaces. The two black dot-dash arrows M_i and N_i represent two individual vectors. The two black dot-dash arrows M_i and N_i represent two individual vectors. When the cross rate in the differential evolution algorithm (DE) [44] [45] [46] [47] [48] is equal to 1, the black solid arrow M_iN_i can be used to indicate the search trajectory of an updating unit. When the updating unit selects only one dimension at a time, for example, the x-dimension, its search trajectory can be represented by a green dotted arrow. If the updating unit has different scales in different dimensions, such as in the PSO [49] [50] [51] [52] [53], its search trajectory in the two-dimensional space can be represented by the blue dotted arrow. However, these search trajectories cannot break through the search scope of the rectangle PN_iQM_i. In high-dimensional space, the rectangle becomes a hypercube.

Distinguished from the NMHs mentioned above, the SE assumes spherical search style.

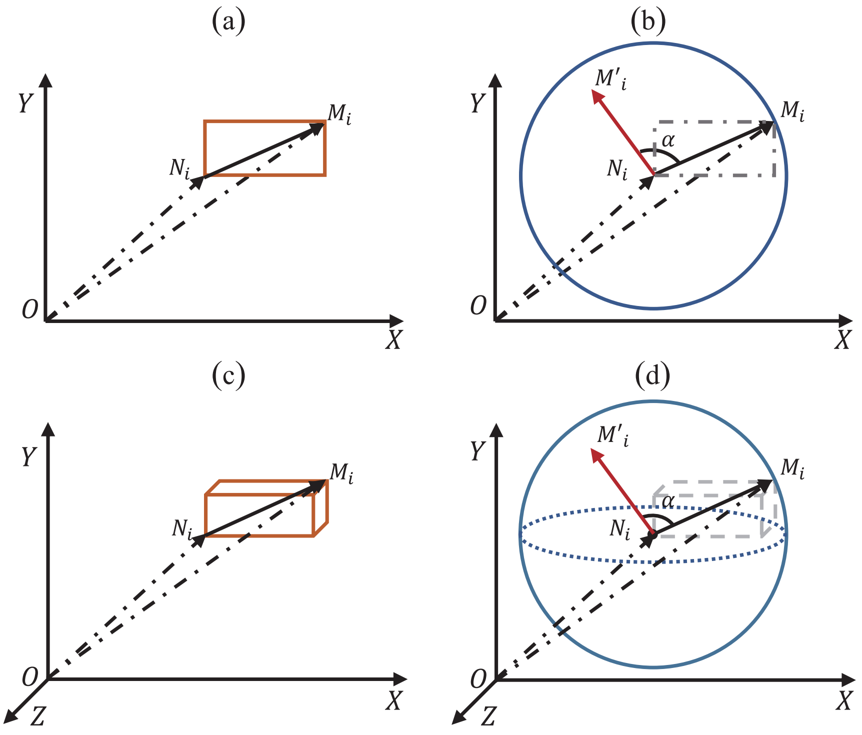


Figure 2.2: Spherical search style in 2-dimensional and 3-dimensional space.

In the SE updating unit, the search trajectory is determined not by the two sides of the rectangle, such as M_iP and M_iQ as depicted in Fig. 2.2, but by the angle and radius. Fig. 2.2(b) depicts the search style of the SE in two-dimensional space. OM_i and ON_i are two individual vectors. The area of a circle with the point N_i as the center and the length M_iN_i as the radius is the maximum search range of the SE updating unit. By adjusting the angle α and the radius length, a new point M'_i can be generated. If the range of α is $[0, 2\pi]$ and the length of the radius is $[0, M_iN_i]$, the maximum search range can be reached. Comparing Fig. 2.2(a) and Fig. 2.2(b), we can observe that the search range of the SE is larger than that of the hypercube search. Additionally, Figs. 2.2(c) and (d) depict instances of updating units in three-dimensional space. The search style of the SE in a high-dimensional space can be expressed as Eqs. (2.3) – (2.5):

$$SS_3\left(M_{i,d}, N_{i,d}\right) = ScaleFun_{i,d}\left(\right) \cdot \left\|\left(M_{i,*} - N_{i,*}\right\|_2 \cdot \prod_{k=d}^{D-1} sin(\theta_d),\ d = 1 \tag{2.3}$$

$$SS_3\left(M_{i,d}, N_{i,d}\right) = ScaleFun_{i,d}\left(\right) \cdot \left\|\left(M_{i,*} - N_{i,*}\right\|_2 \cdot \cos\left(\theta_{d-1}\right) \cdot \prod_{k=d}^{D-1} \sin\left(\theta_d\right),\ 1 < d \leq D-1 \tag{2.4}$$

$$SS_3\left(M_{i,d}, N_{i,d}\right) = ScaleFun_{i,d}\left(\right) \cdot \left\|\left(M_{i,*} - N_{i,d*}\right\|_2 \cdot \cos\left(\theta_{d-1}\right),\ d = D \tag{2.5}$$

where the Euclidean normal form $\left\|\left(M_{i,*} - N_{i,*}\right\|_2$ represents the distance between the vectors $M_{i,*}$ and $N_{i,*}$. Using a scale function $ScaleFun_{i,d}$, the actual search radius is adjusted. θ_d $(d = 1, 2, ..., D-2)$ is a random number with a uniform distribution within $[0, \pi]$, and θ_d $(d = D-1)$ is a random number with a uniform distribution within $[0, 2\pi]$. According to the search pattern and search style, the SE proposes seven spherical evolution operators; we adopted the most promising one [19], as expressed by Eq. (2.6):

$$x_{i,d}^{new} = x_{r1,d} + SS\left(x_{r2,d}, x_{r3,d}\right) \tag{2.6}$$

Here, $x_{r1,d}$, $x_{r2,d}$, and $x_{r3,d}$ represent three random individuals, and i and d represent the index of the individual and the dimension, respectively. The process of the SE updating the population is represented by Algorithm 1.

Algorithm 1: Spherical evolution

```
Function SE():
    for $i = 1 : popsize$ do
        Randomly pick an individual $x_{r1}$ as the parent
        Randomly select two individuals $x_{r2}$ and $x_{r3}$ to participate in the updating
          unit $SS$
        $x_i^{new} = x_{r1} + SS(x_{r2}, x_{r3})$
```

2.3 Adaptive chaotic spherical evolution algorithm

2.3.1 CLS in CSE

The CSE is a combination of SE and chaotic local search to expect a higher global convergence speed as well as a higher possibility of jumping out of the local optimum. In this CLS, the standard random number generator is replaced by chaotic sequences. The chaotic values are derived from J chaotic sequences to adjust the search radius. This CLS is executed on the best individual in each iteration of the SE. The manner in which it works can be expressed by Eq. (2.7):

$$\begin{aligned} x_{g'}^{j}(k) = x_g(k) + v^j(k) \cdot (x_{r1}(k) - x_{r2}(k)),\ j &= 1, 2, 3, \ldots, J \\ k &= 1, 2, 3, \ldots, MaxIter \end{aligned} \tag{2.7}$$

where $x_g(k)$ represents the global best individual in the kth generation, and $x_{r1}(k)$ and $x_{r2}(k)$ are two random individuals in this generation. $v^j(k)$ is the random number generated by the jth chaotic map, and its value is between 0 and 1. $x_{g'}^{j}(k)$ represents the individual searched by the CLS.

Roulette wheel selection method is used here, and the ratio of the roulette area is determined by the intensity of success caused by each chaotic map. The roulette wheel is a mechanism that selects based on probability; having a larger area on the roulette wheel gives a greater chance of being selected. Different from previous usage of CLS, we innovatively propose a new variable, namely success intensity, to control the utilization of chaotic maps. The success intensity expresses the positive change that a certain chaotic map brings to the development of the algorithm. When a particular chaotic map causes greater improvements to an algorithm than others, we believe that it should be accorded a higher probability of being selected in the next iteration. Considering that different chaotic maps may have varying effects at various stages of algorithm convergence, we set a length L for historical information. This means that only the recent Lth generation historical information will have an impact on the new generation. The probability of a chaotic map being selected in the roulette, $P_{k,j}$, can be expressed as Eqs. (2.8)–(2.10):

$$\begin{aligned} P_{k,j} &= \frac{p_{k,j}}{\sum_{j=1}^{J} p_{k,j}} \\ p_{k,j} &= \frac{\omega_{k,j}}{\sum_{j=1}^{J} \omega_{k,j}} + \frac{1}{J} \end{aligned} \tag{2.8}$$

$$\omega_{k,j} = \begin{cases} \sum\limits_{n=1}^{k} \Delta_{n,j} & k \leq L \\ \sum\limits_{n=k-L}^{k} \Delta_{n,j} & k > L \end{cases} \tag{2.9}$$

$$\Delta_{n,j} = f\left(x_g(k)\right) - f\left(x_{g'}^{j}(k)\right) \tag{2.10}$$

where $x_g(k)$ denotes the current optimal solution, $x_{g'}^{j}(k)$ represents the temporary solution generated by CLS, and f is the fitness function. It is notable that Eqs. (2.8)–(2.10) are executed only when $f\left(x_{g'}^{j}(k)\right) < f\left(x_g(k)\right)$. The $\Delta_{n,j}$ represents the success intensity of each successful local search of the jth chaotic map, n represents the number of successes, and $\omega_{k,j}$ represents the sum of the intensities of its success in the Lth generation. In $p_{k,j}$, $\frac{1}{J}$ is introduced to avoid the situation in which certain chaotic sequences are not selected and the probability is zero. $P_{k,j}$ is the actual probability of each chaotic map being selected in the actual roulette. A pseudocode of CLS used in CSE is given in Algorithm 2 for a clearer understanding.

To represent the operation of CLS in the CSE more intuitively, Fig. 2.3 depicts an illustrative process of its implementation. In summary, the intensity of success produced by each chaotic map in the L generation determines its probability of being selected in a roulette. The actually chosen chaotic map j will act on the difference between two random individuals x_{r1} and x_{r2}. The current best individual generates a temporary individual based on this mechanism. This whole process is the CLS of the CSE. Furthermore, Fig. 2.4

presents the flowchart of the CSE.

Algorithm 2: Chaotic local search

```
Function CLS(l, L, j):
    Select two individuals, $x_{r1}$ and $x_{r2}$, at random.
    Pick the currently best individual $x_g$
    Select the value $v$ from the $j$th chaotic sequence
    $x'_g \leftarrow x_g + v \cdot (x_{r1} - x_{r2})$
    if $x'_g < x_g$ then
        $x_g \leftarrow x'_g$;
        $l \leftarrow l + 1$
    Update $\omega$
    Calculate $p$ according to Eq. (2.9)
    Calculate $P$ according to Eq. (2.8)
    Set roulette according to $P$
    Use roulette to determine the chaotic map $j$
```

2.3.2 Description of chaotic maps and sequences

The chaotic sequence used to adjust the parameters is generated by the chaotic maps. Table 2.1 lists the formulas and parameter descriptions of the J ($J = 12$) chaotic maps used in this experiment, where z_k represents the kth chaotic number and z_0 represents the initial one. It should be noted that the range of random values generated by the iterative chaotic map with infinite collapses (ICMIC) is $(-1, 0) \cup (0, 1)$. When the value is negative, we consider the absolute value.

The histogram presented in Fig. 2.5 —a diagram for observation— records the distribution of the values generated by the 12 chaotic maps. Its horizontal axis represents the interval in which the value is located, and its vertical axis represents the number of values in the interval. It can be observed that the distributions under different chaotic maps are different. For example, compared to other maps, the circle map has a higher probability of producing values within [0.4, 0.6]. The range of the sinusoidal map is distinctive in that it can only take values within (0.48, 0.92). The other chaotic maps have a range of values within (0, 1), which leads the sinusoidal map to provide a different search range in the CLS. Since all chaotic maps have a probability of being selected, this allows the diversity of the

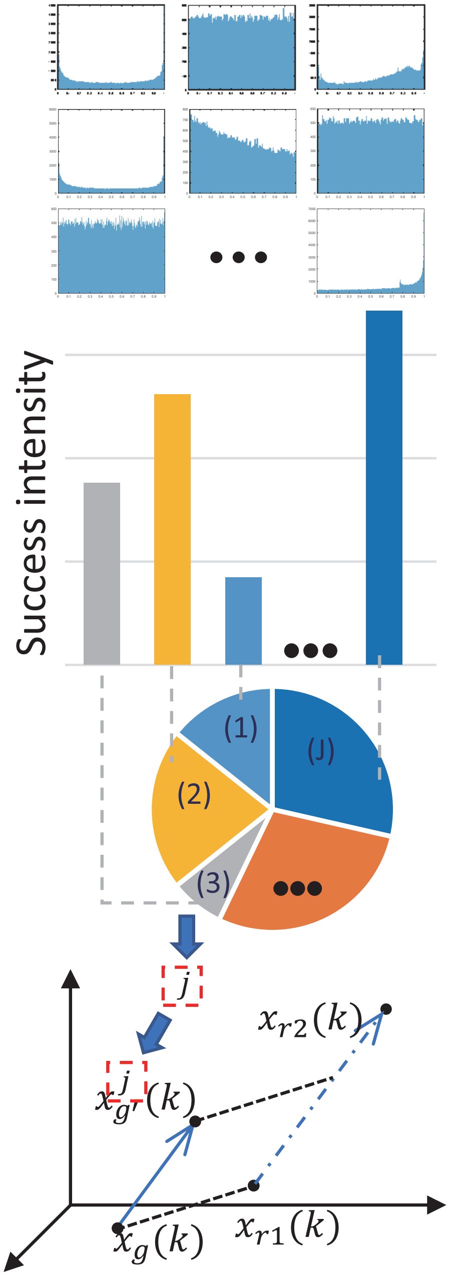


Figure 2.3: Illustrative process of the implementation of CLS.

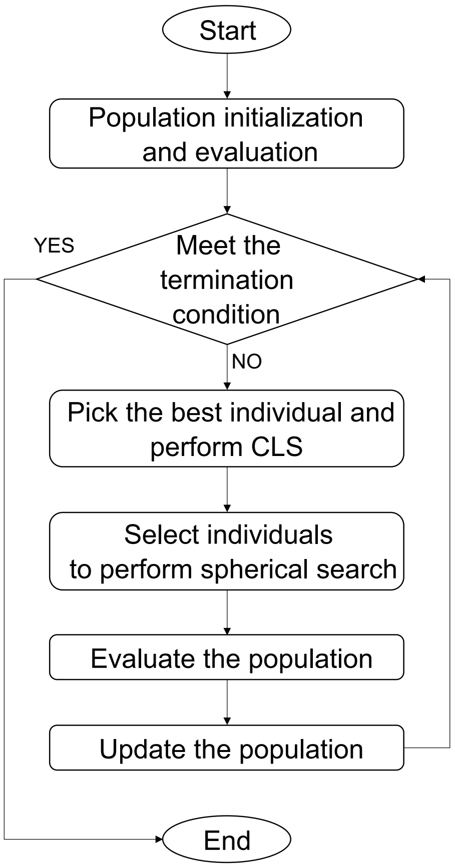

Start
Population initialization and evaluation
YES
Meet the termination condition
NO
Pick the best individual and perform CLS
Select individuals to perform spherical search
Evaluate the population
Update the population
End

Figure 2.4: Flowchart of CSE.

Table 2.1: The formula and parameter setting of chaotic maps.

Chaotic maps	Formula	Parameter
Logistic map	$z_{k+1} = \mu z_k \left(1 - z_k\right)$	$\mu = 4, z_0 = 0.152$
PWLCM map	$z_{k+1} = \begin{cases} z_k/p, & z_k \in (0, p) \\ (1 - z_k)(1 - p), & z_k \in [p, 1) \end{cases}$	$p = 0, 7, z_0 = 0.002$
Singer map	$z_{k+1} = \mu\left(7.86 z_k - 23.31 z_k^2 + 28.75 z_k^3 - 13.302875 z_k^4\right)$	$\mu = 1.073, z_0 = 0.152$
Sine map	$z_{k+1} = \frac{a}{4} \sin\left(\pi z_k\right)$	$a = 4, z_0 = 0.152$
Gaussian map	$z_{k+1} = \begin{cases} 0, & z_k = 0 \\ (\mu/z_k) \bmod (1), & z_k \neq 0 \end{cases}$	$\mu = 1, z_0 = 0.152$
Tent map	$z_{k+1} = \begin{cases} z_k/\beta, & 0 < z_k \leq \beta \\ (1 - z_k)/(1 - \beta), & \beta < z_k \leq 1 \end{cases}$	$\beta = 0.4, z_0 = 0.152$
Bernoulli map	$z_{k+1} = \begin{cases} z_k/(1 - \lambda), & 0 < z_k \leq 1 - \lambda \\ (z_k - 1 + \lambda)/\lambda, & 1 - \lambda < z_k < 1 \end{cases}$	$\lambda = 0.4, z_0 = 0.152$
Chebyshev map	$z_{k+1} = \cos\left(\phi \cos^{-1} z_k\right)$	$\phi = 5, z_0 = 0.152$
Circle map	$z_{k+1} = z_k + a - \frac{b}{2\pi} \sin\left(2\pi z_k\right) \bmod (1)$	$a = 0.5, b = 2.2, z_0 = 0.152$
Cubic map	$z_{k+1} = \rho z_k \left(1 - z_k^2\right)$	$\rho = 2.59, z_0 = 0.242$
Sinusoidal map	$z_{k+1} = a z_k^2 \sin\left(\pi z_k\right)$	$a = 2.3, z_0 = 0.74$
ICMIC map	$z_{k+1} = \sin\left(a/z_k\right)$	$a = 70$

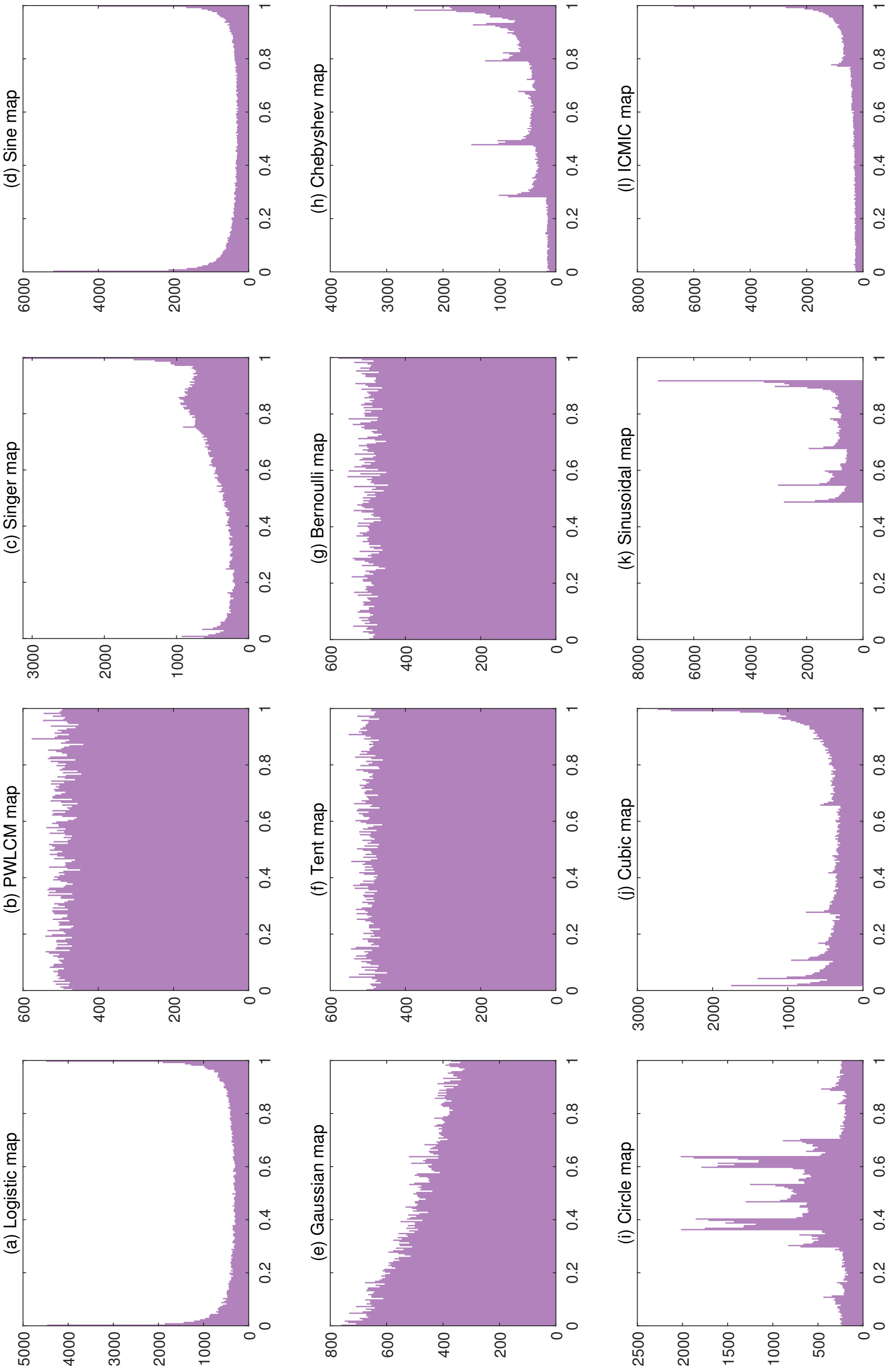


Figure 2.5: Histogram distribution graph of 12 chaotic maps with 10^5 iterations.

CLS to be increased without increasing the complexity of the algorithm.

2.3.3 A selection mechanism to increase the impact of CLS

In the population updating mechanism of the SE, each individual is likely to be involved in generating offspring with equal probability. In the SE population updating mechanism, each individual may participate in updating and producing offspring. We expect that individuals with CLS have a higher chance of participation, which reduces the participation of low-quality individuals. Therefore, a declining updating strategy was introduced. In the early stage of algorithm execution, we allow all individuals to produce offspring. With iterations, the worst individual is prohibited from producing offspring, the number of prohibited individuals increases, and the number of allowed individuals decreases gradually from the population size to one. As CLS is always performed on the best individual, the individuals generated by CLS will always have opportunities; such a possibility increases constantly until only the best one is left at the end. Its update mechanism is expressed through Eqs. (2.11) and (2.12):

$$\begin{aligned} A_{i,d}^{new} = B_{C,d}^{old} + \sum_{k=1}^{n} SS\left(M_{i,d}^{k}, N_{i,d}^{k}\right), i = 1, 2, 3, \ldots popsize, \\ d = 1, 2, 3, \ldots, D \end{aligned} \tag{2.11}$$

$$C = (1 - Iter/MaxIter) \cdot popsize \tag{2.12}$$

where $B_{C,d}^{old}$ represents the individual selected for the update and C represents the top C individuals. $Iter$ denotes the current number of iterations, and, as it increases, the value of C decreases and the number of alternative individuals decreases.

2.3.4 Computational Complexity

To calculate the time complexity, the pseudocode of the entire CSE process is presented in Algorithm 3. Combining Algorithm 1 and Algorithm 2, the time complexity required to gather the key steps of the algorithm is as follows:

(1) The population size is N, so the time complexity of initializing the population is $O(N)$.

(2) Each individual needs boundary detection, so it requires $O(N)$ time complexity.

(3) The time complexity of individual evaluation in each iteration is $O(N)$.

(4) The complexity of selecting chaotic sequences is related to the number of chaotic sequences, which is $O(J)$.

(5) The local search is performed only once per iteration so its time complexity is $O(1)$.

(6) The time complexity of SE to generate new individuals is $O(N)$.

If the algorithm performs K iterations in total, its time complexity is shown as follows

$$\begin{aligned} & O(N) + K[O(N) + O(N) + O(J) + O(1) + O(N)] \\ & = O(N) + 3K \cdot O(N) + K \cdot O(J) + K \cdot O(1) \\ & = (3K + 1) \cdot O(N) + K \cdot O(J) + K \cdot O(1) \end{aligned} \tag{2.13}$$

In this paper, J is a constant and is smaller than the population size N. Therefore, the total time complexity of the CSE is $O(N)$, which is the same as that of SE.

2.4 Experimental Results and Analysis

Herein, the CSE is compared with several other algorithms that are divided into two types. As the proposed CSE is improved based on the SE, the first type of algorithm is the SE alone. If CSE is better than SE, the improvement can be proven effective. Moreover, we expect the improved algorithm to be significantly competitive. Therefore, the second

Algorithm 3: CSE

```
Function CSE():
    Population initialization
    Randomly select the $j - th$ map from $L$ chaotic maps
    Set historical information length $L$
    $l \leftarrow 0$
    for $NFEs = 1 : FES$ do
        Population evaluation
        Perform $CLS$
        Perform $SE$
```

type of algorithm used for comparison is the mainstream algorithm with a strong ability to search. Additionally, the CSE contains a preset parameter L. Its value affects the performance of the algorithm. Therefore, we added certain experiments to find a suitable value for L before the other experiments.

2.4.1 Description of the test set

To express the searching capability of each algorithm, the benchmark function set CEC2017 [54] and the real-world optimization problem set CEC2011 [55] are used.

The CEC2017 contains 29 test functions. F1-F2 are unimodal functions, F3-F9 are simple multimodal functions, F10-F19 are hybrid functions, and F20-F29 are composition functions. The CEC2011 contains 13 real-world optimization problems, such as parameter estimation for frequency-modulated sound waves, Lennard Jones potential problems, and bifunctional catalyst blend control problems. Among these, the 11th one contains 10 subproblems. Therefore, we consider CEC2011 as a set of 22 problems. Thus, the 10 subproblems of the 11th problem are relabeled as F11-F20, and the 12th and 13th problems are relabeled as F21 and F22.

2.4.2 Experiment setup

In the experiment, we set the same number of evaluations as the termination condition to ensure the fairness of the experiment. The number of evaluations is equal to $D * 10^4$, where D is the dimension of the benchmark function. As suggested in [19], the SE can achieve ideal results when there are 20 individuals; therefore, we set the population size to 20 for both the SE and CSE. Other algorithms set the population size to 100 according to the original paper. In addition, in CEC2017, we conducted experiments on 30 and 50 dimensions respectively. In CEC2011, the dimensions of different problems are different, and we use the preset dimension of each question. Finally, all experiments were performed using MATLAB on a PC with a 3.00 GHz Intel(R) Core(TM) i7-9700 CPU and 8 GB of RAM. To achieve accurate results, all experiments were run independently 30 times.

2.4.3 Discussion of the parameter L

The parameter L determines the impact of historical information on the algorithm. Considering that the algorithm may need the guidance of different chaotic maps in different convergence stages, the historical information L must have timeliness. When the value of L is too large, the selection will be affected by the early information, leading to inaccurate judgment of the current state. On the contrary, Too little use of information can lead to loss of reference information. Under the same termination conditions, the difference in population size makes the number of iterations different. We believe that setting L as a parameter linked to the number of iterations can improve the correlation with historical information. In the comparison experiment, we set the value of L to MaxIter/100, MaxIter/300, MaxIter/500, MaxIter/800, and MaxIter/1000, where MaxIter denotes the maximum number of iterations.

In the comparison experiment, we used the CEC2017 test set on 30 dimensions and computed the average of the final results of 30 experiments. We used the Friedman test [56] to rank the five groups of experimental results. The Friedman test, a non-parametric statistical test, is suitable for use with data that have no significant difference in distribution. It can calculate the individual ranking of these data with respect to each question (there are 29 instances in all) and calculate a comprehensive ranking. The smaller the ranking value, the better. As can be seen from Table 2.2, when L = MaxIter/300, the ranking value is the smallest. Therefore, in subsequent experiments, the value of L was set to MaxIter/300.

Table 2.2: Friedman test for parameter *L*.

L	Friedman Rank	Final Rank
MaxIter/100	3.2759	4
MaxIter/300	2.3448	1
MaxIter/500	3	2
MaxIter/800	3.069	3
MaxIter/1000	3.3103	5

2.4.4 Discussion on single chaotic map and multiple chaotic maps

Some other issues worth discussing are whether chaos plays a role in local search and whether the selection strategy of multi-chaotic maps is better than the single chaotic map local search. Therefore a set of comparison experiments based on the CEC2017 test set are implemented. In the experiments, the CSE algorithm with multiple chaotic maps proposed in this paper is labeled as CSE_M, the 12 CSEs using different single chaotic map are labeled as CSE_1 to CSE_12, respectively, and the one not using chaotic map is labeled as NoChaos. NoChaos can be expressed as Eq. (2.14). Table 2.3 gives the Friedman test results of the experiment. It can be seen that NoChaos has the largest ranking value, which means that the chaos strategy in local search does make the algorithm improved. And CSE_M achieves the smallest ranking value, which shows that the multi-chaotic maps strategy is more competitive than the single chaotic map.

$$\begin{aligned} x_{g'}^{j}(k) = x_g(k) + (x_{r1}(k) - x_{r2}(k)),\ j = 1, 2, 3, \ldots, J \\ k = 1, 2, 3, \ldots, MaxIter \end{aligned} \tag{2.14}$$

2.4.5 Comparison experiments with SE

This part of the comparative experiment content includes the experimental data table, part of the convergence curve, and a box-and-whisker diagram. Additionally, we have presented several convergence trajectories for CSE to describe its ability to jump out of the local optimum.

Table 2.3: Friedman test for multi-chaotic maps discussion.

Algorithm	Friedman Rank	Final Rank
CSE_M	5.5517	1
CSE_1	7.069	6
CSE_2	6.5862	3
CSE_3	7.8621	12
CSE_4	6.6897	4
CSE_5	6.6897	4
CSE_6	7.3793	11
CSE_7	7.3103	10
CSE_8	6.1379	2
CSE_9	7.2069	8
CSE_10	7.2414	9
CSE_11	10.5172	13
CSE_12	7.1724	7
NoChaos	11.5862	14

Table 2.4: Experiment data of CSE versus SE on CEC2017 30 dimensions.

Algorithm	CEC2017 F1	CEC2017 F2	CEC2017 F3	CEC2017 F4	CEC2017 F5
CSE	**3.55E-05 ± 7.96E-05**	**3.36E+01 ± 3.27E+01**	**5.37E+01 ± 3.81E+01**	**3.60E+01 ± 7.43E+00**	6.29E-02 ± 9.01E-02
SE	2.75E+03 ± 4.41E+03 +	6.54E+03 ± 2.13E+03 +	8.54E+01 ± 1.79E+01 +	4.40E+01 ± 5.80E+00 +	**1.65E-04 ± 5.47E-04** −
	CEC2017 F6	CEC2017 F7	CEC2017 F8	CEC2017 F9	CEC2017 F10
CSE	**6.67E+01 ± 6.98E+00**	**3.68E+01 ± 7.97E+00**	3.54E-01 ± 4.12E-01	**1.87E+03 ± 2.74E+02**	3.00E+01 ± 2.53E+01
SE	7.83E+01 ± 8.10E+00 +	4.73E+01 ± 9.07E+00 +	6.97E-01 ± 2.07E+00 ≈	2.35E+03 ± 3.03E+02 +	2.42E+01 ± 1.94E+01 ≈
	CEC2017 F11	CEC2017 F12	CEC2017 F13	CEC2017 F14	CEC2017 F15
CSE	**4.38E+04 ± 2.69E+04**	**5.93E+01 ± 2.88E+01**	**4.70E+01 ± 2.09E+01**	**2.33E+01 ± 1.57E+01**	**4.76E+02 ± 1.21E+02**
SE	9.29E+05 ± 8.73E+05 +	5.89E+03 ± 7.32E+03 +	7.78E+04 ± 8.06E+04 +	1.90E+03 ± 1.46E+03 +	5.75E+02 ± 1.15E+02 +
	CEC2017 F16	CEC2017 F17	CEC2017 F18	CEC2017 F19	CEC2017 F20
CSE	9.04E+01 ± 6.29E+01	**3.14E+03 ± 2.17E+03**	**1.16E+01 ± 2.90E+00**	**1.27E+02 ± 8.76E+01**	2.41E+02 ± 6.75E+00
SE	8.45E+01 ± 6.49E+01 ≈	1.87E+05 ± 1.19E+05 +	1.97E+03 ± 1.63E+03 +	1.59E+02 ± 8.24E+01 +	2.44E+02 ± 1.10E+01 ≈
	CEC2017 F21	CEC2017 F22	CEC2017 F23	CEC2017 F24	CEC2017 F25
CSE	**3.29E+02 ± 7.03E+02**	**3.89E+02 ± 8.95E+00**	**4.80E+02 ± 1.27E+01**	3.88E+02 ± 1.40E+00	1.27E+03 ± 4.59E+02
SE	5.80E+02 ± 1.05E+03 +	3.95E+02 ± 8.32E+00 +	4.79E+02 ± 4.80E+01 +	**3.87E+02 ± 9.42E-01** −	1.04E+03 ± 5.23E+02 ≈
	CEC2017 F26	CEC2017 F27	CEC2017 F28	CEC2017 F29	w/t/l
CSE	5.08E+02 ± 5.89E+00	3.84E+02 ± 6.57E+01	5.14E+02 ± 6.44E+01	**2.97E+03 ± 4.93E+02**	
SE	5.11E+02 ± 4.66E+00 ≈	4.06E+02 ± 4.39E+00 ≈	5.37E+02 ± 7.88E+01 ≈	6.04E+03 ± 1.97E+03 +	19/8/2

Table 2.5: Experiment data of CSE versus SE on CEC2017 50 dimensions.

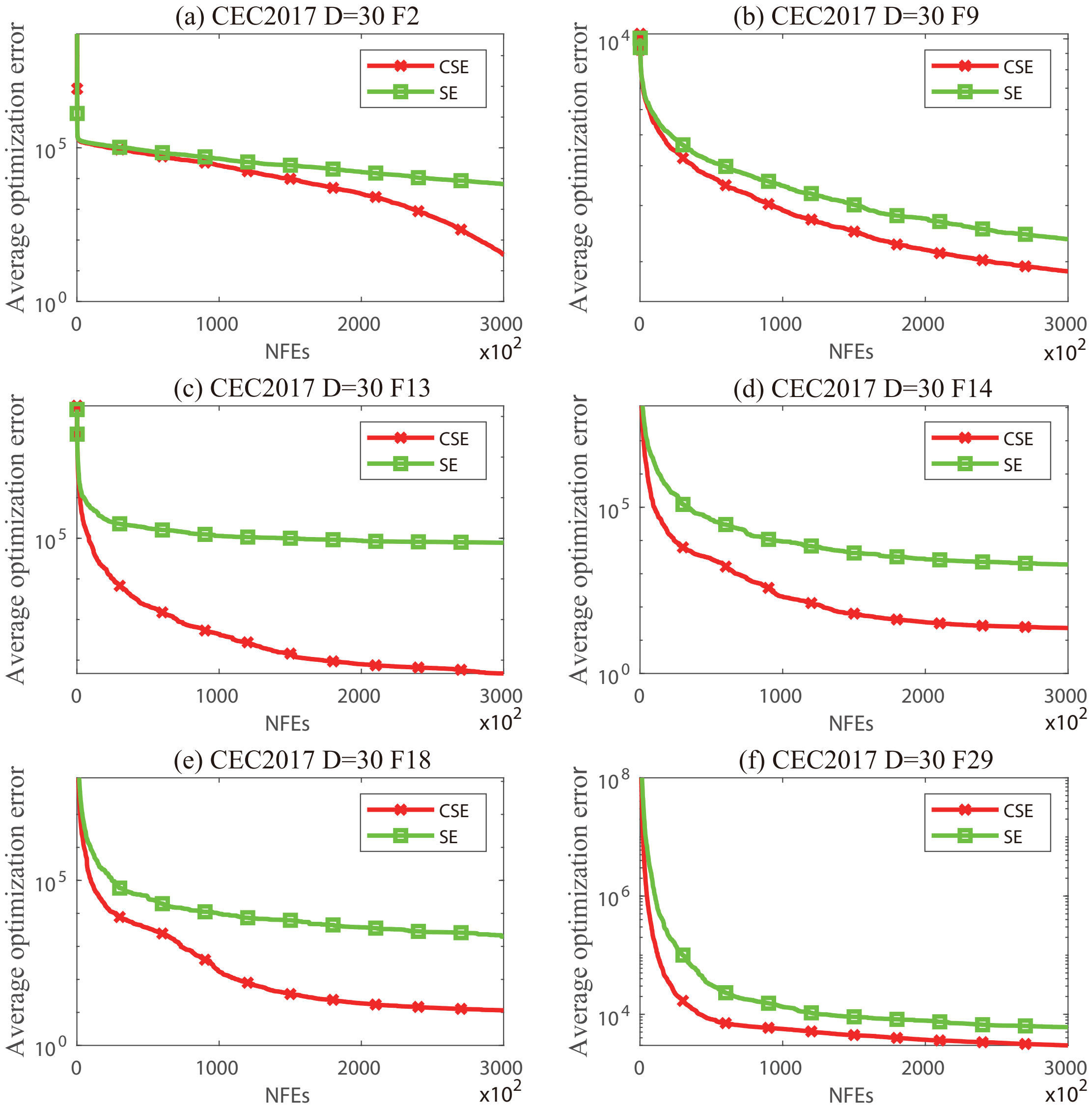
Algorithm	CEC2017 D50 F1	CEC2017 D50 F2	CEC2017 D50 F3	CEC2017 D50 F4	CEC2017 D50 F5
CSE	**2.07E-01 ± 3.60E-01**	**3.59E+03 ± 1.48E+03**	**9.36E+01 ± 4.73E+01**	**8.99E+01 ± 1.17E+01**	4.14E-02 ± 7.57E-02
SE	2.45E+03 ± 3.54E+03 +	4.43E+04 ± 8.07E+03 +	1.04E+02 ± 3.41E+01 ≈	1.04E+02 ± 1.68E+01 +	**1.43E-03 ± 7.62E-03** −
	CEC2017 D50 F6	CEC2017 D50 F7	CEC2017 D50 F8	CEC2017 D50 F9	CEC2017 D50 F10
CSE	**1.45E+02 ± 1.33E+01**	**9.04E+01 ± 1.27E+01**	**1.17E+01 ± 2.39E+01**	**3.80E+03 ± 5.12E+02**	**5.45E+01 ± 8.95E+00**
SE	1.61E+02 ± 1.58E+01 +	1.06E+02 ± 8.63E+00 +	3.57E+01 ± 3.56E+01 +	4.35E+03 ± 5.24E+02 +	6.11E+01 ± 1.20E+01 +
	CEC2017 D50 F11	CEC2017 D50 F12	CEC2017 D50 F13	CEC2017 D50 F14	CEC2017 D50 F15
CSE	**2.82E+05 ± 2.47E+05**	**3.85E+02 ± 4.19E+02**	**1.29E+03 ± 1.51E+03**	**6.53E+01 ± 2.74E+01**	**1.14E+03 ± 2.47E+02**
SE	5.87E+06 ± 2.08E+06 +	3.89E+03 ± 3.49E+03 +	6.67E+05 ± 4.17E+05 +	2.78E+03 ± 2.62E+03 +	1.21E+03 ± 1.94E+02 ≈
	CEC2017 D50 F16	CEC2017 D50 F17	CEC2017 D50 F18	CEC2017 D50 F19	CEC2017 D50 F20
CSE	**6.90E+02 ± 2.03E+02**	**2.98E+04 ± 1.96E+04**	**2.53E+01 ± 6.95E+00**	**5.44E+02 ± 1.54E+02**	**2.97E+02 ± 1.34E+01**
SE	7.41E+02 ± 1.71E+02 ≈	1.81E+06 ± 8.37E+05 +	4.47E+03 ± 3.58E+03 +	6.01E+02 ± 1.56E+02 ≈	3.11E+02 ± 1.46E+01 +
	CEC2017 D50 F21	CEC2017 D50 F22	CEC2017 D50 F23	CEC2017 D50 F24	CEC2017 D50 F25
CSE	**4.04E+03 ± 1.38E+03**	**5.36E+02 ± 2.10E+01**	**6.68E+02 ± 2.77E+01**	**5.24E+02 ± 3.65E+01**	2.20E+03 ± 4.08E+02
SE	4.88E+03 ± 1.33E+03 +	5.44E+02 ± 1.62E+01 ≈	6.83E+02 ± 2.47E+01 +	5.25E+02 ± 1.96E+01 ≈	**2.18E+03 ± 3.63E+02** ≈
	CEC2017 D50 F26	CEC2017 D50 F27	CEC2017 D50 F28	CEC2017 D50 F29	w/t/l
CSE	**5.69E+02 ± 2.86E+01**	5.07E+02 ± 4.56E+00	6.89E+02 ± 1.10E+02	**6.04E+05 ± 2.32E+04**	
SE	6.00E+02 ± 2.17E+01 +	**5.03E+02 ± 1.38E+01** ≈	**6.38E+02 ± 1.14E+02** −	7.77E+05 ± 6.83E+04 +	19/8/2

Table 2.6: Experiment data of CSE versus SE on CEC2011.

Algorithm	CEC2011 F1	CEC2011 F2	CEC2011 F3	CEC2011 F4	CEC2011 F5
CSE	**1.89E+00 ± 4.16E+00**	**-2.58E+01 ± 9.41E-01**	1.15E-05 ± 2.13E-19	2.14E+01 ± 3.87E-01	-3.58E+01 ± 1.03E+00
SE	4.49E+00 ± 4.36E+00 +	-2.41E+01 ± 1.25E+00 +	1.15E-05 ± 2.51E-19 ≈	**1.91E+01 ± 3.12E+00** –	-3.57E+01 ± 9.45E-01 ≈
	CEC2011 F6	CEC2011 F7	CEC2011 F8	CEC2011 F9	CEC2011 F10
CSE	**-2.92E+01 ± 4.20E-03**	**1.12E+00 ± 1.26E-01**	2.20E+02 ± 0.00E+00	**7.44E+02 ± 2.80E+02**	**-2.13E+01 ± 2.47E-01**
SE	-2.91E+01 ± 8.66E-02 +	1.25E+00 ± 9.36E-02 +	2.20E+02 ± 0.00E+00 ≈	8.40E+02 ± 1.52E+02 +	-2.01E+01 ± 1.19E+00 +
	CEC2011 F11	CEC2011 F12	CEC2011 F13	CEC2011 F14	CEC2011 F15
CSE	**5.15E+04 ± 5.34E+02**	**1.73E+07 ± 1.06E+04**	1.55E+04 ± 1.29E+01	**1.92E+04 ± 1.58E+02**	3.30E+04 ± 6.24E+01
SE	5.25E+04 ± 3.39E+02 +	1.73E+07 ± 1.14E+04 +	1.55E+04 ± 9.75E+00 ≈	1.93E+04 ± 1.42E+02 +	**3.30E+04 ± 5.50E+01** –
	CEC2011 F16	CEC2011 F17	CEC2011 F18	CEC2011 F19	CEC2011 F20
CSE	1.34E+05 ± 1.87E+03	**1.93E+06 ± 1.20E+04**	**9.40E+05 ± 1.52E+03**	**1.03E+06 ± 4.23E+04**	**9.40E+05 ± 1.86E+03**
SE	1.35E+05 ± 2.17E+03 ≈	1.93E+06 ± 1.22E+04 +	9.43E+05 ± 3.13E+03 +	1.16E+06 ± 9.27E+04 +	9.43E+05 ± 2.17E+03 +
	CEC2011 F21	CEC2011 F22	w/t/l		
CSE	**1.52E+01 ± 2.27E+00**	1.78E+01 ± 2.73E+00			
SE	1.64E+01 ± 2.15E+00 +	1.83E+01 ± 2.79E+00 ≈	14/6/2		

It should be stated in advance that the optimum value of each problem of CEC2017 is known, and the aim of the experiment is to determine errors with known optimum values. The smaller the error, the better the performance of the algorithm. The CEC2011 problems are real-world optimization problems, and the optimum values are unknown. The purpose of this experiment is to find the minimum value. In the experiment, the result of CEC2011 is the real value evaluated, and the smaller the value, the better the performance of the algorithm. The data presented in Tables 2.4 and 2.5 are the *mean ± standard deviation* of the final results of 30 experiments for CEC2017 on 30 and 50 dimensions, respectively. And Table 2.6 show the same thing for CEC2011. Among them, the better solution for each problem is set in bold font. The content "$w/t/l$" in the last part of the table denotes the result of the Wilcoxon signed rank test [56]. This is a nonparametric statistical test, where it is used to determine whether there is a significant advantage between the two. The significance level was set to 0.05, and if the statistic was less than 0.05 then one side was significantly better than the other. When the CSE is significantly superior to the SE, it is marked as " + ", otherwise it is marked as " – ". When the two algorithms are tied, it is recorded as " ≈ ". That is, for the 29 problems in CEC2017, the CSE won 19 times, failed 2 times, and tied 8 times both on 30 and 50 dimensions. For the 22 questions in CEC2011, the CSE won 14 times, failed 2 times, and tied 6 times.

To visually represent the performance of the algorithms, the convergence graphs of the CSE and SE are illustrated for some typical problems. Fig. 2.6 depicts the change process of the optimum solution obtained by the algorithm as the iterations increase for some problems of CEC2017 on 30 dimensions. The x-axis indicates the number of function evaluations and the *y*-axis indicates the average error value. It can be seen that in the unimodal function F2, the convergence curve for the SE is relatively flat, whereas the CSE still maintains a high convergence speed even in the final stage. For the simple multimodal function F9, the CSE also achieved better results earlier. For the hybrid function and composition functions F13, F14, F18, and F29, the convergence curve of the CSE is lower than that of the SE, which indicates better convergence ability. Fig. 2.7 describes the convergence on 50 dimensions, and some conclusions similar to the above can be drawn.



(a) CEC2017 D=30 F2

(b) CEC2017 D=30 F9

(c) CEC2017 D=30 F13

(d) CEC2017 D=30 F14

(e) CEC2017 D=30 F18

(f) CEC2017 D=30 F29

Figure 2.6: Convergence graphs of CSE versus SE on CEC2017 30 dimensions.

The box-and-whisker diagrams presented in Fig. 2.8 and Fig. 2.9 depict the distribu-

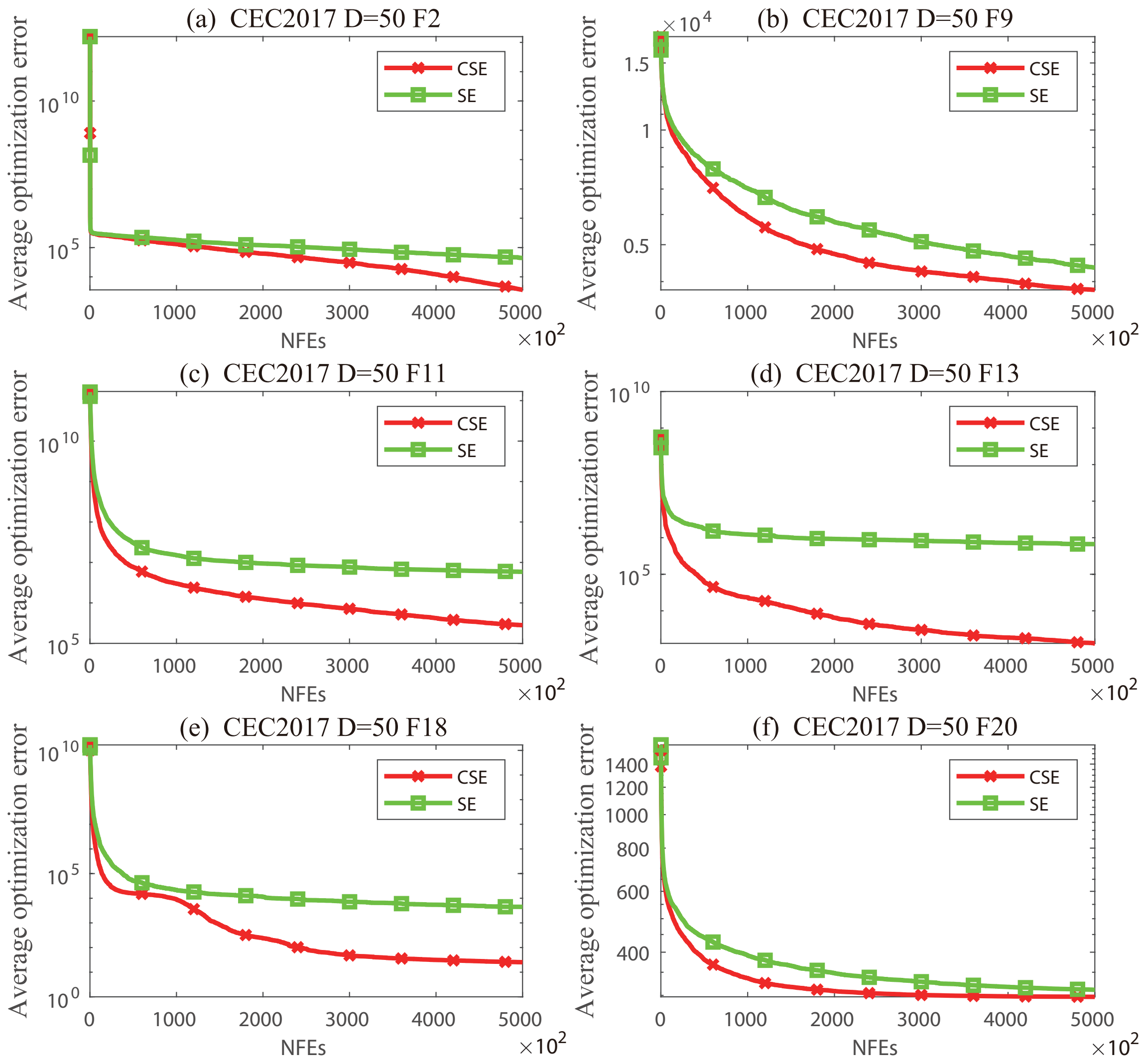


Figure 2.7: Convergence graphs of CSE versus SE on CEC2017 50 dimensions.

tion of the results of the 30 experiments, thus expressing the quality of the solution obtained using the algorithm. The top of the blue box indicates the first quartile and the bottom indicates the third quartile. The red line within the box represents the median. The black line above the blue box indicates the maximum value, and which below the blue box indicates the minimum value. The red " + " represents extreme values. The long distance between the maximum and minimum implies that the solution fluctuates significantly and the performance of the algorithm is unstable. On the contrary, it means that the algorithm has the ability to search out the solution within a certain area stably. In addition, the lower the position of the box the higher the overall quality of the solutions. It can be seen in Fig. 2.8 that with respect to the benchmark function set CEC2017 on 30 dimensions, when compared with the SE, the CSE has excellent stability with respect to the unimodal function F1; hybrid functions F13, F14, and F18; and composition function F29 and can find better solutions. However, although the stability is not evident for the simple multimodal function F9, the overall quality of the solution is significantly better than that obtained using the SE. A similar situation is also reflected in F6 and F20 on 50 dimensions, which is shown in Fig. 2.9. In most cases, the advantages of CSE are more obvious.

Regarding CEC2011, Fig. 2.10 indicates the similar performances of both algorithms; the difference is that the y-axis represents the true value of the evaluation result after being logged. It is evident that the CSE can achieve better results than the SE in most cases. Although the gap between the two is not apparent from Fig. 2.10(d), it is observed that the CSE curve has achieved a lower value.

Fig. 2.11 presents a comparison of the quality of solutions obtained using the CSE and the SE on CEC2011. With respect to function F1, the solution obtained using the CSE is generally better when compared with the SE. Although there are several extreme values, these extreme solutions are better than the maximum value obtained using the SE. For the functions F7 and F11, the fluctuation in the CSE solution is relatively large, but the quality of the solution still exceeds that of the SE solution. For the function F19, the stability of the CSE is evident.

For the CSE, we performed a supplementary experiment to observe its ability to jump out of the local optimum. The CEC2017 is used as the test function set, and the ability of the

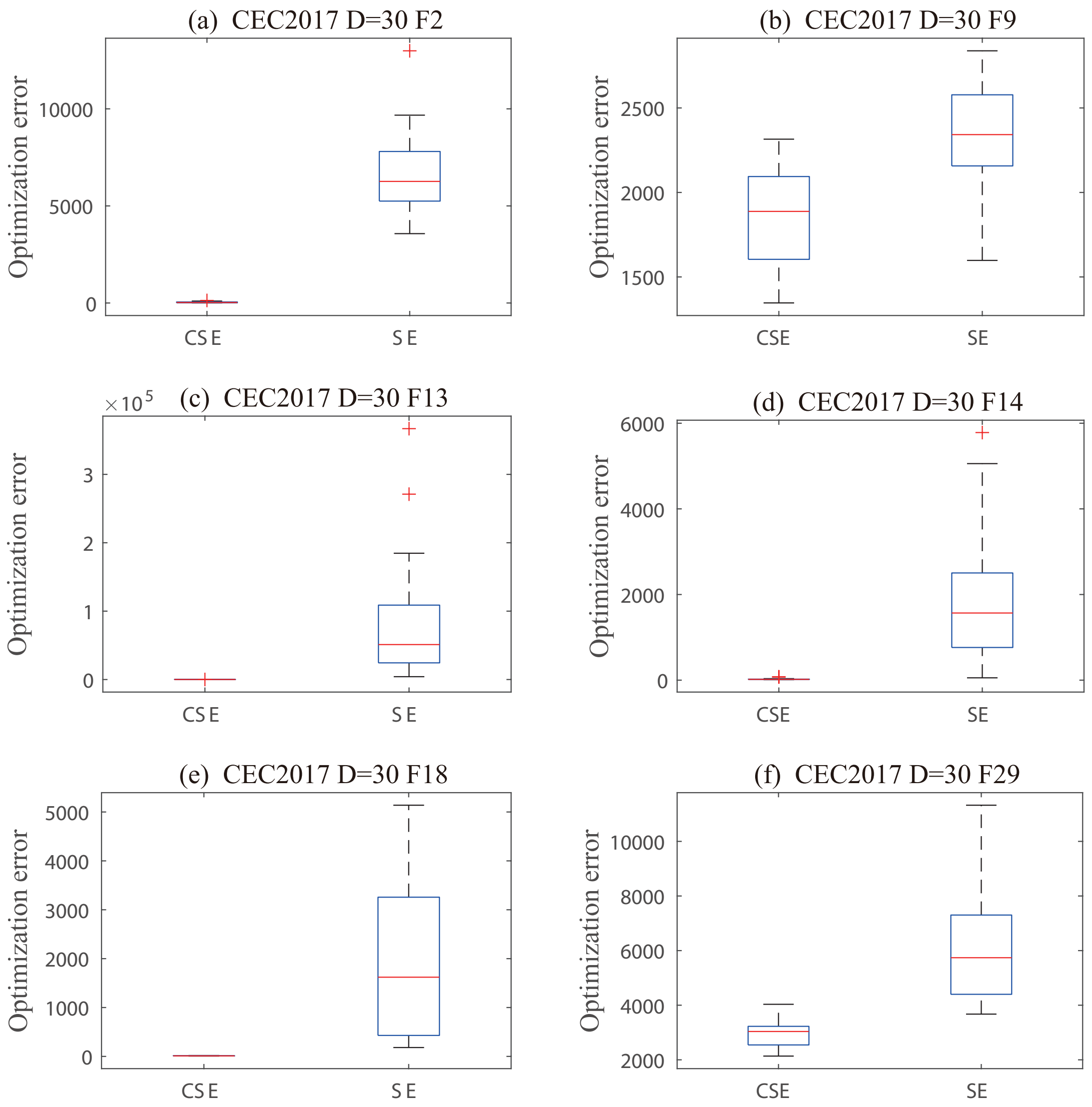


Figure 2.8: Box-and-whisker diagrams of CSE versus SE on CEC2017 30 dimensions.

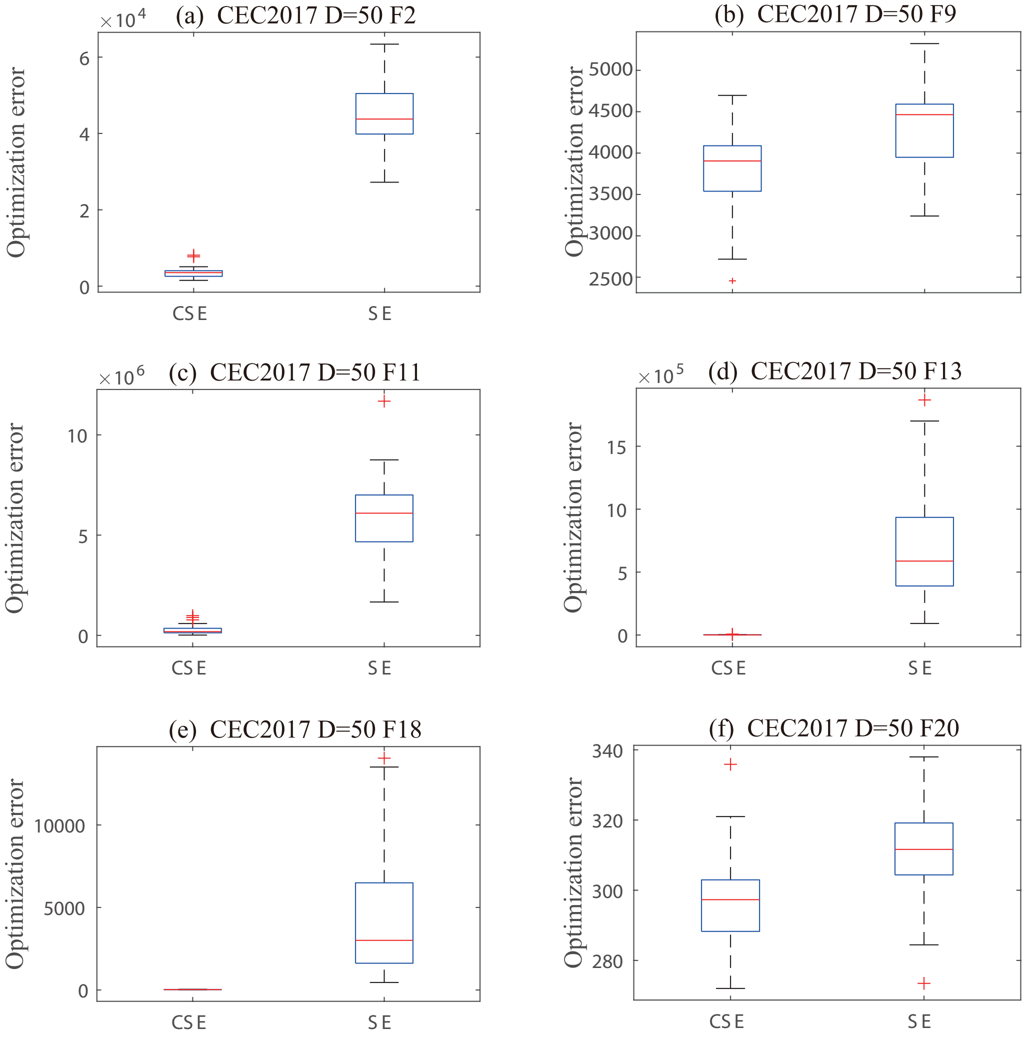


Figure 2.9: Box-and-whisker diagrams of CSE versus SE on CEC2017 50 dimensions.

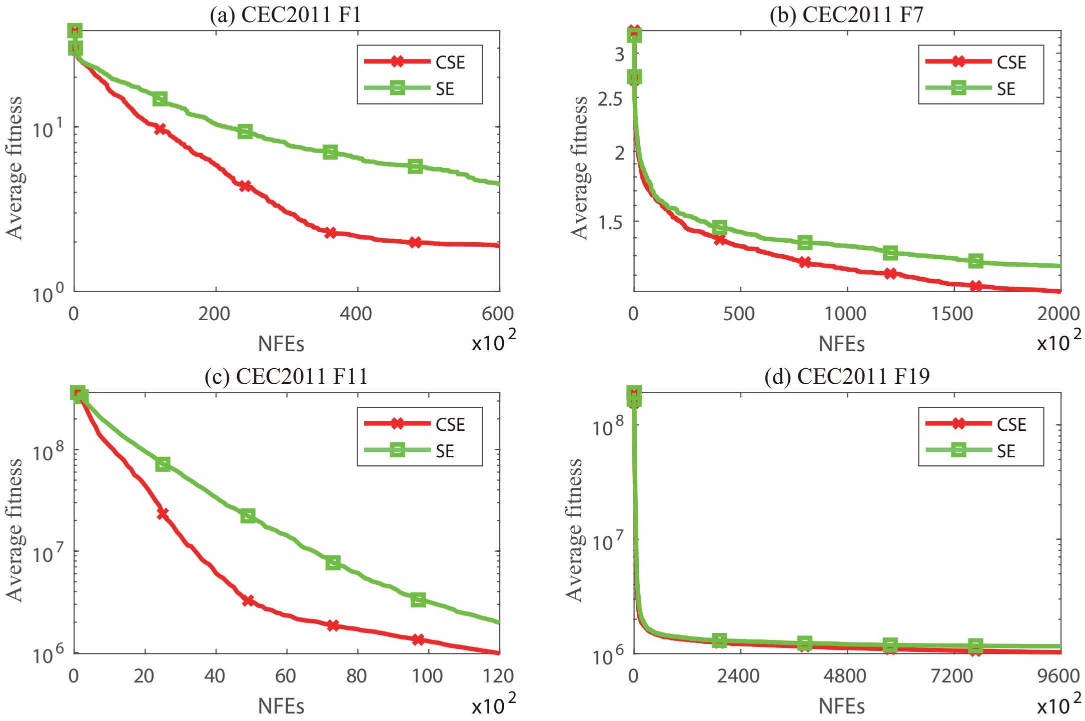


Figure 2.10: Convergence graphs of CSE versus SE on CEC2011.

CSE to jump out of the local optimum is demonstrated through the movement trajectories of individuals in the past generations on the function contour map. Since high-dimensional space images cannot be displayed, the experiment was carried out in a two-dimensional space and used as a reference. Fig. 2.12 depicts the distribution of the population over several iterations of the functions F4 and F22. The *t* in the subtitle represents the number of iterations. In the contour map, bright colors represent peaks, dark colors represent valleys, and the darkest color is the optimum value we are looking for. The blue square dots represent individuals, the orange dots represent the best individuals in the current generation, and the yellow triangles represent temporary individuals generated by CLS. For F4 and F22, we can see that the population is scattered at the beginning, and over some iterations, the individuals move closer to a certain local optimum. However, the population does not stagnate; but the algorithm gradually explores other regions and finally finds the global optimum.

Based on these experiments, it can be concluded that whether it is a benchmark function or a real-world optimization problem, the CSE performs better than the SE. This conclusion

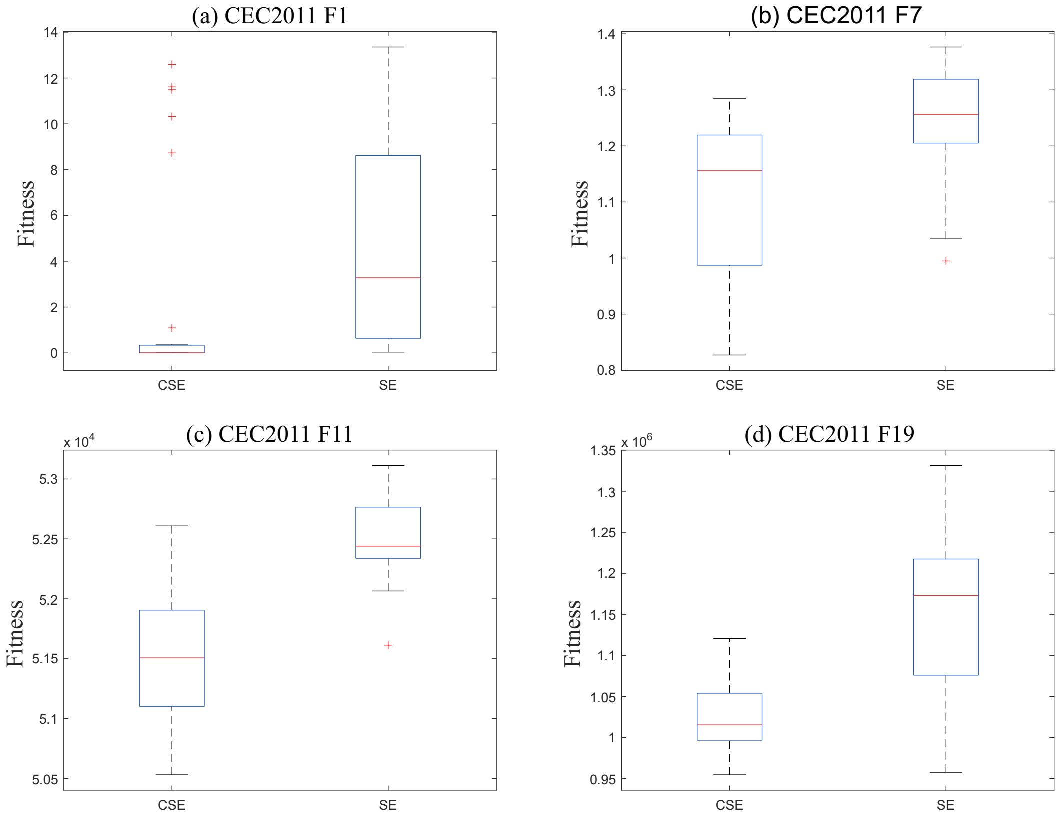


Figure 2.11: Box-and-whisker diagrams of CSE versus SE on CEC2011.

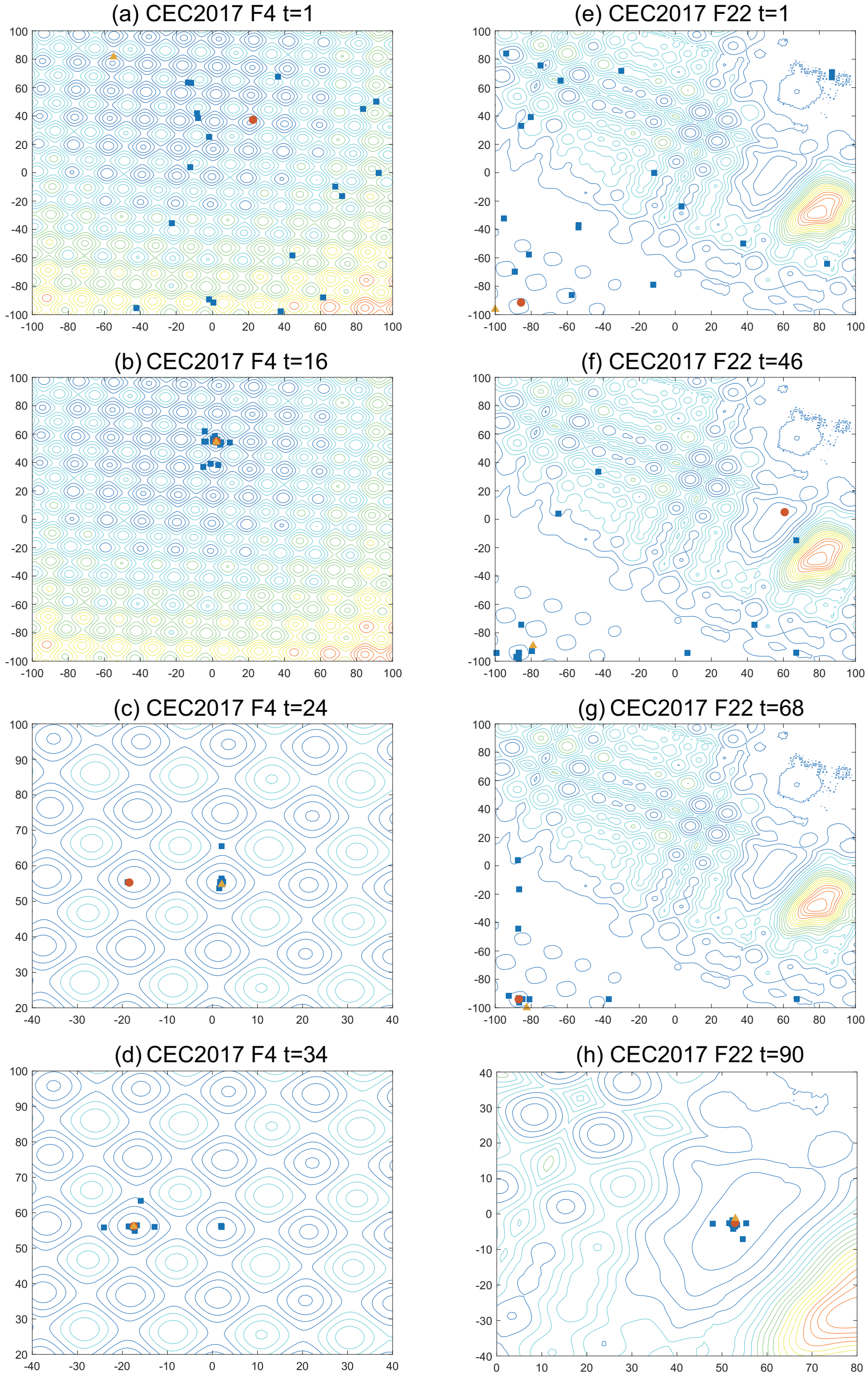


Figure 2.12: Search trajectories of CSE on CEC2017.

Table 2.7: Parameter setting of the compared algorithms.

Algorithm	Parameters
HGSA	G_0 = 100,L = 20, $w_1(t)$ = 1 - t^6/T^6,$w_2(t)$ = t^6/T^6
IGSA	$\alpha_{mean}(0)$ = 20, σ = 0.3, p = 0.1, k = 6
RGBSO	K = 5, $P_{clus/1}$ = 0.8, $P_{cen/1}$ = 0.4, $P_{cen/2}$ = 0.5
MDBSO	K = 5, μ = 0.5
SCA	α = 2
DE	F = 0.5,CR = 0.9

is manifested in two aspects: the improvement of the global convergence speed and the ability to jump out of a local optimum, which is the motivation for including CLS initially. The proposed improvement is definitely effective.

2.4.6 Comparison with other algorithms

Algorithms used for comparison include HGSA [57], IGSA [58], RGBSO [59], MDBSO [60], SCA [61], and DE [45]. These algorithms are all competitive with mainstream algorithms. Table 2.7 presents the parameter settings of each algorithm in the experiment.

The experimental results of the 29 problems of CEC2017 are presented in Table 2.8 and Table 2.9. Compared with the other six algorithms, in 29 benchmark function cases, CSE got the first place 12 times on 30 dimensions and 11 times on 50 dimensions. These functions include all the function types of the CEC2017, which indicates that the CSE is competitive with respect to a variety of test functions. From the results presented in the "$w/t/l$" column, it can be seen that the CSE has also achieved evident advantages in one-to-one competitions.

Furthermore, a comparison with respect to the real-world optimization problem set CEC2011 has also been considered. The results of the comparison are presented in Table 2.10. In the mean and standard deviation statistics, the CSE won first place 15 times out of 22 questions. It is also evident from the results of the Wilcoxon rank-sum test that the performance of the CSE is significantly better when compared with other algorithms for most problems.

Fig. 2.13 shows the convergence status of the algorithms on 30 dimensions of CEC2017. In F4, F9 and F20, CSE converges weaker than IGSA in the early stage, but it still has a

Table 2.8: Experiment data of CSE versus its peers on CEC2017 30 dimensions.

Algorithm	CEC2017 D30 F1	CEC2017 D30 F2	CEC2017 D30 F3	CEC2017 D30 F4	CEC2017 D30 F5
CSE	3.55E-05 ± 7.96E-05	3.36E+01 ± 3.27E+01	**5.37E+01** ± **3.81E+01**	**3.60E+01** ± **7.43E+00**	6.29E-02 ± 9.01E-02
HGSA	2.58E+03 ± 2.50E+03 +	4.33E+04 ± 5.49E+03 +	1.19E+02 ± 2.63E+00 +	1.53E+02 ± 1.28E+01 +	8.19E+00 ± 4.54E+00 +
IGSA	1.78E+03 ± 1.37E+03 +	6.01E+04 ± 7.02E+03 +	1.22E+02 ± 2.10E+01 +	4.24E+01 ± 8.18E+00 +	3.83E-03 ± 1.29E-02 −
RGBSO	2.40E+03 ± 2.90E+03 +	2.36E+04 ± 7.78E+03 +	7.52E+01 ± 1.10E+01 +	1.37E+02 ± 2.49E+01 +	1.25E+00 ± 8.14E-01 +
MDBSO	3.07E+03 ± 4.68E+03 +	3.13E+02 ± 6.28E+02 ≈	6.18E+01 ± 3.28E+01 ≈	1.02E+02 ± 3.10E+01 +	1.31E+01 ± 7.18E+00 +
SCA	1.18E+10 ± 1.77E+09 +	3.47E+04 ± 6.31E+03 +	1.00E+03 ± 2.74E+02 +	2.71E+02 ± 2.17E+01 +	4.93E+01 ± 5.34E+00 +
DE	**6.63E-15** ± **7.21E-15** −	**1.85E+01** ± **9.58E+00** ≈	5.93E+01 ± 1.92E+00 −	1.75E+02 ± 1.25E+01 +	**1.15E-08** ± **2.67E-08** −
	CEC2017 D30 F6	CEC2017 D30 F7	CEC2017 D30 F8	CEC2017 D30 F9	CEC2017 D30 F10
CSE	6.67E+01 ± 6.98E+00	3.68E+01 ± 7.97E+00	3.54E-01 ± 4.12E-01	**1.87E+03** ± **2.74E+02**	**3.00E+01** ± **2.53E+01**
HGSA	**4.08E+01** ± **3.01E+00** −	9.98E+01 ± 9.03E+00 +	7.96E-14 ± 5.30E-14 −	3.21E+03 ± 2.93E+02 +	9.73E+01 ± 2.98E+01 +
IGSA	4.34E+01 ± 5.60E+00 −	**3.34E+01** ± **7.72E+00** −	3.79E-15 ± 2.08E-14 −	2.58E+03 ± 4.60E+02 +	1.84E+02 ± 7.44E+01 +
RGBSO	2.10E+02 ± 3.11E+01 +	1.08E+02 ± 1.89E+01 +	1.45E+03 ± 5.33E+02 +	3.22E+03 ± 5.39E+02 +	1.24E+02 ± 4.44E+01 +
MDBSO	1.74E+02 ± 6.95E+01 +	1.13E+02 ± 3.75E+01 +	4.09E+02 ± 5.38E+02 +	6.26E+03 ± 1.21E+03 +	1.15E+02 ± 5.06E+01 +
SCA	4.19E+02 ± 2.88E+01 +	2.47E+02 ± 1.63E+01 +	4.62E+03 ± 1.10E+03 +	7.12E+03 ± 3.34E+02 +	1.09E+03 ± 3.99E+02 +
DE	2.10E+02 ± 1.19E+01 +	1.78E+02 ± 1.31E+01 +	**0.00E+00** ± **0.00E+00** −	6.73E+03 ± 3.31E+02 +	4.79E+01 ± 2.31E+01 +
	CEC2017 D30 F11	CEC2017 D30 F12	CEC2017 D30 F13	CEC2017 D30 F14	CEC2017 D30 F15
CSE	4.38E+04 ± 2.69E+04	**5.93E+01** ± **2.88E+01**	**4.70E+01** ± **2.09E+01**	**2.33E+01** ± **1.57E+01**	**4.76E+02** ± **1.21E+02**
HGSA	1.28E+05 ± 8.15E+04 +	1.33E+04 ± 5.32E+03 +	5.32E+03 ± 3.05E+03 +	7.01E+02 ± 7.21E+02 +	1.23E+03 ± 2.32E+02 +
IGSA	1.40E+06 ± 7.32E+05 +	2.93E+04 ± 7.97E+03 +	1.94E+05 ± 1.37E+05 +	1.16E+04 ± 3.65E+03 +	1.11E+03 ± 2.16E+02 +
RGBSO	1.73E+06 ± 1.06E+06 +	1.63E+04 ± 1.66E+04 +	1.41E+04 ± 1.65E+04 +	3.07E+03 ± 5.42E+03 +	1.33E+03 ± 3.46E+02 +
MDBSO	4.27E+04 ± 2.26E+04 ≈	1.33E+04 ± 1.61E+04 +	6.31E+03 ± 5.29E+03 +	6.01E+03 ± 8.01E+03 +	8.38E+02 ± 4.43E+02 +
SCA	1.21E+09 ± 2.30E+08 +	4.07E+08 ± 1.98E+08 +	1.17E+05 ± 7.05E+04 +	1.56E+07 ± 1.29E+07 +	2.04E+03 ± 2.13E+02 +
DE	**7.46E+03** ± **7.34E+03** −	8.06E+01 ± 7.43E+00 +	6.25E+01 ± 4.78E+00 +	3.72E+01 ± 6.49E+00 +	7.21E+02 ± 4.28E+02 +
	CEC2017 D30 F16	CEC2017 D30 F17	CEC2017 D30 F18	CEC2017 D30 F19	CEC2017 D30 F20
CSE	9.04E+01 ± 6.29E+01	3.14E+03 ± 2.17E+03	**1.16E+01** ± **2.90E+00**	1.27E+02 ± 8.76E+01	**2.41E+02** ± **6.75E+00**
HGSA	1.07E+03 ± 1.99E+02 +	5.98E+04 ± 1.47E+04 +	3.52E+03 ± 1.25E+03 +	8.56E+02 ± 2.24E+02 +	3.10E+02 ± 5.90E+01 +
IGSA	5.18E+02 ± 2.14E+02 +	3.79E+05 ± 3.85E+05 +	1.38E+04 ± 8.10E+03 +	4.13E+02 ± 1.70E+02 +	2.54E+02 ± 6.54E+00 +
RGBSO	6.97E+02 ± 2.67E+02 +	1.42E+05 ± 8.64E+04 +	4.27E+03 ± 4.20E+03 +	5.84E+02 ± 2.44E+02 +	3.47E+02 ± 3.67E+01 +
MDBSO	2.77E+02 ± 1.94E+02 +	9.10E+04 ± 5.38E+04 +	6.35E+03 ± 8.22E+03 +	2.04E+02 ± 1.19E+02 +	2.98E+02 ± 4.16E+01 +
SCA	7.15E+02 ± 1.64E+02 +	2.80E+06 ± 1.21E+06 +	2.48E+07 ± 1.13E+07 +	6.12E+02 ± 1.29E+02 +	4.56E+02 ± 1.92E+01 +
DE	**8.02E+01** ± **1.22E+01** +	**3.59E+01** ± **4.40E+00** −	1.62E+01 ± 6.70E+00 +	**3.14E+01** ± **1.75E+01** −	3.68E+02 ± 1.05E+01 +
	CEC2017 D30 F21	CEC2017 D30 F22	CEC2017 D30 F23	CEC2017 D30 F24	CEC2017 D30 F25
CSE	3.29E+02 ± 7.03E+02	**3.89E+02** ± **8.95E+00**	4.80E+02 ± 1.27E+01	3.88E+02 ± 1.40E+00	1.27E+03 ± 4.59E+02
HGSA	1.00E+02 ± 3.91E-09 −	4.57E+02 ± 1.33E+02 +	5.23E+02 ± 3.58E+01 +	3.90E+02 ± 7.59E+00 ≈	2.53E+02 ± 5.07E+01 −
IGSA	**1.00E+02** ± **0.00E+00** −	4.38E+02 ± 2.26E+01 +	**4.19E+02** ± **2.24E+01** −	4.22E+02 ± 9.79E+00 +	**2.30E+02** ± **4.66E+01** −
RGBSO	1.34E+03 ± 1.82E+03 +	5.96E+02 ± 8.97E+01 +	9.16E+02 ± 1.27E+02 +	3.90E+02 ± 1.76E+01 ≈	2.87E+03 ± 1.89E+03 +
MDBSO	2.58E+03 ± 3.01E+03 +	4.30E+02 ± 2.30E+01 +	5.25E+02 ± 5.27E+01 +	3.93E+02 ± 1.20E+01 +	1.99E+03 ± 8.24E+02 +
SCA	6.05E+03 ± 2.37E+03 +	6.91E+02 ± 2.34E+01 +	7.56E+02 ± 2.96E+01 +	6.96E+02 ± 4.91E+01 +	4.27E+03 ± 2.56E+02 +
DE	**1.00E+02** ± **0.00E+00** −	5.18E+02 ± 1.31E+01 +	5.86E+02 ± 1.13E+01 +	**3.87E+02** ± **2.72E-02** −	2.49E+03 ± 1.41E+02 +
	CEC2017 D30 F26	CEC2017 D30 F27	CEC2017 D30 F28	CEC2017 D30 F29	w/t/l
CSE	5.08E+02 ± 5.89E+00	3.84E+02 ± 6.57E+01	**5.14E+02** ± **6.44E+01**	2.97E+03 ± 4.93E+02	
HGSA	5.51E+02 ± 2.08E+01 +	**3.07E+02** ± **2.82E+01** −	1.15E+03 ± 1.88E+02 +	7.99E+03 ± 2.60E+03 +	23/1/5
IGSA	6.75E+02 ± 6.87E+01 +	4.63E+02 ± 3.48E+01 +	1.13E+03 ± 2.22E+02 +	3.31E+05 ± 3.68E+05 +	22/0/7
RGBSO	5.89E+02 ± 3.33E+01 +	4.25E+02 ± 2.24E+01 +	1.00E+03 ± 2.39E+02 +	4.76E+04 ± 4.62E+04 +	28/1/0
MDBSO	5.35E+02 ± 2.01E+01 +	3.87E+02 ± 5.78E+01 ≈	7.96E+02 ± 1.92E+02 +	5.49E+03 ± 3.24E+03 +	25/4/0
SCA	6.94E+02 ± 4.83E+01 +	9.76E+02 ± 1.36E+02 +	1.72E+03 ± 2.62E+02 +	7.44E+07 ± 2.59E+07 +	29/0/0
DE	**4.89E+02** ± **9.90E+00** −	3.29E+02 ± 4.89E+01 −	5.45E+02 ± 1.11E+02 ≈	**2.01E+03** ± **5.69E+01** −	15/2/12

Table 2.9: Experiment data of CSE versus its peers on CEC2017 50 dimensions.

Algorithm	CEC2017 D50 F1	CEC2017 D50 F2	CEC2017 D50 F3	CEC2017 D50 F4	CEC2017 D50 F5
CSE	**2.07E-01 ± 3.60E-01**	**3.59E+03 ± 1.48E+03**	9.36E+01 ± 4.73E+01	8.99E+01 ± 1.17E+01	4.14E-02 ± 7.57E-02
HGSA	8.42E+02 ± 1.17E+03 +	1.18E+05 ± 1.15E+04 +	2.02E+02 ± 2.91E+01 +	2.68E+02 ± 1.36E+01 +	2.53E+01 ± 3.97E+00 +
IGSA	2.48E+03 ± 1.54E+03 +	1.24E+05 ± 1.16E+04 +	1.89E+02 ± 3.28E+01 +	**7.83E+01 ± 1.30E+01** −	4.22E-03 ± 6.93E-03 −
RGBSO	7.34E+03 ± 3.64E+03 +	2.61E+04 ± 1.35E+04 +	1.20E+02 ± 5.19E+01 +	2.61E+02 ± 5.03E+01 +	2.99E-01 ± 2.36E-01 +
MDBSO	5.79E+03 ± 5.73E+03 +	1.95E+04 ± 1.10E+04 +	1.00E+02 ± 5.67E+01 ≈	2.58E+02 ± 6.77E+01 +	2.75E+01 ± 8.43E+00 +
SCA	3.86E+10 ± 4.99E+09 +	1.04E+05 ± 1.44E+04 +	5.59E+03 ± 1.34E+03 +	5.50E+02 ± 2.92E+01 +	6.89E+01 ± 4.44E+00 +
DE	6.14E+01 ± 2.13E+02 +	1.02E+05 ± 1.49E+04 +	**6.18E+01 ± 4.45E+01** −	3.55E+02 ± 1.49E+01 +	**1.93E-08 ± 9.71E-08** −
	CEC2017 D50 F6	CEC2017 D50 F7	CEC2017 D50 F8	CEC2017 D50 F9	CEC2017 D50 F10
CSE	1.45E+02 ± 1.33E+01	9.04E+01 ± 1.27E+01	1.17E+01 ± 2.39E+01	**3.80E+03 ± 5.12E+02**	**5.45E+01 ± 8.95E+00**
HGSA	**6.99E+01 ± 3.44E+00** −	2.93E+02 ± 2.04E+01 +	**8.72E-14 ± 4.89E-14** −	5.83E+03 ± 5.55E+02 +	1.27E+02 ± 1.92E+01 +
IGSA	8.03E+01 ± 1.31E+01 −	**8.22E+01 ± 1.20E+01** −	2.56E-01 ± 9.86E-01 −	4.19E+03 ± 6.86E+02 ≈	6.25E+02 ± 1.91E+02 +
RGBSO	3.65E+02 ± 5.92E+01 +	2.80E+02 ± 4.76E+01 +	3.07E+03 ± 1.03E+03 +	5.79E+03 ± 7.41E+02 +	1.93E+02 ± 3.42E+01 +
MDBSO	5.05E+02 ± 2.29E+02 +	2.52E+02 ± 7.40E+01 +	2.97E+03 ± 2.35E+03 +	1.26E+04 ± 2.31E+03 +	1.78E+02 ± 6.46E+01 +
SCA	9.18E+02 ± 6.49E+01 +	5.59E+02 ± 2.08E+01 +	2.13E+04 ± 3.71E+03 +	1.32E+04 ± 4.05E+02 +	4.81E+03 ± 1.28E+03 +
DE	4.02E+02 ± 1.33E+01 +	3.56E+02 ± 1.65E+01 +	3.33E-02 ± 1.16E-01 −	1.31E+04 ± 4.41E+02 +	1.38E+02 ± 3.06E+01 +
	CEC2017 D50 F11	CEC2017 D50 F12	CEC2017 D50 F13	CEC2017 D50 F14	CEC2017 D50 F15
CSE	2.82E+05 ± 2.47E+05	3.85E+02 ± 4.19E+02	1.29E+03 ± 1.51E+03	**6.53E+01 ± 2.74E+01**	1.14E+03 ± 2.47E+02
HGSA	7.10E+05 ± 3.35E+05 +	6.04E+02 ± 5.38E+02 +	2.57E+04 ± 3.76E+04 +	7.11E+03 ± 1.64E+03 +	1.95E+03 ± 3.50E+02 +
IGSA	4.70E+06 ± 2.16E+06 +	4.00E+04 ± 1.71E+04 +	5.42E+05 ± 5.22E+05 +	1.07E+04 ± 4.88E+03 +	**1.09E+03 ± 3.59E+02** ≈
RGBSO	1.37E+07 ± 6.92E+06 +	2.32E+03 ± 2.79E+03 +	3.25E+04 ± 1.75E+04 +	5.56E+03 ± 5.80E+03 +	2.16E+03 ± 3.94E+02 +
MDBSO	6.42E+05 ± 4.39E+05 +	5.69E+03 ± 6.04E+03 +	3.91E+04 ± 4.07E+04 +	8.70E+03 ± 8.65E+03 +	1.64E+03 ± 7.99E+02 +
SCA	1.11E+10 ± 2.71E+09 +	2.64E+09 ± 8.77E+08 +	1.85E+06 ± 7.82E+05 +	3.41E+08 ± 1.43E+08 +	3.70E+03 ± 3.54E+02 +
DE	**4.86E+04 ± 2.36E+04** −	**3.26E+02 ± 1.95E+02** ≈	**1.29E+02 ± 8.01E+00** −	1.14E+02 ± 5.93E+00 +	2.74E+03 ± 5.07E+02 +
	CEC2017 D50 F16	CEC2017 D50 F17	CEC2017 D50 F18	CEC2017 D50 F19	CEC2017 D50 F20
CSE	**6.90E+02 ± 2.03E+02**	2.98E+04 ± 1.96E+04	**2.53E+01 ± 6.95E+00**	**5.44E+02 ± 1.54E+02**	2.97E+02 ± 1.34E+01
HGSA	1.74E+03 ± 3.14E+02 +	1.72E+05 ± 7.74E+04 +	1.50E+04 ± 3.36E+03 +	1.38E+03 ± 2.75E+02 +	4.62E+02 ± 3.22E+01 +
IGSA	1.26E+03 ± 2.60E+02 +	1.85E+06 ± 7.08E+05 +	3.50E+04 ± 1.48E+04 +	5.75E+02 ± 1.89E+02 ≈	**2.83E+02 ± 9.87E+00** −
RGBSO	1.68E+03 ± 4.22E+02 +	2.55E+05 ± 8.90E+04 +	1.42E+04 ± 6.80E+03 +	1.20E+03 ± 2.67E+02 +	5.14E+02 ± 4.57E+01 +
MDBSO	1.40E+03 ± 4.57E+02 +	1.07E+05 ± 8.22E+04 +	1.69E+04 ± 9.37E+03 +	1.19E+03 ± 5.97E+02 +	4.03E+02 ± 4.82E+01 +
SCA	2.59E+03 ± 2.79E+02 +	1.31E+07 ± 6.03E+06 +	2.16E+08 ± 1.20E+08 +	1.80E+03 ± 1.89E+02 +	7.56E+02 ± 2.84E+01 +
DE	1.48E+03 ± 4.08E+02 +	**7.04E+02 ± 5.64E+02** −	5.80E+01 ± 1.25E+01 +	1.14E+03 ± 4.68E+02 +	5.57E+02 ± 1.43E+01 +
	CEC2017 D50 F21	CEC2017 D50 F22	CEC2017 D50 F23	CEC2017 D50 F24	CEC2017 D50 F25
CSE	4.04E+03 ± 1.38E+03	**5.36E+02 ± 2.10E+01**	6.68E+02 ± 2.77E+01	5.24E+02 ± 3.65E+01	2.20E+03 ± 4.08E+02
HGSA	7.94E+03 ± 4.22E+02 +	9.98E+02 ± 1.80E+02 +	8.89E+02 ± 4.71E+01 +	5.82E+02 ± 1.89E+01 +	**3.00E+02 ± 6.91E-13** −
IGSA	**1.00E+02 ± 1.85E-13** −	5.78E+02 ± 3.96E+01 +	**5.51E+02 ± 2.74E+01** −	6.57E+02 ± 3.50E+01 +	3.02E+02 ± 9.98E+00 −
RGBSO	6.69E+03 ± 9.35E+02 +	9.12E+02 ± 9.75E+01 +	1.64E+03 ± 1.65E+02 +	**4.50E+02 ± 2.53E+01** −	6.23E+03 ± 1.66E+03 +
MDBSO	1.28E+04 ± 2.10E+03 +	6.39E+02 ± 6.64E+01 +	7.48E+02 ± 1.17E+02 +	5.44E+02 ± 4.71E+01 ≈	4.12E+03 ± 8.80E+02 +
SCA	1.37E+04 ± 4.06E+02 +	1.21E+03 ± 5.83E+01 +	1.25E+03 ± 5.05E+01 +	3.38E+03 ± 4.72E+02 +	9.11E+03 ± 6.65E+02 +
DE	4.55E+03 ± 6.41E+03 −	7.68E+02 ± 1.95E+01 +	8.25E+02 ± 4.85E+01 +	5.07E+02 ± 3.44E+01 −	4.02E+03 ± 6.85E+02 +
	CEC2017 D50 F26	CEC2017 D50 F27	CEC2017 D50 F28	CEC2017 D50 F29	w/t/l
CSE	5.69E+02 ± 2.86E+01	5.07E+02 ± 4.56E+00	**6.89E+02 ± 1.10E+02**	**6.04E+05 ± 2.32E+04**	
HGSA	1.32E+03 ± 3.01E+02 +	5.08E+02 ± 1.56E+01 +	1.80E+03 ± 3.13E+02 +	1.32E+06 ± 1.17E+05 +	26/0/3
IGSA	9.65E+02 ± 1.29E+02 +	6.21E+02 ± 8.62E+01 +	2.30E+03 ± 4.58E+02 +	3.53E+07 ± 8.77E+06 +	17/3/9
RGBSO	1.14E+03 ± 1.65E+02 +	4.86E+02 ± 1.26E+01 −	1.60E+03 ± 3.92E+02 +	1.18E+06 ± 1.40E+05 +	27/0/2
MDBSO	8.01E+02 ± 7.79E+01 +	5.19E+02 ± 9.69E+01 ≈	1.26E+03 ± 2.89E+02 +	1.01E+06 ± 2.35E+05 +	26/3/0
SCA	1.70E+03 ± 1.36E+02 +	3.54E+03 ± 4.84E+02 +	4.25E+03 ± 6.23E+02 +	6.05E+08 ± 1.73E+08 +	29/0/0
DE	**5.06E+02 ± 9.36E+00** −	**4.67E+02 ± 1.85E+01** −	7.83E+02 ± 4.70E+02 −	6.21E+05 ± 2.01E+04 +	17/1/11

Table 2.10: Experiment data of CSE versus its peers on CEC2011.

Algorithm	CEC2011 F1	CEC2011 F2	CEC2011 F3	CEC2011 F4	CEC2011 F5
CSE	**1.89E+00** ± **4.16E+00**	**-2.58E+01** ± **9.41E-01**	**1.15E-05** ± **2.13E-19**	2.14E+01 ± 3.87E-01	**-3.58E+01** ± **1.03E+00**
HGSA	1.49E+01 ± 4.92E+00 +	-2.37E+01 ± 2.64E+00 +	1.15E-05 ± 3.44E-12 +	1.60E+01 ± 1.93E+00 –	-3.22E+01 ± 2.47E+00 +
IGSA	2.68E+01 ± 2.92E+00 +	-2.15E+01 ± 2.03E+00 +	1.15E-05 ± 1.17E-18 +	1.97E+01 ± 2.48E+00 –	-3.04E+01 ± 2.63E+00 +
RGBSO	8.86E+00 ± 4.60E+00 +	-2.03E+01 ± 2.73E+00 +	1.15E-05 ± 3.58E-13 +	**1.52E+01** ± **1.39E+00** –	-3.36E+01 ± 1.19E+00 +
MDBSO	1.01E+01 ± 7.25E+00 +	-2.17E+01 ± 5.73E+00 ≈	1.15E-05 ± 1.21E-18 +	1.88E+01 ± 2.97E+00 –	-1.43E+01 ± 2.35E+00 +
	CEC2011 F6	CEC2011 F7	CEC2011 F8	CEC2011 F9	CEC2011 F10
CSE	**-2.92E+01** ± **4.20E-03**	1.12E+00 ± 1.26E-01	**2.20E+02** ± **0.00E+00**	**7.44E+02** ± **2.80E+02**	**-2.13E+01** ± **2.47E-01**
HGSA	-2.18E+01 ± 2.28E+00 +	**6.93E-01** ± **1.40E-01** –	2.21E+02 ± 2.79E+00 ≈	1.79E+05 ± 4.17E+04 +	-1.25E+01 ± 5.67E-01 +
IGSA	-2.08E+01 ± 2.49E+00 +	1.23E+00 ± 1.53E-01 +	9.79E+02 ± 4.82E+02 +	8.15E+05 ± 1.12E+05 +	-1.29E+01 ± 8.32E-01 +
RGBSO	-2.68E+01 ± 2.05E+00 +	6.96E-01 ± 1.58E-01 –	2.20E+02 ± 0.00E+00 ≈	1.31E+05 ± 1.12E+04 +	-1.53E+01 ± 1.80E+00 +
MDBSO	-2.24E+01 ± 3.05E+00 +	1.59E+00 ± 1.82E-01 +	2.20E+02 ± 1.83E+00 ≈	2.28E+03 ± 1.27E+03 +	-1.37E+01 ± 2.53E+00 +
	CEC2011 F11	CEC2011 F12	CEC2011 F13	CEC2011 F14	CEC2011 F15
CSE	5.15E+04 ± 5.34E+02	1.73E+07 ± 1.06E+04	**1.55E+04** ± **1.29E+01**	1.92E+04 ± 1.58E+02	3.30E+04 ± 6.24E+01
HGSA	5.13E+04 ± 5.31E+02 ≈	2.05E+07 ± 1.93E+05 +	4.48E+04 ± 3.54E+04 +	1.92E+04 ± 1.28E+02 ≈	3.33E+04 ± 2.21E+01 +
IGSA	**5.11E+04** ± **4.79E+02** –	2.72E+07 ± 1.15E+06 +	4.45E+04 ± 3.57E+04 +	1.92E+04 ± 1.50E+02 ≈	1.47E+05 ± 5.82E+04 +
RGBSO	5.27E+04 ± 4.70E+02 +	**5.00E+06** ± **2.88E+04** –	1.55E+04 ± 2.87E+01 +	**1.90E+04** ± **1.53E+02** –	3.32E+04 ± 5.65E+01 +
MDBSO	5.67E+04 ± 1.40E+04 +	1.75E+07 ± 9.45E+04 +	1.55E+04 ± 1.87E+01 +	1.92E+04 ± 1.52E+02 ≈	**3.29E+04** ± **7.25E+01** –
	CEC2011 F16	CEC2011 F17	CEC2011 F18	CEC2011 F19	CEC2011 F20
CSE	1.34E+05 ± 1.87E+03	**1.93E+06** ± **1.20E+04**	**9.40E+05** ± **1.52E+03**	**1.03E+06** ± **4.23E+04**	**9.40E+05** ± **1.86E+03**
HGSA	1.43E+05 ± 1.96E+03 +	1.94E+06 ± 5.57E+03 +	9.43E+05 ± 1.75E+03 +	1.15E+06 ± 7.85E+04 +	9.43E+05 ± 2.32E+03 +
IGSA	1.42E+05 ± 1.51E+03 +	1.94E+06 ± 5.46E+03 +	3.67E+06 ± 2.92E+06 +	6.26E+06 ± 4.52E+06 +	4.49E+06 ± 3.65E+06 +
RGBSO	7.41E+05 ± 2.19E+05 +	1.94E+06 ± 1.24E+04 +	9.44E+05 ± 2.50E+03 +	1.16E+06 ± 9.61E+04 +	9.45E+05 ± 2.72E+03 +
MDBSO	**1.33E+05** ± **2.55E+03** –	1.96E+06 ± 2.31E+04 +	1.01E+06 ± 6.90E+04 +	1.41E+06 ± 3.10E+05 +	1.08E+06 ± 1.82E+05 +
	CEC2011 F21	CEC2011 F22	w/t/l		
CSE	**1.52E+01** ± **2.27E+00**	**1.78E+01** ± **2.73E+00**			
HGSA	2.73E+01 ± 4.96E+00 +	3.66E+01 ± 5.93E+00 +	17/3/2		
IGSA	2.68E+01 ± 5.41E+00 +	3.30E+01 ± 4.06E+00 +	19/1/2		
RGBSO	2.01E+01 ± 2.89E+00 +	2.27E+01 ± 3.29E+00 +	17/1/4		
MDBSO	1.88E+01 ± 3.43E+00 +	2.51E+01 ± 2.93E+00 +	16/3/3		

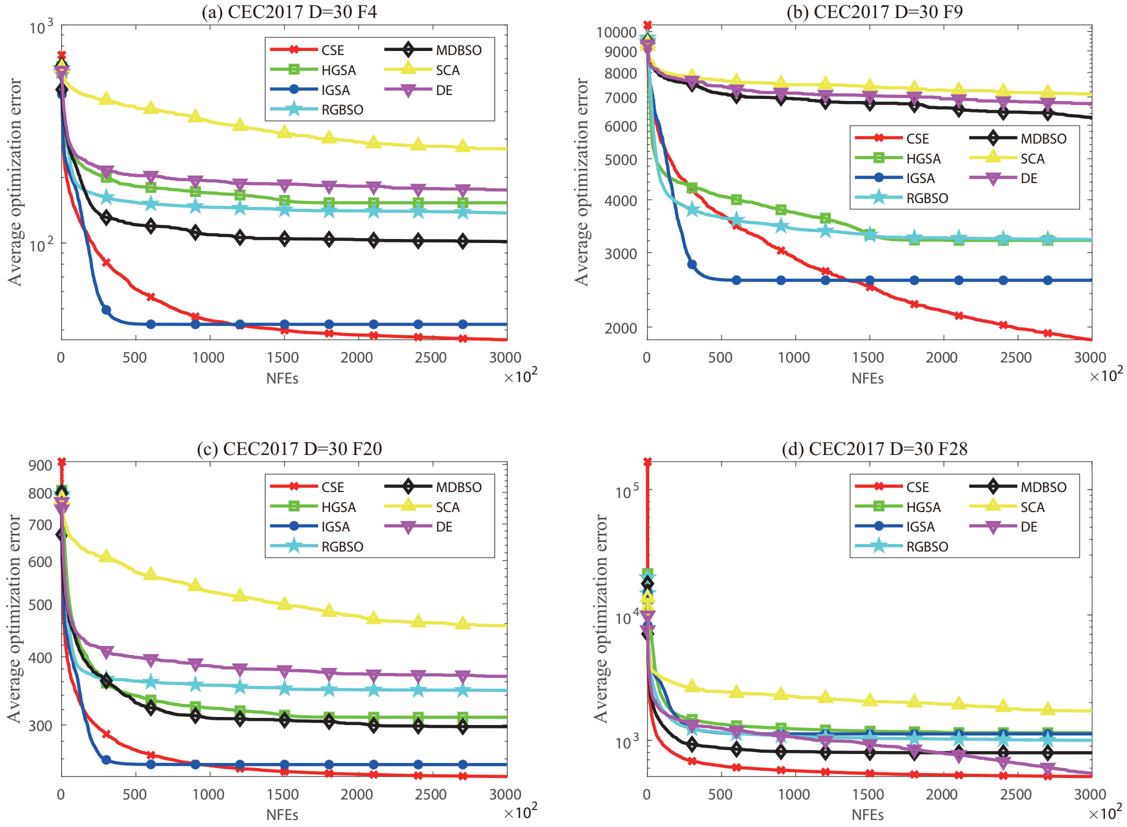
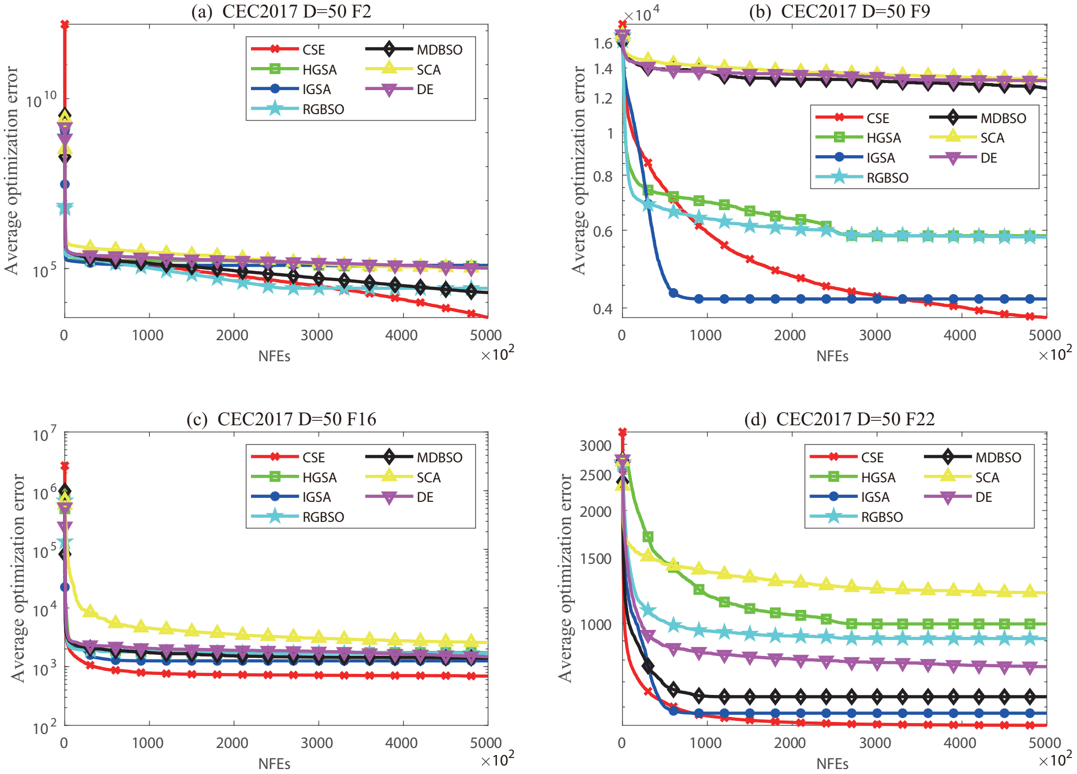


Figure 2.13: Convergence graphs of CSE versus its peers on CEC2017 30 dimensions.

strong convergence state when IGSA convergence stalls and is maintained until the process terminates. A similar situation is posted in F9 on 50 dimensions, which can be seen from Fig. 2.14. It can be seen that the CSE has an excellent ability to jump out of a local optimum. On other problems such as F28 on 30 dimensions, F2 and F16 on 50 dimensions, CSE converges earlier than other algorithms and achieved better solutions.


(a) CEC2017 D=50 F2

(b) CEC2017 D=50 F9

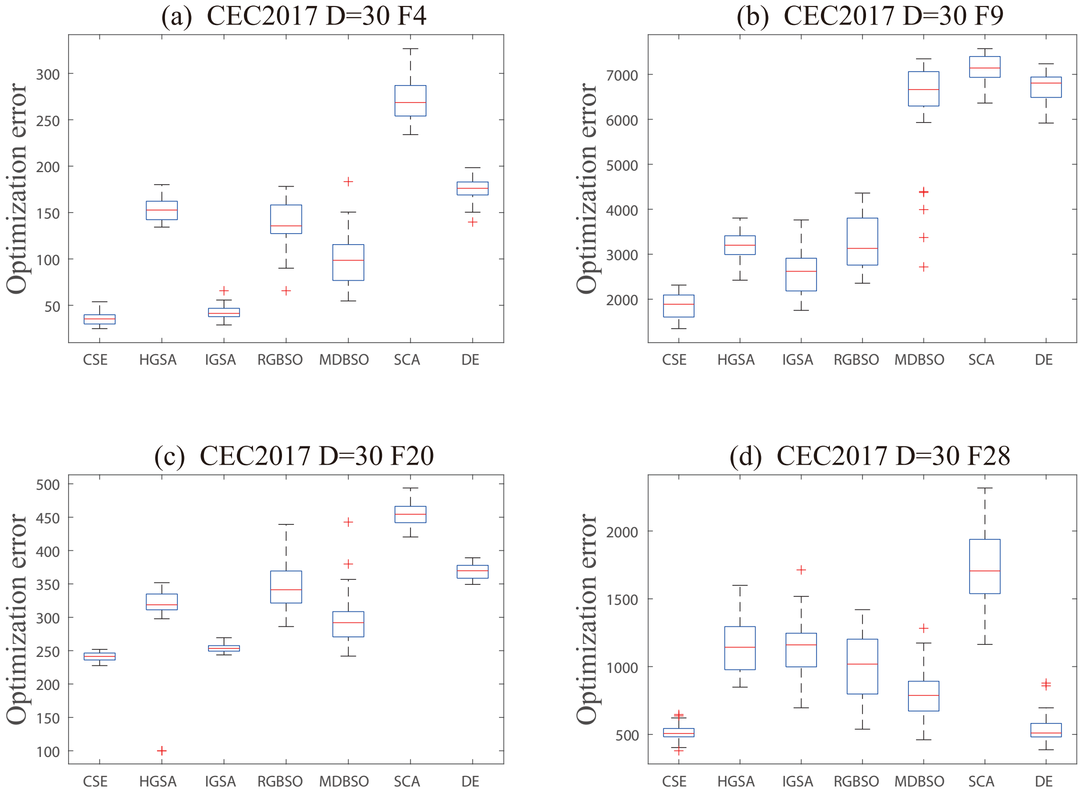
(c) CEC2017 D=50 F16

(d) CEC2017 D=50 F22

Figure 2.14: Convergence graphs of CSE versus its peers on CEC2017 50 dimensions.

The box-and-whisker diagrams used to observe the distribution and quality of the solutions are presented in Fig. 2.15 and Fig. 2.16. For most of the test problems in these two diagrams, CSE not only has higher solution quality, but also has excellent stability. One exception is that in F20 on 30 dimensions, HGSA achieved an extreme value, which is smaller than the minimum of all other algorithms. However, the extreme value does not represent the performance of the algorithm, and globally, the solution quality of CSE is generally higher than HGSA on F20.

With respect to CEC2011, the convergence status of the CSE and the quality and distribution of the solutions are depicted in Fig. 2.17 and Fig. 2.18, respectively. For F1,



(a) CEC2017 D=30 F4

(b) CEC2017 D=30 F9

(c) CEC2017 D=30 F20

(d) CEC2017 D=30 F28

Figure 2.15: Box-and-whisker diagrams of CSE versus its peers on CEC2017 30 dimensions.

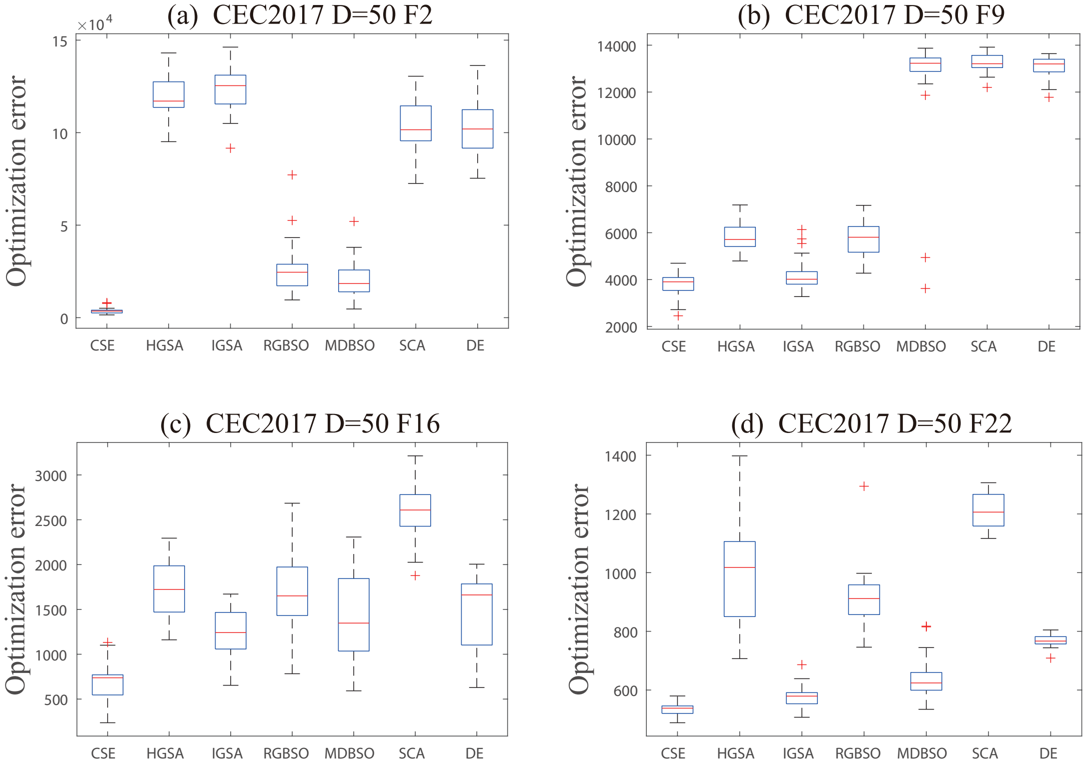


Figure 2.16: Box-and-whisker diagrams of CSE versus its peers on CEC2017 50 dimensions.

F21, and F22, the convergence performance of the CSE is better than that of several other algorithms. Moreover, the solution has a high quality and strong stability.

2.5 Conclusion

In this paper, we propose an adaptive CSE algorithm guided by the intensity of success based on historical information. The key of this paper is the application of chaotic local search. Chaotic sequences are used instead of standard random number generators to adjust the chaotic search radius. Several chaotic maps are used as alternatives to generate random values. The improvement in the algorithm caused by each chaotic sequence generating random values is referred to as success intensity. The success intensity caused by each chaotic map is recorded over several iterations. The higher the historical success intensity, the larger the area occupied by the roulette and the higher the probability of being selected. This means that the success information of several recent iterations will guide the choice of chaotic maps in the following iteration, thereby affecting the search range of the CLS. The change in search radius balances exploration and exploitation, avoiding both falling into local optimum and accelerating convergence.

To verify whether CSE is efficient, we conducted comparative experiments with SE and other metaheuristic algorithms. Prior to all experiments, we discussed the length of historical information. Among the results of the Friedman test, we found that the best results were obtained in this experiment when the length of the historical information was equal to MaxIter/300. The comparative tests are based on the benchmark function set CEC2017 and the real-world optimization problem set CEC2011. The results of the experiment with the SE prove that the CSE has a higher global convergence speed and better ability to address real-world problems. Additionally, the search trajectory graph of the CSE illustrates that it has the ability to jump out of the local optimum, which proves that our scheme of improvement is effective. Moreover, the comparison results with other algorithms also prove that the CSE is competitive.

In summary, the CSE is an effective algorithm, which also makes us more interested in CLS based on historical information. The historical information used in this paper only

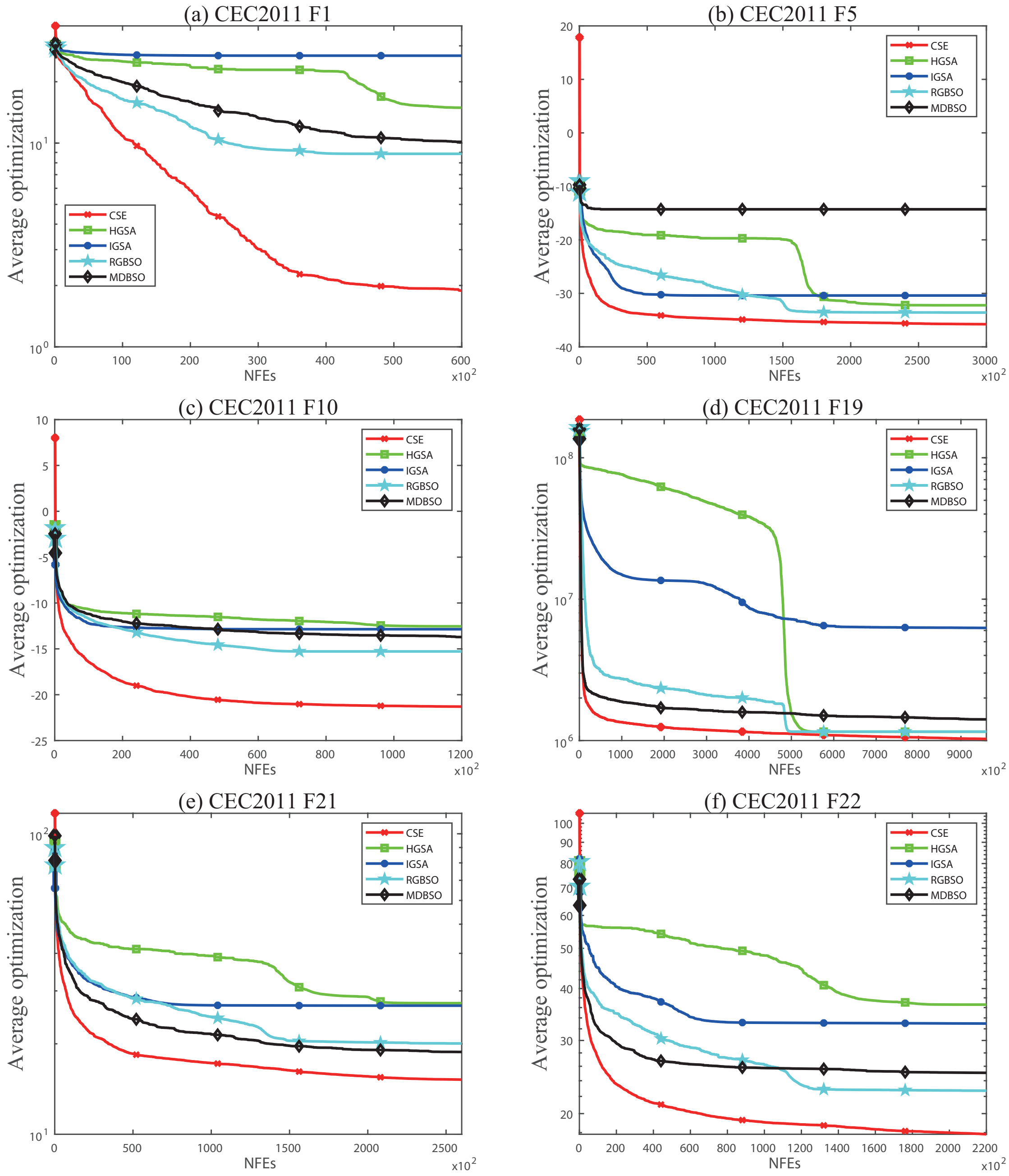


Figure 2.17: Convergence graphs of CSE versus its peers on CEC2011.

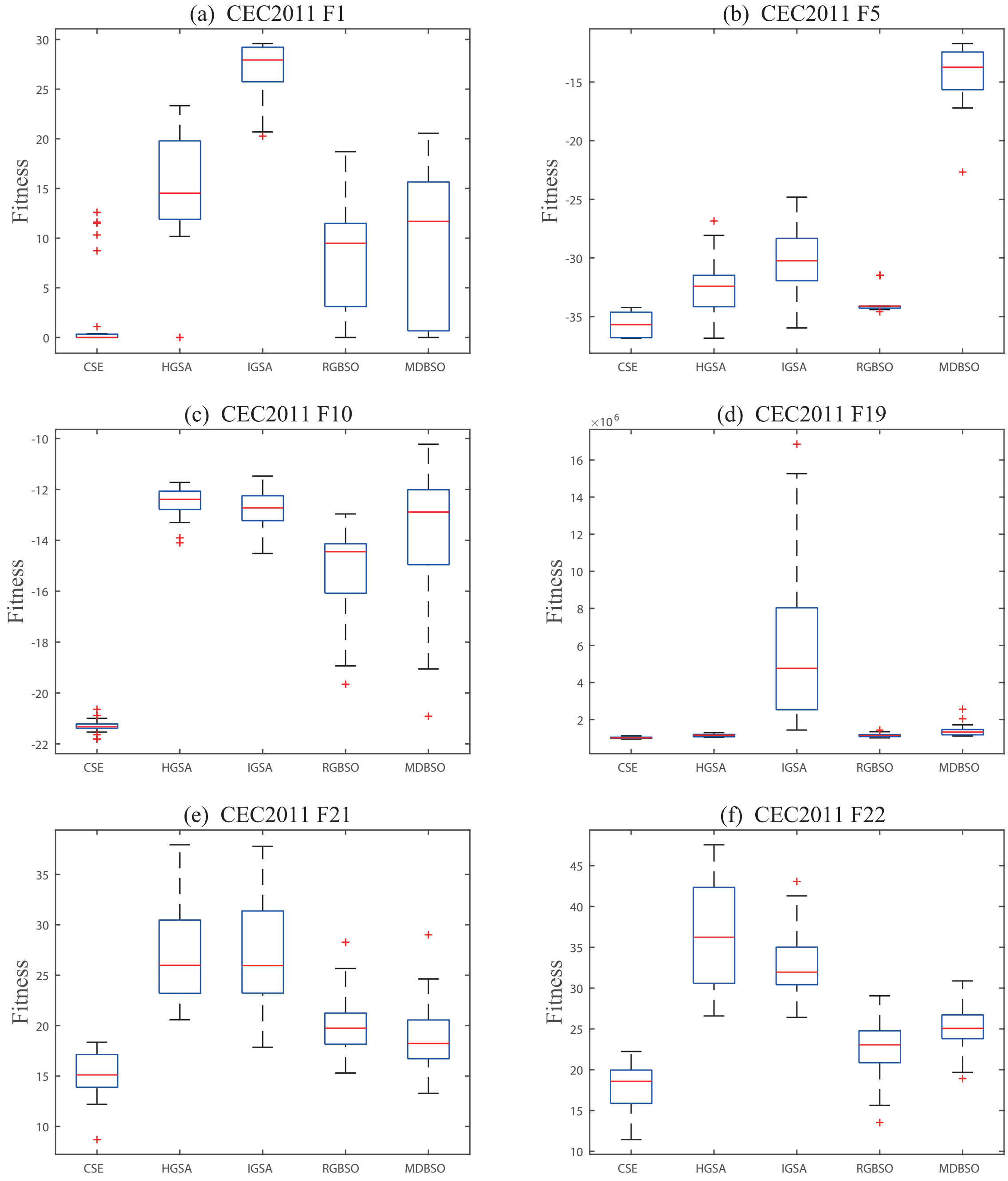


Figure 2.18: Box-and-whisker diagrams of CSE versus its peers on CEC2011.

considered the success intensity, both the intensity and the number of successes should be utilized simultaneously. Additionally, whether it can be integrated well with other global search algorithms, e.g., gravitational search algorithm [62, 63] will be the subject of future work. In addition, the applicability of CSE will be verified on other real-world applications, such as neural network learning [64], protein structure prediction [65], and Internet of vehicles [66] [67].

Chapter 3

Adaptive chaotic equilibrium optimizer

3.1 Introduction

Metaheuristic algorithms (MHAs) have received a lot of attention in the last decades because of their achievements in various fields [14, 16]. The inspiration determines the flow and structure of the algorithm. The genetic algorithm (GA) inspired by evolution and genetics uses crossover to exchange individual information [8]. The ant colony algorithm (ACO) inspired by the foraging behavior of ants uses pheromones to find the shortest path [9]. The gravitational search algorithm (GSA), which is inspired by the law of gravity, uses mass and interaction to update the position of particles and thus find the optimal solution [62, 63]. Brain storm optimization (BSO), which is inspired by human brainstorming behavior, clusters the population and updates the population using cluster centers and random individuals [11, 13, 60]. The particle swarm algorithm, which is inspired by the foraging behavior of bird flocks, gives each particle a direction and speed to find the optimal solution in space [68]. With various inspirations, MHAs have shown powerful performance in complex real-world problems [5], e.g., the lung cancer prognosis [69], no-wait flow shop scheduling problem (citeengin2018new, financial crisis prediction [70], optimal power flow problem [71], and classification of brain tumor images [72].

The major difference between MHAs and traditional algorithms is that the former can find the approximate solutions even if they do not find optimal solutions. For many engineering problems, this characteristics can save a lot of time cost. For MHAs, most of the research lies in improving the search ability of the algorithm, i.e., making the solution found closer to the true optimal solution. Search ability depends on the ability to exploit and explore the search space. Exploitation determines the ability to trap into the local region and improve the quality of the solution. Exploration determines the ability of the algorithm to jump out of the current region and find the region where the optimal solution is located. However, too much exploitation will make the algorithm lack the ability to jump out of the local optimum, while too much exploration will make convergence decrease or even stagnant [18]. Therefore, the balance of exploitation and exploration is crucial to the performance of the algorithm [73].

Equilibrium optimizer (EO) is proposed in 2020 with a novel inspiration, which is a mass balance model in a control volume [74]. Within this model, the variation of the input and generated masses in the volume with time is measured to estimate the equilibrium state. In EO, the particles represent the individuals and the concentrations are corresponding to the variables of the individuals. The particle update is composed of three separate terms, i.e., the equilibrium term, the exploration term and the exploitation term. These three terms have enabled EO to be one of the best-performing metaheuristic algorithms. Nevertheless, as a new algorithm, EO still have room for improvement. In this study, we attempt to improve the performance of EO by achieving a better balance between exploration and exploitation.

EO's equilibrium terms are selected from the equilibrium pool, which contains four best particles and an average particle. EO is expected to be enhanced by improving the quality of the equilibrium pool. A direct method is to improve the quality of the four best particles by a local search. Such combination of global and local search is usually termed as memetic algorithms (MA) [20]. MAs have advantages in solving complex and large-scale problems and have been applied in numerous fields, including the traveling salesman problem [75], quadratic assignment problem [76], and very large scale integrated-circuit floorplanning [77]. Within the framework of MA, the local search method needs to be determined.

Chaotic local search (CLS) is an effective strategy to improve the quality of solutions. Chaos has two properties of randomicity and ergodicity, which makes it unpredictable, but enabling MHAs to possess more ability to jump out of the local optima [78]. It is widely recognized that chaos is a good alternative to random numbers, and some researches have shown that in various cases chaos performs better than random numbers. In [24], the authors have shown that the convergence properties of the evolutionary algorithms (EAs) are strongly related to the random sequence, and it has been verified that the chaotic sequence replacing the random sequence can improve the performance of EAs. The method of replacing random numbers with chaos has also been used for the particle swarm optimization algorithm (PSO) [28], where all random variables in the PSO are replaced with chaotic numbers, and the results show that the competitiveness of the chaotic PSO is improved. Another chaotic PSO [29], which used multiple chaotic maps to adjust the attraction parameter, also improved the performance of the algorithm. On the other hand, CLS is a method that uses chaotic sequences to search for better individuals in the neighborhood of the current solution. It acts directly on the search space and is more efficient than just replacing random numbers. The ergodicity of chaos makes local search have a higher chance of finding better individuals. A chaotic artificial bee colony algorithm [43] proposes three chaotic strategies, which are replacing random numbers with chaos, the CLS, and a combination of these two. Experiments show that the latter two are more effective than the first one. In [37], the chaotic gravitational search algorithm also shows that the CLS improves the performance of the algorithm more than replacing random numbers. CLS is also used in BSO to avoid convergence stagnation [12]. A mechanism of utilizing multiple chaotic maps is proposed in [38], which leads to a huge performance improvement for differential evolution algorithms.

The above-mentioned research makes CLS the point of focus. From these previous related work, it can be found that CLS can generally enhance the local exploitation ability of a meta-heuristic algorithm. However, most of them only adopted one chaotic map to generate chaotic sequence, which obviously limits the adaptability and flexibility of the search. Some recent research [38] has shown that multiple chaotic maps can improve the search ability of an algorithm when encountering various landscapes of an optimization problem. The usage of multiple-chaos-based CLS thus motivates us to design a sophisticated incorporation method for the recently proposed equilibrium optimizer. Thus, in this paper, we propose an adaptive chaotic local search method based on the differential radius, and apply it on the equilibrium optimizer. The resultant algorithm is called chaotic equilibrium optimizer (CEO). The adaptability of the CEO comes from two factors, i.e., the differential radius and the selection of chaotic maps.

The differential radius denotes the difference between the four best particles in the equilibrium pool. When facing with complex problems, these four particles are often distributed in different regions with a large distance in early convergence stage, and a small one in late convergence stage. Therefore, this differential radius should also be decreased with iterations to better fine-tune solutions. By innovatively using the differential radius, the population diversity can be well maintained. In search search stage, the search radius is relatively large, the particles generated are far from the current region, thus preserves diversity for the population. As diversity is related to the performance of exploration and exploitation, it also provides the ability to jump out of the local optimum. Along with the iteration, the differential radius becomes smaller with the same magnitude of four particles in the equilibrium pool, thus enabling the search to perform as an exploitation manner.

The selection of the chaotic maps depends on a variable called success intensity. In this mechanism, several different chaotic maps are used to generate chaotic sequences, which are used to adjust the search radius. Since multiple chaotic maps are employed, selecting the appropriate chaotic map for the algorithm at the right time is the core task. The success intensity refers to the degree of improvement that a chaotic map brings to the algorithm. The success intensity of each chaotic map is recorded separately and accumulated for a period of time. The higher the cumulative success intensity, the larger the probability that the chaotic map will be selected for the next iteration. In comparison with previous CLS-based meta-heuristic algorithms, the distinct characteristics of CEO is the proposal of success intensity, which guides the algorithm to select the most promising chaotic map to generate sequences to perform the search. To verify the performance of CEO, extensive experiments based on a set of IEEE benchmark functions and four real-world applications are conducted. Comparative results with several mainstream meta-heuristics demonstrate the superiority of CEO in terms of solution quality and convergence speed.

In summary, this paper incorporates the properties of differential radius and multi-chaotic local search into the equilibrium optimizer, which makes the proposed CEO have better ability to balance global exploration with local exploitation. It contributes to the literature from the following aspects:

1) To our best knowledge, this is the first attempt to hybridize the chaotic local search with the equilibrium optimizer. Thus, an effective optimization algorithm is successfully designed to tackle the exploitation and exploration key problems in metaheuristic algorithms, which might give more insights into the mechanism of how to improve the search efficiency of an algorithm.

2) A novel success intensity-based multiple chaos embedding scheme is proposed in the local search operator, which significantly improving the local exploitation capacity of the algorithm, and meanwhile due to the inherent ergodicity of chaos, it also enables the algorithm to possess a strong ability of jumping out the local optimum.

3) Extensive experimental results based on 29 benchmark optimization functions and 4 different engineering problems verify the effectiveness and robustness of the proposed CEO algorithm in comparison with its peers. This solid results support the main conclusion of this paper that CEO is a powerful optimization algorithm that has great potential to deal with various real-world problems.

The remainder of this chapter is structured as follows: Sections 2 describes the origin and main processes of EO. Section 3 introduces the motivation of the CEO and the way it works. In section 4, CEO is compared with EO and other mainstream algorithms to illustrate its effectiveness and competitiveness. In section 5, the change of population diversity brought by the CEO and the computational complexity are discussed. Section 6 summarizes the chapter and presents the future work.

3.2 Equilibrium optimizer

EO is a population-based algorithm inspired by a well-mixed dynamic mass balance on a control volume. This dynamic balance describes an equation in which the change in mass over time within the system is equal to the input mass plus the generated mass minus the output mass, expressed as:

$$V\frac{\mathrm{d}C}{\mathrm{d}t} = H \cdot C^e - H \cdot C + X \qquad (3.1)$$

where V indicates the control volume, and C is the concentration inside it. Thus $V\frac{\mathrm{d}C}{\mathrm{d}t}$ describes the change rate in this control volume. H donates the flow rate input or output of the control volume. C^e is the concentration at an equilibrium state. In this equilibrium state, there is no generation in the volume. X represents the generation in this volume. Eq. (3.1) can be regarded as the function of $\frac{H}{V}$ solving for $\frac{\mathrm{d}C}{\mathrm{d}t}$, and $\frac{H}{V}$ is called a turnover rate. By dividing $\frac{H}{V}$ with ξ, Eq. (3.1) can be converted into an integration over time, shown as:

$$\int_{C^0}^{C} \frac{\mathrm{d}C}{\xi C^e - \xi C + \frac{X}{V}} = \int_{t^0}^{t} \mathrm{d}t \qquad (3.2)$$

Thus, Eq. (3.2) can be presented with a variable w as follows:

$$C = C^e + (C^0 - C^e)w + \frac{X}{\xi V}(1 - w) \qquad (3.3)$$

where w is expressed as:

$$w = e^{-\xi(t-t^0)} \qquad (3.4)$$

In Eqs. (3.2), (3.3) and (3.4), C^0 and t^0 mean the initial concentration and time, respectively. Based on the these equations, the framework of EO can be constructed in three steps, as shown in Fig. 3.1, and they are elaborated in the following.

3.2.1 Concentration initialization

Concentrations are the search agents of EO, and the initial concentrations are generated randomly in the given range. This process is expressed as follow:

$$C_{i,d}^0 = C_d^l + r_{i,d}^0 \cdot (C_d^u - C_d^l) \quad i = 1, ..., N \quad d = 1, ..., D \qquad (3.5)$$

where C_d^l and C_d^u are the lower and upper boundaries of the range set by the evaluation functions, respectively. The subscript i represents the ith particle in the population. N denotes the size of the population. D is the dimension of the objective functions. r^0 denotes

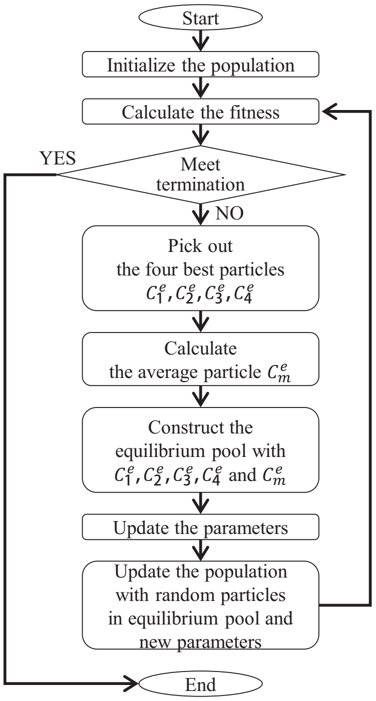

Start
Initialize the population
Calculate the fitness
Meet termination
YES
NO
Pick out the four best particles $C_1^e, C_2^e, C_3^e, C_4^e$
Calculate the average particle C_m^e
Construct the equilibrium pool with $C_1^e, C_2^e, C_3^e, C_4^e$ and C_m^e
Update the parameters
Update the population with random particles in equilibrium pool and new parameters
End

Figure 3.1: Flowchart of EO.

a random value between 0 and 1. This equation makes the particles randomly distributed in the search space.

3.2.2 Equilibrium pool and candidates

This step is a preparation for the population update. It is similar to the clustering strategy in BSO, where the update of the population depends on the cluster centers or random individuals. The population update of EO is based on the particles in the equilibrium pool. The equilibrium pool contains five particles. Four of them are the four best particles in the current iteration, and the other one is a particle composed of the mean of these four particles. The equilibrium pool can expressed as follow:

$$Z = \left\{ \overrightarrow{C_1^q}, \overrightarrow{C_2^q}, \overrightarrow{C_3^q}, \overrightarrow{C_4^q}, \overrightarrow{C_m^q} \right\} \tag{3.6}$$

where Z means the equilibrium pool, $\overrightarrow{C_1^q}$ to $\overrightarrow{C_4^q}$ denotes the best four particles, and $\overrightarrow{C_m^q}$ is the average particle. The particles in the equilibrium pool are randomly selected to generate the particles for the next iteration. It is stated that the best four particles provide the exploration ability for the algorithm, and the average particle provides the exploitation ability [74].

3.2.3 Concentration update

The particle set of EO represents the population of the algorithm, and its update method is reflected in the concentration update of each particle. The update equation is a variation of Eq. (3.3), shown as:

$$\begin{aligned} & C_{i,d}(k+1) = z_{i,d}(k) \\ & +(C_{i,d}(k) - z_{i,d}(k)) \cdot w_{i,d}(k) + \tfrac{X_{i,d}(k)}{\xi_{i,d}(k) \cdot V} \cdot (1 - w_{i,d}(k)) \\ & i = 1, 2, 3, ..., N \quad d = 1, 2, 3, ..., D \quad k = 1, 2, 3, ..., K \end{aligned} \tag{3.7}$$

where V is set to 1. ξ denotes the turnover rate, which is a value that changes with iteration time. Here, it is set to a random value between 0 and 1. z is a particle randomly selected from the equilibrium pool Z. k denotes the number of current iterations and K denotes the maximum number of iterations. w is used to help balance exploitation and exploration, and it is expressed as:

$$\overrightarrow{w} = e^{-\overrightarrow{\xi}(t-t^0)} \tag{3.8}$$

where t^0 and t are time-related parameters, respectively. They are set as functions of the number of iterations, shown as:

$$\overrightarrow{t^0} = \tfrac{1}{\overrightarrow{\xi}} \ln [-a \times S(\overrightarrow{r^1} - 0.5) \cdot (1 - e^{-\overrightarrow{\xi} t})] + t \tag{3.9}$$

$$t = (1 - \tfrac{k}{K})^{(b \frac{k}{K})} \tag{3.10}$$

where a and b are two constant values to adjust the exploration and exploitation ability, respectively. Regarding these parameters, more detailed analysis can be found in [74]. In addition, r^1 is a random value between 0 and 1. $S()$ represents the sign function, and its value depends on the positive or negative of the value in the parentheses. The parameter X in Eq. (3.7) denotes the generation rate which is important to perform the exploitation. It is defined as follow:

$$\overrightarrow{X} = \overrightarrow{X^0} e^{-\overrightarrow{\xi}(t-t^0)} \tag{3.11}$$

where X^0 represents the initial generation rate, shown as:

$$\begin{aligned} & X^0_{i,d} = F_{i,d} \cdot (z_{i,d} - \xi_{i,d} \cdot C_{i,d}) \\ & i = 1, 2, ...N \quad d = 1, 2, ..., D \end{aligned} \tag{3.12}$$

$$\overrightarrow{F} = \begin{cases} 0.5\overrightarrow{r^G} & r^P \geq P \\ \overrightarrow{0} & r^P < P \end{cases} \tag{3.13}$$

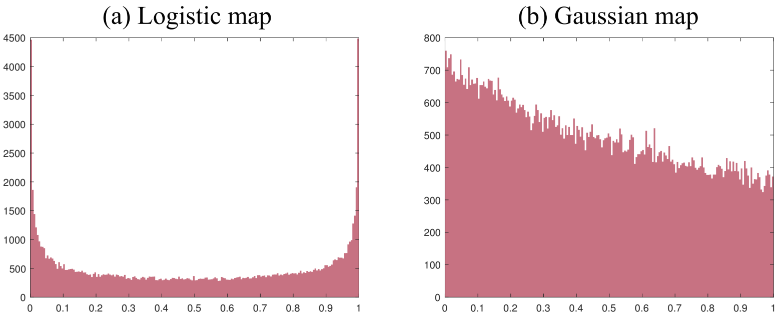
Here F is called a generation rate control parameter, and it determines the probability that the generation rate will take effect during the update. r^G and r^P are two random value between 0 to 1, respectively, and they affect the value of F. This process affects the number of particles that are updated using the generation rate.

3.3 Adaptive chaotic equilibrium optimizer (CEO)

3.3.1 Motivation

In the process of EO, the particles in the equilibrium pool are taking part in each particle update. Along with iterations, the particle's quality in the equilibrium pool is improved and the quality of the population is also gradually improved. It is reasonable to infer that the quality of the particles in the equilibrium pool is crucial for the population. Therefore, enhancing the search performance of the algorithm by improving the quality of the equilibrium pool is taken into account in this study.

To realize this, a direct approach is to perform a local search operation for the four best particles in each generation. When a better particle is found by the local search, it will replace the current particle, thus improving the overall quality of the equilibrium pool. This local search is expected to be adaptive and lead to a better balance between exploration and exploitation. First, a differential radius is considered, which can be combined with the characteristics of the EO itself. The distribution of the four best particles in the equilibrium


(a) Logistic map

(b) Gaussian map

Figure 3.2: Histogram of two typical chaotic maps.

pool in the complex problem is from divergence to convergence. To clearly illustrate this, Fig. 3.3 describes the convergence process of EO on the a benchmark function (F22) taken from CEC2017, where the red square indicates the location of the global optimum and the red dots indicate the four best particles of the current generation. The differential radius is the difference between two of these four particles at random, which also decreases with iteration.

As mentioned above, the random values used to adjust the radius are replaced by the values generated by chaotic maps, since some researches have shown the advantage of chaotic numbers over random numbers. Chaotic maps have different rules for generating values, which leads to their own characteristics for radius adjustment. As an example, Fig. 3.2 shows the distribution of values generated by the logistic map and the Gaussian map. It counts the distribution of 10,000 generated values by these two maps. From it, it can be observed that the logistic map produces more values near the boundary, which means that it is more likely to have a larger variation in the scale of the radius adjustment. On the other hand, the Gaussian map produces values with a decreasing frequency from 0 to 1, indicating that it is more possibly to have a small scale radius adjustment. Additionally, multi-chaos mechanism has made achievements, an adaptive selection method of chaotic maps can bring improvement to the algorithm. In this study, the selection method is designed to depend on the degree of contribution each chaotic map that makes to optimization. The variable that records this degree is named success intensity. The success intensity of each chaotic map is accumulated separately, and the higher the cumulative success intensity, the larger the probability of a chaotic map being selected. Furthermore, the chaotic maps

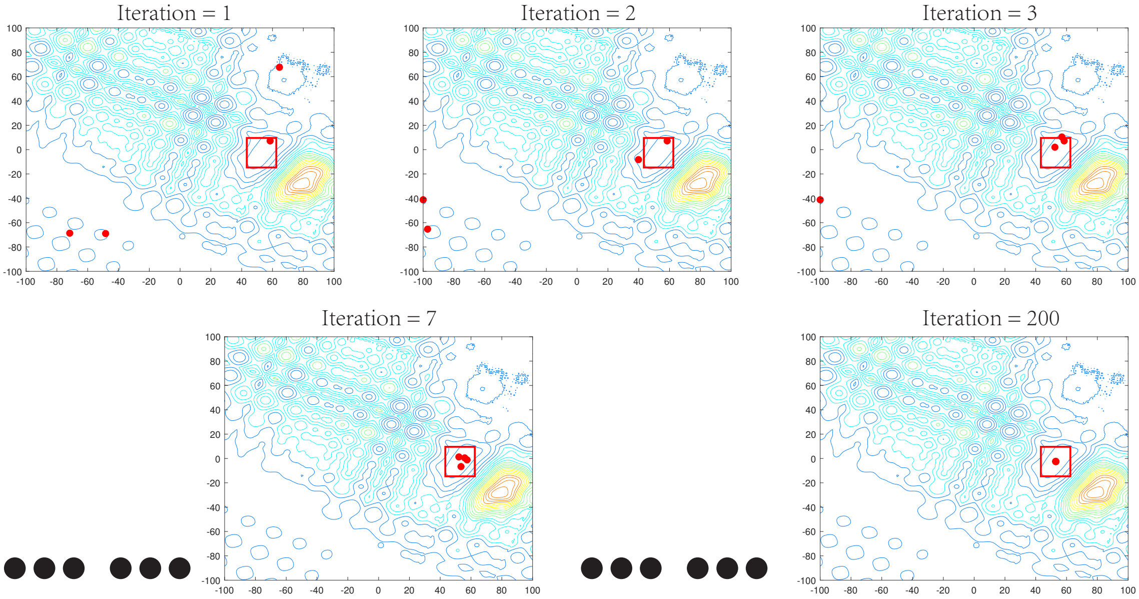


Figure 3.3: Diagram of the position change of the four best particles with iterations.

required by the algorithm at different convergence stages are different. Therefore, only the intensity in the latest several times of success is recorded and accumulated. This is also to avoid the impact of the huge convergence rate in the early stage on the later stage.

3.3.2 CLS in CEO

When it comes to local search, it is necessary to determine the radius of the search. CEO adopts an adaptive radius based on chaos and difference to enhance the exploration in the early stage and the exploitation in the later stage. It can be expressed as follows:

$$\begin{gathered}\overrightarrow{C_{g'}^{q,j}}(k) = \overrightarrow{C_g^q} + M_g^j(k)(\overrightarrow{C^{r1}}(k) - \overrightarrow{C^{r2}}(k)) \\ g = 1, 2, 3, 4, \quad j = 1, 2, 3, ..., J, \quad k = 1, 2, 3, ..., K\end{gathered} \tag{3.14}$$

where $\overrightarrow{C_g^q}$ denotes one particle in the equilibrium pool, $\overrightarrow{C_{g'}^{q,j}}$ is the particles temporarily generated by the CLS, which will replace the original one $\overrightarrow{C_g^q}$ if it is better than $\overrightarrow{C_g^q}$. $\overrightarrow{C^{r1}}$ and $\overrightarrow{C^{r2}}$ are two of the four best particles in the current iteration, and they are selected randomly. In this study, $\overrightarrow{C^{r1}} - \overrightarrow{C^{r2}}$ is regarded as the differential radius. $M_g^j(k)$ denotes a value generated by the jth chaotic map, and it is used to adjust the search radius. There are

J chaotic maps, and one of them is selected in each local search using a roulette wheel.

In roulette wheel selection, the percentage of each chaotic map depends on the magnitude of the improvement they bring to the algorithm. The greater the improvement, the bigger the percentage. It aims to pick the appropriate chaotic map to adjust the local search radius. This magnitude is referred to as the success intensity. It is the difference between the temporary particle and the current particle only when the local search finds a better particle. The success intensity is defined as:

$$\Delta_n^j = f(C_g^q(k)) - f(C_{g'}^{q,j}(k)) \tag{3.15}$$

where $f()$ is the objective function to calculate the fitness of the particle. Δ_n^j denotes the success intensity brought by the jth chaotic map at the nth successful local search. Δ_n^j is not recorded in every generation; it is executed only when a local search is performed successfully.

It is widely accepted that the operations currently perform well might also be a good choice for the remaining iterations. To maintain the long time influence of a selection result, the cumulative success intensity is used to calculate the roulette percentage. It is the sum of the success intensity brought by each chaotic maps in L times of successful local search. It is worth noting that most heuristic algorithms converge fast in the early stage and slow in the later stage. This results in an algorithm to require different values to adjust the search radius at different stages. It, thus, suggests that the adjustment of the radius must be timely. If the value of L is too large, a larger success intensity in the early stages is likely to affect the selection in the later stages. That is, the cumulative success intensity only records information of L times of local search before the current generation. Here L is set to 100 based on empirical testing. With the cumulative success intensity, The percentage of each chaotic map in roulette is expressed as follows:

$$\eta^j(k) = \begin{cases} \sum_{n=1}^{l} \Delta_n^j & l \leq L \\ \sum_{n=l-L+1}^{l} \Delta_n^j & l > L \end{cases} \tag{3.16}$$

Algorithm 4: Chaotic local search

```
Function CLS(l, λ,J):
    for $g = 1 : \lambda$ do
        Select two particles $C^{r1}$ and $C^{r2}$ randomly from the top 4 best particles
        Mark the top $g$ particle as $C_g$
        Generate an adjustment value $M$ by the roulette
        $C_{g'} \leftarrow C_g + M \cdot (C^{r1} - C^{r2})$
        if $C_{g'} < C_g$ then
            $C_g \leftarrow C_{g'}$;
            $l \leftarrow l + 1$
        Calculate cumulative success intensity $\eta$
        Update the success intensity percentage $p$ of chaotic maps
        Update $P$ by $1/J$
        Reset the roulette
```

$$p^j(k) = \frac{\eta^j(k)}{\sum_{j=1}^{J} \eta^j(k)} + \frac{1}{J} \tag{3.17}$$

$$P^{r,j}(k) = \frac{p^j(k)}{\sum_{j=1}^{J} p^j(k)} \tag{3.18}$$

In Eq. (3.16), $\eta^j(k)$ denotes the cumulative success intensity of the jth chaotic map in the latest L times of local search. l refers to the total number of successful local searches so far. p^j is the variable used to avoid unfairness, since the roulette may lead to certain chaotic maps that are never selected in the L times of local search. This will cause the cumulative success intensity of these chaotic maps to be recorded as 0. To give every chaotic map a chance of being selected, $1/J$ is introduced to keep the possibility of each chaotic map to be selected, which is shown in Eq. (3.17). J represents the number of chaotic maps. In this study, 12 widely chaotic maps are considered (i.e., $J = 12$), including the logistic map, piecewise linear chaotic map, singer map, sine map, sinusoidal map, tent map, bernoulli shift map, chebyshev map, circle map, cubic map, guassian map, and iterative chaotic map with infinite collapses. More details regarding the used 12 chaotic maps can be referred in [79]. $P^{r,j}$ in Eq. (3.18) is the percentage of a chaotic map for roulette in the actual algorithmic process.

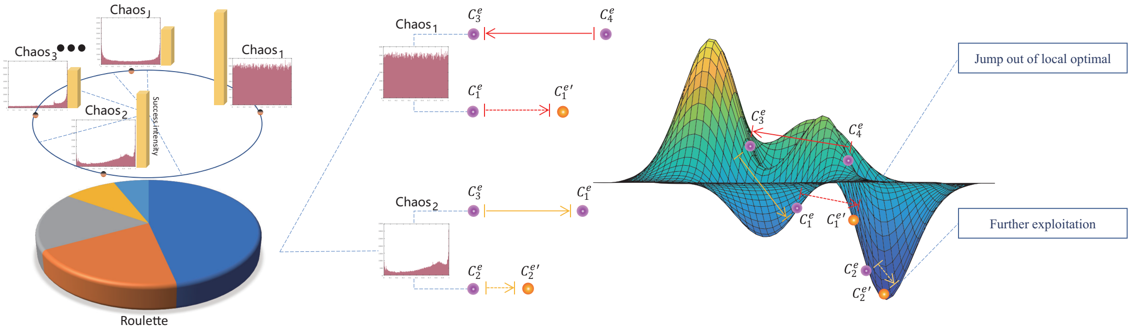


Figure 3.4: Descriptive process of CLS in CEO.

Algorithm 5: Chaotic equilibrium optimizer

```
Function CEO($N$, $FEs$):
    Initialize the population with N particles
    for $NFEs = 1 : FEs$ do
        Evaluate the population
        Construct the equilibrium pool with four best particles
        Perform $CLS$
        Update $w$ with Eqs. (3.8)~(3.10)
        Update generation rate $X$ with Eqs. (3.11)~(3.13)
        Update the population
```

Algorithm 4 shows the pseudo-code of the CLS in CEO. In it, λ indicates the number of CLS in each iteration. As the CLS acts on the four best particles in the equilibrium pool, the λ here equals to four. The pseudo-code of the proposed CEO is represented in Algorithm 5, where $NFEs$ denotes the number of current evaluations, FEs means the maximum number of function evaluations, and it is used as the termination condition of the algorithm.

Fig. 3.4 describes the operation of the CLS in the CEO, which expresses the two ideal cases of jumping out of the local optimal and further performing exploitation. On the left side of the diagram, $Chaos_1$ to $Chaos_J$ are the distribution histograms of these chaotic maps, which expresses the distribution of the values generated between 0 and 1. The yellow bars beside them represent their cumulative success intensity. The greater the cumulative success intensity, the larger the percentage in the roulette of being selected to perform CLS. C_1^e to C_4^e denote the four best particles in the equilibrium pool, respectively. $C_1^{e\prime}$ and $C_2^{e\prime}$ are two particles generated by the CLS. The search information of C_1^e comes from C_3^e and C_4^e, which makes the particle jump out of the local optimal. The search information of C_2^e is arisen from C_3^e and C_1^e, which makes the particle perform to perform further exploitation. As illustrated in Fig. 3.4, it can be expected that CEO can possess a better balance of exploration and exploitation of search.

3.4 Experiment

To verify the performance of CEO, the main experiment was divided into two phases. In the first phase, we compared CEO with EO on the benchmark function set CEC2017 [54]. CEC2017 contains 29 test functions, labeled as F1 to F29, where F1 and F2 are unimodal functions, F3 to F9 are simple multimodal functions, F10 to F19 are hybrid functions, and F20 to F29 are composition functions. Then two real-world optimization problems [55], i.e., the spread spectrum radar polly phase code design problem and the dynamic economic dispatch problem, were introduced to further verify the performance of the algorithm. In the second phase, CEO is compared with several recently proposed mainstream algorithms, including genetic learning particle swarm optimization (GLPSO) [51], grey wolf optimizer (GWO) [80], improved gravitational search algorithm (IGSA) [58], sine cosine algorithm (SCA) [61], and wingsuit flying search (WFS) [81]. Two additional real-world optimization problems, including the large scale transmission pricing problem and the spacecraft trajectory optimization problem are also tested [55].

3.4.1 Experiment setup

For the fairness of the experiment, all algorithms were set with the same maximum number of function evaluations as the termination condition. It was set to $D * 10^4$, where D represents the dimension of the test function. In this experiment, CEC2017 benchmark functions with 30 and 50 dimensions were used to evaluate the quality of the obtained solutions, respectively. The dimension of the real-world optimization problem varies from problem to problem, which will be clarified in the description of the experiment. To avoid randomness, all experiments were run 51 times independently and the mean value was evaluated as the final result.

Table 3.1: Friedman test for multi-chaotic maps discussion.

Algorithm	Rank	Final Rank
CEO_M	7.2759	1
CEO_1	7.6897	7
CEO_2	7.6552	6
CEO_3	7.8966	10
CEO_4	8.4828	14
CEO_5	7.6207	5
CEO_6	7.8276	8
CEO_7	7.8621	9
CEO_8	8.0345	11
CEO_9	7.3103	2
CEO_10	7.3793	3
CEO_11	8.069	12
CEO_12	8.4483	13
NoCEO	11.0345	15
RandEO	7.4138	4

3.4.2 Discussion of multi-chaos mechanism

Before the main experiment, a preliminary experiment was performed to ensure that the multi-chaos mechanism worked. The experiment used CEC2017 as the test function and the dimension was set to 30. The 12 chaotic maps used in the multi-chaos mechanism are used separately in the same local search mechanism. Table 3.1 shows the results of this experiment, where CEO_1 to CEO_12 represent the 12 single-chaos versions. In addition, the no-chaos mechanism version and the rand-based version are added to the comparison objects, labeled as NoCEO and RandEO, respectively. The Friedman test [56] is used to rank the performance of the algorithms. It can be found that the local search mechanism of multi-chaos outperforms other search mechanisms. In addition, NoCEO, which uses only difference without chaos for adjustment, has the least satisfactory search performance. These allow the competitiveness of multi-chaos CEO mechanism to be confirmed.

3.4.3 Comparison experiments with EO on benchmark functions

Since the improvement of CEO is based on EO, the most direct way to judge whether the solution is effective is to observe the quality of the solution of these two. The experiments

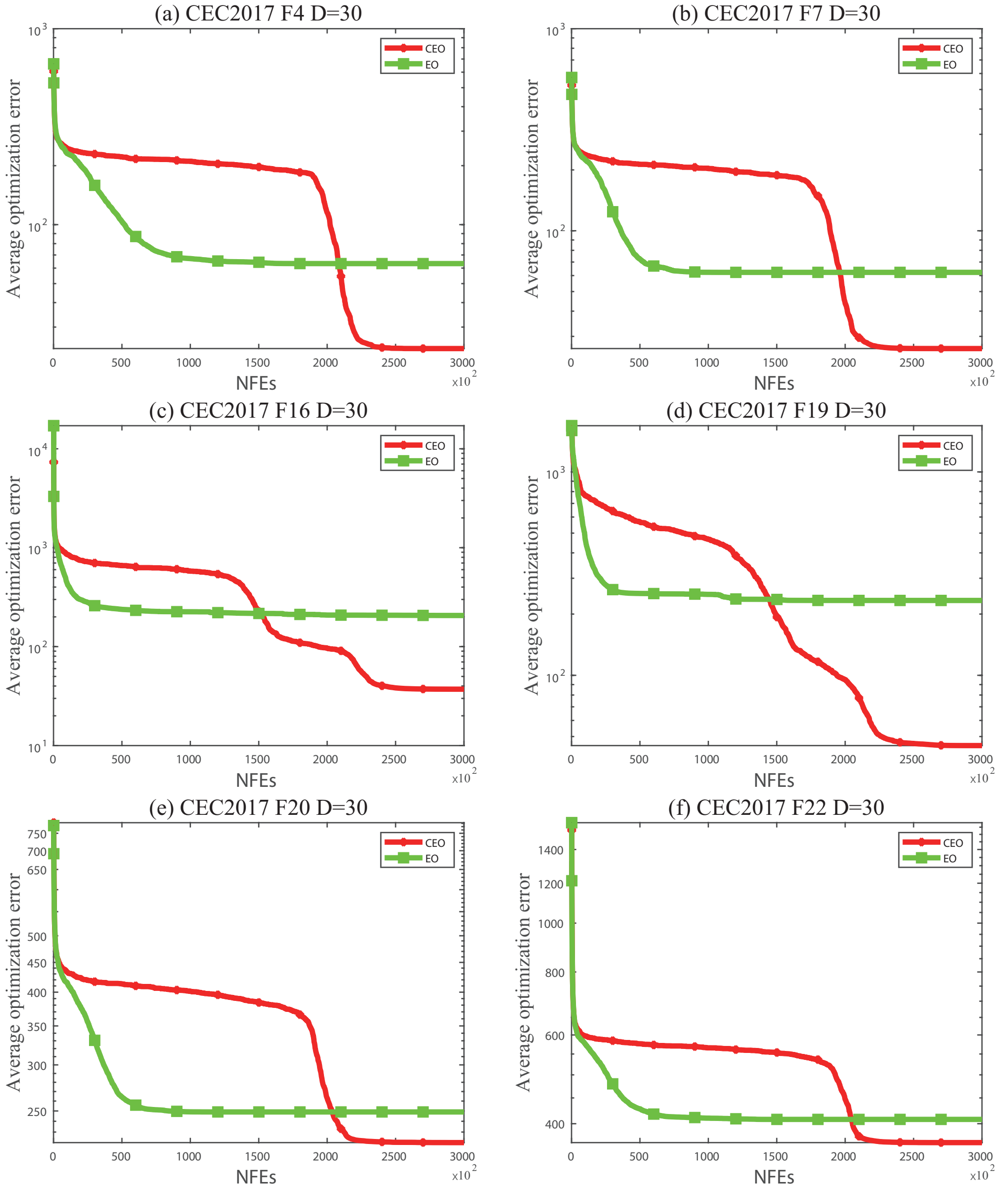
Table 3.2: Experiment data of CEO versus EO on CEC2017 benchmark functions with 30 dimensions.

Algorithm	CEC2017 D30 F1	CEC2017 D30 F2	CEC2017 D30 F3	CEC2017 D30 F4	CEC2017 D30 F5
CEO	6.05E+03 ± 5.59E+03	3.36E+04 ± 8.77E+03	8.72E+01 ± 3.94E-01	**2.33E+01** ± **6.93E+00**	4.34E-02 ± 7.97E-02
EO	4.94E+03 ± 4.91E+03 ≈	**4.71E+01** ± **6.13E+01** –	**7.87E+01** ± **2.59E+01** –	6.33E+01 ± 1.86E+01 +	**9.79E-03** ± **3.32E-02** –
	CEC2017 D30 F6	CEC2017 D30 F7	CEC2017 D30 F8	CEC2017 D30 F9	CEC2017 D30 F10
CEO	**5.30E+01** ± **5.61E+00**	**2.61E+01** ± **6.53E+00**	**2.85E-01** ± **3.15E-01**	**1.50E+03** ± **5.71E+02**	**3.66E+01** ± **2.86E+01**
EO	8.84E+01 ± 1.64E+01 +	6.22E+01 ± 1.48E+01 +	9.95E+00 ± 1.38E+01 +	3.40E+03 ± 6.28E+02 +	4.34E+01 ± 2.70E+01 +
	CEC2017 D30 F11	CEC2017 D30 F12	CEC2017 D30 F13	CEC2017 D30 F14	CEC2017 D30 F15
CEO	5.46E+05 ± 5.03E+05	4.54E+04 ± 2.28E+04	4.47E+04 ± 3.30E+04	3.49E+03 ± 4.43E+03	**5.25E+01** ± **9.51E+01**
EO	**7.43E+04** ± **5.96E+04** –	**1.97E+04** ± **1.80E+04** –	**7.95E+03** ± **7.00E+03** –	5.68E+03 ± 7.93E+03 ≈	6.03E+02 ± 2.86E+02 +
	CEC2017 D30 F16	CEC2017 D30 F17	CEC2017 D30 F18	CEC2017 D30 F19	CEC2017 D30 F20
CEO	**3.72E+01** ± **5.03E+01**	**1.09E+05** ± **5.75E+04**	7.02E+03 ± 1.15E+04	**4.51E+01** ± **4.02E+01**	**2.21E+02** ± **5.08E+00**
EO	2.06E+02 ± 1.26E+02 +	1.65E+05 ± 1.41E+05 +	1.03E+04 ± 1.55E+04 ≈	2.33E+02 ± 1.46E+02 +	2.49E+02 ± 1.27E+01 +
	CEC2017 D30 F21	CEC2017 D30 F22	CEC2017 D30 F23	CEC2017 D30 F24	CEC2017 D30 F25
CEO	1.00E+02 ± 6.93E-01	**3.67E+02** ± **8.13E+00**	**4.37E+02** ± **7.83E+00**	**3.87E+02** ± **5.23E-01**	**1.03E+03** ± **2.19E+02**
EO	1.38E+03 ± 1.67E+03 ≈	4.08E+02 ± 1.68E+01 +	4.78E+02 ± 1.81E+01 +	3.89E+02 ± 9.52E+00 +	1.48E+03 ± 3.61E+02 +
	CEC2017 D30 F26	CEC2017 D30 F27	CEC2017 D30 F28	CEC2017 D30 F29	w/t/l
CEO	**5.08E+02** ± **7.29E+00**	4.19E+02 ± 1.91E+01	**4.68E+02** ± **5.37E+01**	1.43E+04 ± 7.59E+03	
EO	5.16E+02 ± 8.21E+00 +	**3.36E+02** ± **4.90E+01** –	6.23E+02 ± 1.27E+02 +	**6.19E+03** ± **5.49E+03** –	17/4/8

Table 3.3: Experiment data of CEO versus EO on CEC2017 benchmark functions with 50 dimensions.

Algorithm	CEC2017 D50 F1	CEC2017 D50 F2	CEC2017 D50 F3	CEC2017 D50 F4	CEC2017 D50 F5
CEO	5.39E+03 ± 5.22E+03	9.44E+04 ± 1.54E+04	1.20E+02 ± 3.01E+01	**5.37E+01** ± **9.38E+00**	7.52E-02 ± 3.34E-02
EO	**2.82E+03** ± **4.70E+03** –	**3.06E+03** ± **2.71E+03** –	**6.86E+01** ± **4.61E+01** –	1.55E+02 ± 3.28E+01 +	**8.51E-02** ± **1.49E-01** –
	CEC2017 D50 F6	CEC2017 D50 F7	CEC2017 D50 F8	CEC2017 D50 F9	CEC2017 D50 F10
CEO	**1.05E+02** ± **8.64E+00**	**6.03E+01** ± **9.94E+00**	**1.55E+00** ± **1.29E+00**	**3.70E+03** ± **8.53E+02**	1.61E+02 ± 4.03E+01
EO	1.92E+02 ± 3.08E+01 +	1.58E+02 ± 3.85E+01 +	1.43E+02 ± 2.97E+02 +	6.07E+03 ± 7.96E+02 +	**1.03E+02** ± **3.52E+01** –
	CEC2017 D50 F11	CEC2017 D50 F12	CEC2017 D50 F13	CEC2017 D50 F14	CEC2017 D50 F15
CEO	3.42E+06 ± 2.02E+06	2.36E+04 ± 1.22E+04	7.84E+04 ± 6.63E+04	1.68E+04 ± 6.47E+03	**4.40E+02** ± **2.54E+02**
EO	**8.92E+05** ± **6.69E+05** –	**8.92E+03** ± **9.03E+03** –	**4.27E+04** ± **3.28E+04** –	**1.20E+04** ± **6.77E+03** –	1.37E+03 ± 4.55E+02 +
	CEC2017 D50 F16	CEC2017 D50 F17	CEC2017 D50 F18	CEC2017 D50 F19	CEC2017 D50 F20
CEO	**2.14E+02** ± **1.67E+02**	2.95E+05 ± 1.89E+05	1.87E+04 ± 1.19E+04	**9.05E+01** ± **1.16E+02**	**2.53E+02** ± **1.06E+01**
EO	9.95E+02 ± 3.04E+02 +	3.53E+05 ± 2.31E+05 ≈	1.98E+04 ± 1.29E+04 ≈	7.53E+02 ± 2.89E+02 +	3.23E+02 ± 2.58E+01 +
	CEC2017 D50 F21	CEC2017 D50 F22	CEC2017 D50 F23	CEC2017 D50 F24	CEC2017 D50 F25
CEO	**1.62E+03** ± **2.01E+03**	**4.60E+02** ± **1.31E+01**	**5.38E+02** ± **1.25E+01**	**5.23E+02** ± **3.55E+01**	**1.44E+03** ± **3.55E+02**
EO	6.41E+03 ± 1.36E+03 +	5.48E+02 ± 2.82E+01 +	6.12E+02 ± 2.91E+01 +	5.65E+02 ± 3.34E+01 +	2.56E+03 ± 5.76E+02 +
	CEC2017 D50 F26	CEC2017 D50 F27	CEC2017 D50 F28	CEC2017 D50 F29	w/t/l
CEO	**5.57E+02** ± **3.06E+01**	**4.67E+02** ± **1.75E+01**	**4.21E+02** ± **1.12E+02**	**9.08E+05** ± **2.11E+05**	
EO	6.25E+02 ± 5.07E+01 +	4.91E+02 ± 1.96E+01 +	8.86E+02 ± 2.44E+02 +	1.03E+06 ± 2.35E+05 +	18/2/9

were conducted using CEC2017 as the test function with 30 and 50 dimensions, respectively. The experimental results are shown in Table 3.2 and Table 3.3. The table records the mean and standard deviation of the results of 51 times of independent experiments. "+", "–" and "≈" indicate the statistical results of the Wilcoxon signed rank test. Significant wins are marked as "+", failures are marked as "–", and non-significant differences are marked as "≈". modColorFor ease of observation, the winning side is bolded in the table. The "$w/t/l$" at the end of the table shows that over the 29 test problems with 30 dimensions, the CEO achieved 17 wins, 4 ties, and 8 losses. Among the four problems that tied, the CEO achieved a lower mean on three of them. In addition, most of the problems that the CEO wins are multimodal functions, including simple multimodal functions, hybrid functions and composition functions. This proves that CEOs do have a better ability to bal-


(a) CEC2017 F4 D=30

(b) CEC2017 F7 D=30

(c) CEC2017 F16 D=30

(d) CEC2017 F19 D=30

(e) CEC2017 F20 D=30

(f) CEC2017 F22 D=30

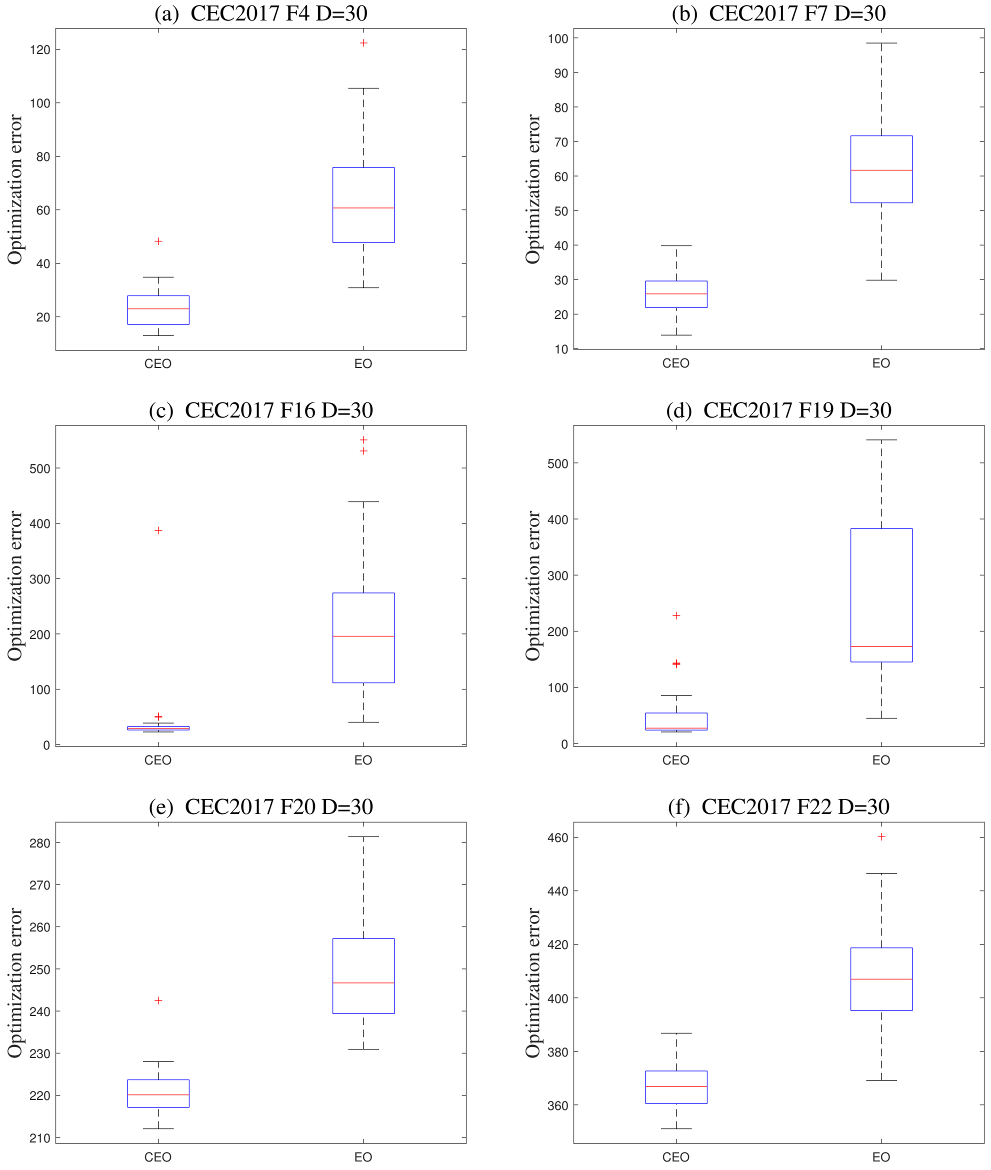
Figure 3.5: Convergence graphs of CEO versus EO on CEC2017 benchmark functions with 30 dimensions.

ance global exploration with local exploitation. Similarly on problems with 50 dimensions, CEO obtained 19 wins, 2 ties, and 9 losses. All these results suggest that CEO significantly outperform EO in terms of solution quality.

Observing the change process of the solution helps understand the properties of the algorithm. Fig. 3.5 shows several typical convergence graphs, which depict the CEO on simple multimodal functions (F4, F7), hybrid functions (F16, F19) and composition functions (F20, F22). The x-axis represents the number of function evaluations and the y-axis represents the average quality of the solution searched by the algorithm. It can be observed that EO possesses a higher convergence rate at the beginning, which is shown by the steep gradient before 50,000 evaluations and better solutions before 150,000 evaluations. But soon its convergence gradient becomes flat, which is possibly due to falling into a local optimum. While the CEO converges smoothly at the beginning, the steep increase in gradient during convergence makes it find better solutions at the later stage. Especially on F16, the CEO experienced two steep gradients. This situation is usually caused by the algorithm jumping out of the current local optimum. This is one of the reasons that CEO is considered to have a better balance between exploitation and exploration.

To observe the distribution of the obtained solutions, the box-and-whisker diagrams are shown in Fig. 3.6. The box-and-whisker diagram can express the overall distribution of solutions over the 51 times of experiment. In the figure, the red line indicates the median of the experimental results, and the top and bottom edges of the blue box indicate the first and third quartiles, respectively. The red + indicates that the extreme value deviates from the distribution. The black lines above and below the blue box represent the maximum and minimum values, respectively. The distance between the two black lines reflects the stability of the algorithm. The smaller the distance, the more stable it is. The position of the box-and-whisker plot expresses the strength of the algorithm. The lower the position, the better the solution it can find. Through Fig. 3.6, it can be found that CEO has a strong search ability ans robustness in the above-mentioned test functions.

In addition, the convergence graph (Fig. 3.7) and the box-and-whisker graph (Fig. 3.8) for 50 dimensions are also presented. It can be found that the convergence process of the CEO on the 50 dimensions is similar to that on the 30 dimensions, both showing a gentle early stage and an accelerated convergence in the later stage. Again, the stability of CEO is also verified.



(a) CEC2017 F4 D=30

(b) CEC2017 F7 D=30

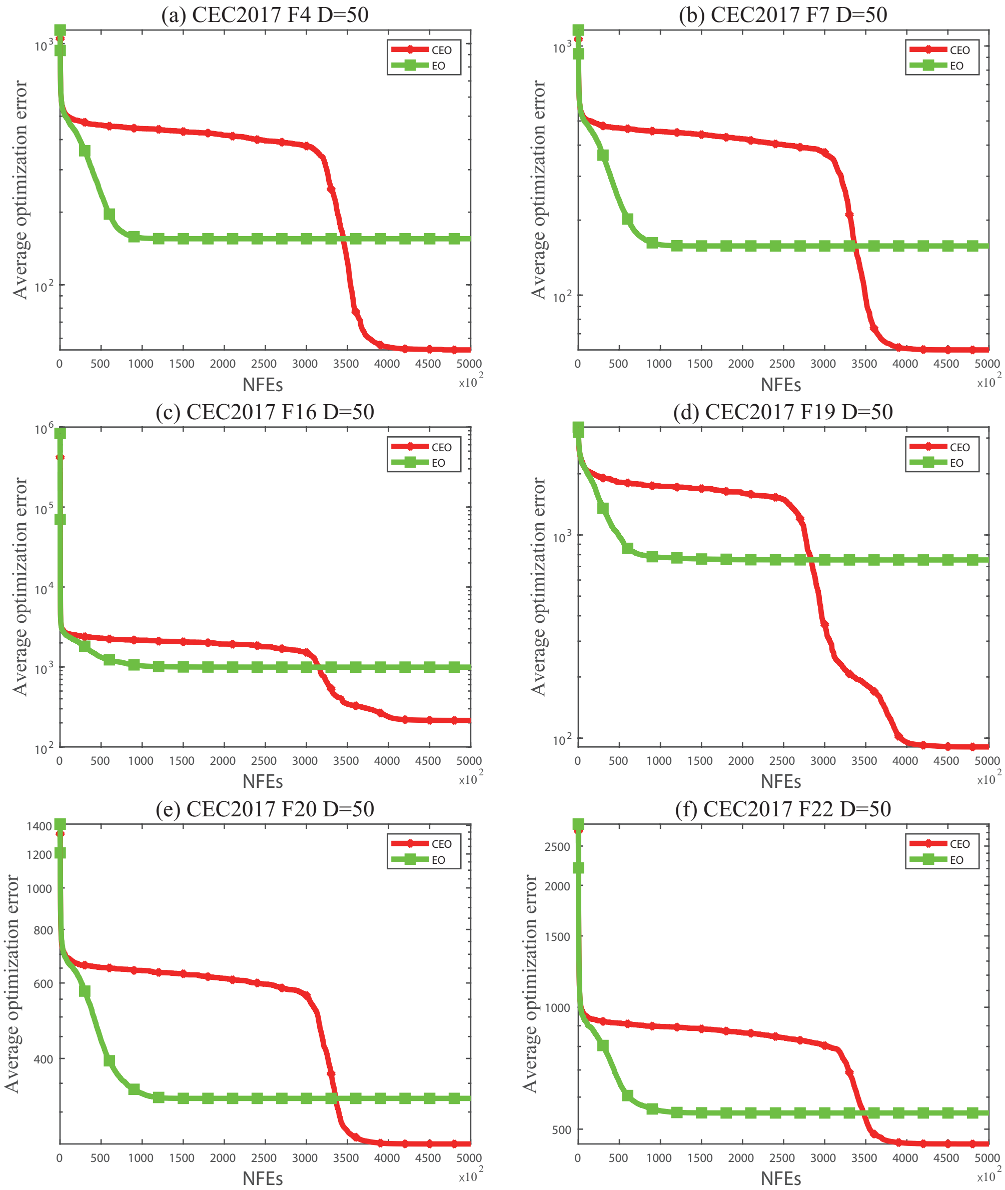
(c) CEC2017 F16 D=30

(d) CEC2017 F19 D=30

(e) CEC2017 F20 D=30

(f) CEC2017 F22 D=30

Figure 3.6: Box-and-whisker diagrams of CEO versus EO on CEC2017 benchmark functions with 30 dimensions.



(a) CEC2017 F4 D=50

(b) CEC2017 F7 D=50

(c) CEC2017 F16 D=50

(d) CEC2017 F19 D=50

(e) CEC2017 F20 D=50

(f) CEC2017 F22 D=50

Figure 3.7: Convergence graphs of CEO versus EO on CEC2017 benchmark functions with 50 dimensions.

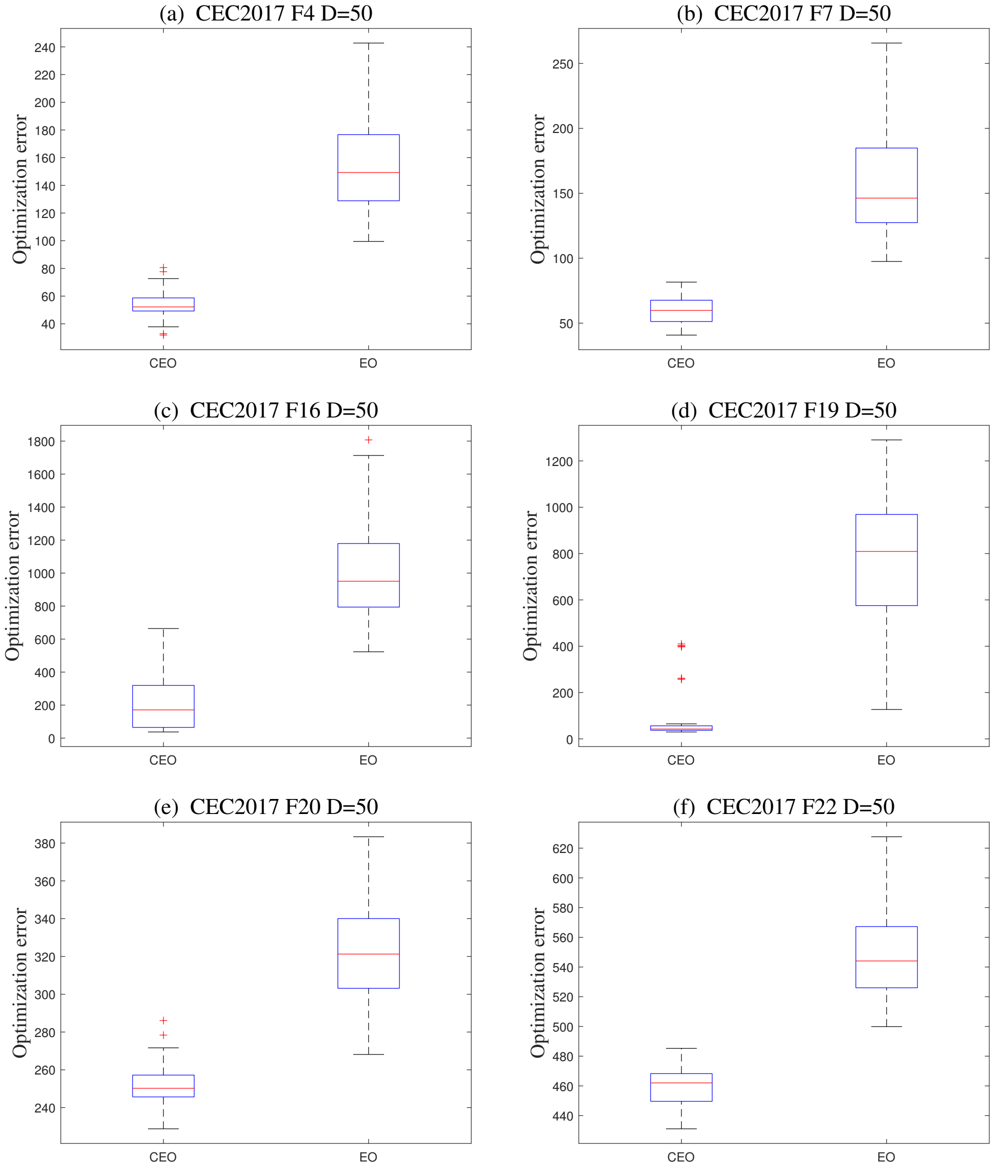


Figure 3.8: Box-and-whisker diagrams of CEO versus EO on CEC2017 benchmark functions with 50 dimensions.

Table 3.4: Experiment data of CEO versus EO on RPCDP.

Algorithm	RPCDP	Significance
CEO	**7.22E-01±1.93E-01**	
EO	1.01E+00±1.83E-01	+

3.4.4 Comparison experiments with EO on real-world optimization

In this section, two real-world optimization problems are used to further evaluate the performance of the CEO. The first problem, the radar polyphase code design problem (RPCDP) [55], arises from pulse compression techniques in radar systems. In a new method for polyphase pulse compression code synthesis, the problem is modeled as a minimum-maximum continuous optimization problem. This is an NP-hard problem, and its dimension is 20. Table 3.4 gives the results of the experiment in the form of "mean ± standard deviation" for 51 times of experiment. Fig. 3.9 gives the convergence graph and box-and-whisker diagram for the CEO and EO on this problem. It can be found that the CEO is slightly less stable than the EO, but the quality of the whole solution is better than that of EO, as shown by the third quartile. These experimental results show that CEO has better ability to solve low-dimensional problems than EO.

The second problem is the dynamic economic dispatch problem (DEDP) [55], which is part of a power system that needs to take into account the ever-changing hourly power demand and needs to consider multiple constraints. The experiment was executed on a 216-

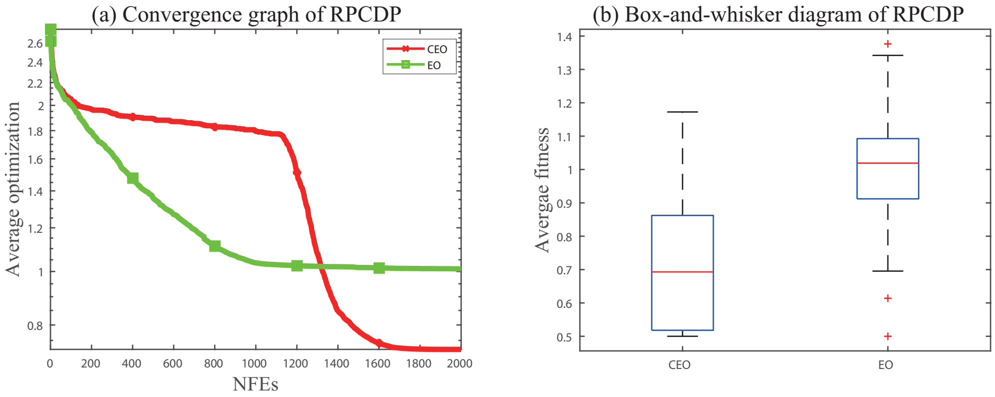


Figure 3.9: Convergence graph and box-and-whisker diagram of CEO versus EO for RPCDP.

Table 3.5: Experiment data of CEO versus EO on DEDP.

Algorithm	DEDP	Significance
CEO	**1.73E+07±6.43E+03**	
EO	1.75E+07±1.73E+05	+

dimensional instance to verify the performance of the CEO. Although, the advantage of the CEO's convergence curve is not so obvious from Fig. 3.10, it is undeniable that the CEO achieves a better solution, as can be seen from Table 3.5. The box-and-whisker plot also apparently shows that CEO is more stable. All these results show that CEOs are capable of achieving better solutions and are more stable when facing high-dimensional problems.

3.4.5 Comparison experiments with other algorithms on benchmark functions

The algorithms used for comparison include GLPSO, GWO, IGSA, SCA and WFS. To be specific, GLPSO hybridizes PSO with genetic evolution to enhance its overall performance. GWO is inspired by the leadership hierarchy and hunting behavior of the grey wolves to search the optimum. IGSA adopts a self-adaptive mechanism to adjust the gravitational parameter to maintain a balance between exploration and exploitation. SCA lets individuals fluctuate in the search space to find out the optimum with a mathematical model based on sine and cosine functions. WFS is new algorithm that simulates the extreme sport –

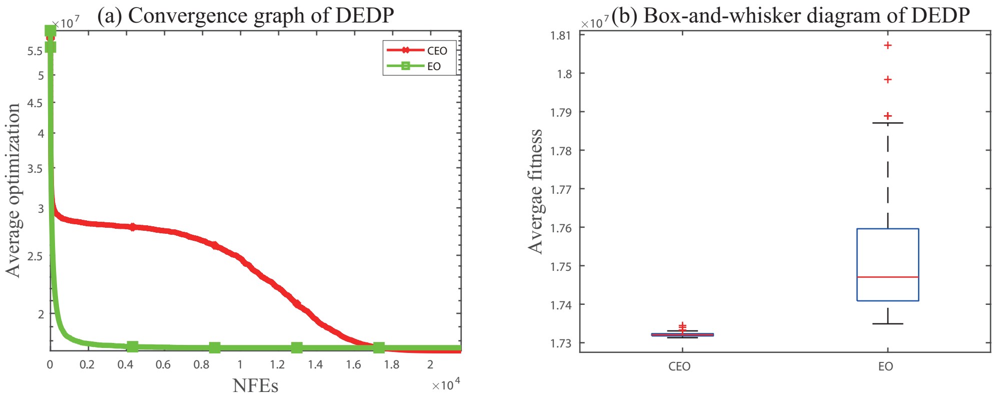


Figure 3.10: Convergence graph and box-and-whisker diagram of CEO versus EO for DEDP.

Table 3.6: Parameter settings of compared algorithms.

Algorithm	Parameters
GLPSO	ω=0.7298, c=1.49618, p_m=0.01, s_g=7
GWO	a linearly decrease from 2 to 0
IGSA	$\alpha_{mean}(0) = 20$, $\sigma = 0.3$, $p = 0.1$, $k = 6$
SCA	$a = 2$
WFS	$v > 0$

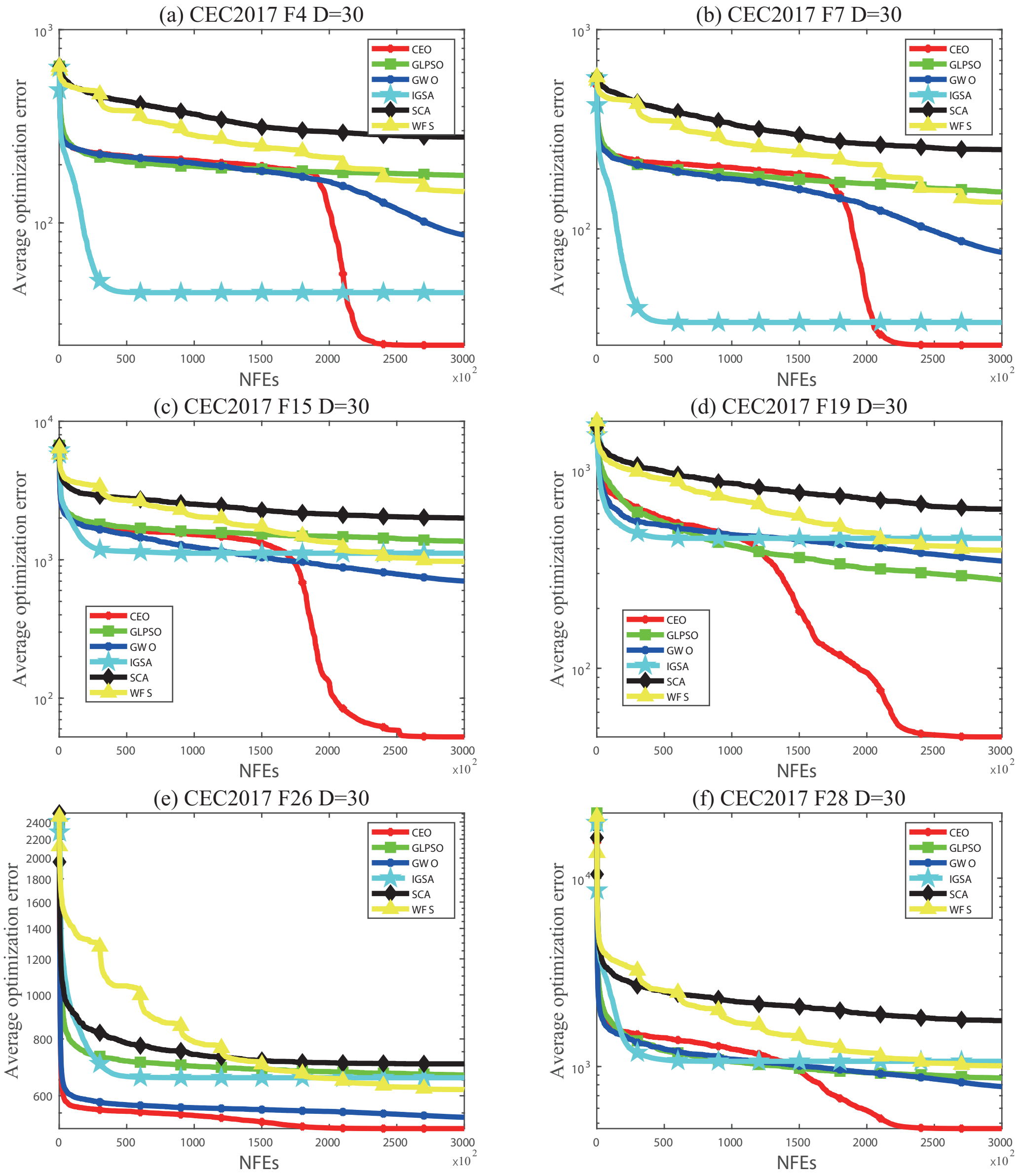
wingsuit flying to search for the optimum in space. Their parameter settings are listed in Table 3.6. The test function set is still CEC2017 with 30 and 50 dimensions.

Table 3.7 and Table 3.8 summarize the mean and standard deviation obtained by each algorithm. "w/t/l" in Table 3.7 shows that CEO gets the first place 19 times on 30 dimensions. CEO significantly outperforms its opponents on 24, 27, 21, 28 and 27 out of 29 benchmark functions, respectively. Table 3.8 exhibits that CEO achieves the first place 20 times on 50 dimensions. Also it is significantly better than its peers on 24, 28, 22, 29 and 28 out of 29 benchmark functions, respectively.

Fig. 3.11 shows the change process of the solutions of CEO and other algorithms on CEC2017 benchmark functions with 30 dimensions. It can be observed that on F4 and F5 (simple multimodal functions), IGSA has a significant advantage in the early stage of convergence, however, it is surpassed by CEO in the late stage. F15 and F19 are hybrid functions and F26 and F28 are composition functions, on these functions CEO still shows excellent search capabilities in the late stage. Fig. 3.12 depicts the box-and-whisker diagrams of CEO and other algorithms. From the length and position of the graphs, it can be observed that the quality of the solutions found by the CEO is better and more stable than that of its opponents. Fig. 3.13 and 3.14 are the convergence graphs and box-and-whisker diagrams on CEC2017 benchmark functions with 50 dimensions, respectively. The convergence trend and solution distribution of CEO are roughly the same as that on 30 dimensions, also showing better properties than its peers.

Table 3.7: Experiment data of CEO versus its peers on CEC2017 benchmark functions with 30 dimensions.

Algorithm	CEC2017 D30 F1	CEC2017 D30 F2	CEC2017 D30 F3	CEC2017 D30 F4	CEC2017 D30 F5
CEO	6.05E+03 ± 5.59E+03	3.36E+04 ± 8.77E+03	**8.72E+01 ± 3.94E-01**	**2.33E+01 ± 6.93E+00**	4.34E-02 ± 7.97E-02
GLPSO	**9.85E+04 ± 4.74E+05** −	**2.19E+04 ± 5.15E+03** −	2.91E+02 ± 9.24E+01 +	1.76E+02 ± 1.92E+01 +	5.09E+00 ± 2.06E+00 +
GWO	9.89E+08 ± 7.80E+08 +	2.99E+04 ± 1.05E+04 −	1.62E+02 ± 4.26E+01 +	8.67E+01 ± 3.32E+01 +	4.43E+00 ± 2.72E+00 +
IGSA	1.89E+03 ± 1.36E+03 −	5.99E+04 ± 7.47E+03 +	1.23E+02 ± 2.40E+01 +	4.36E+01 ± 9.54E+00 +	**3.52E-03 ± 1.21E-02** −
SCA	1.23E+10 ± 1.89E+09 +	3.52E+04 ± 6.48E+03 ≈	9.43E+02 ± 2.41E+02 +	2.78E+02 ± 2.06E+01 +	4.96E+01 ± 5.78E+00 +
WFS	7.06E+08 ± 3.32E+08 +	1.50E+04 ± 4.39E+03 −	2.45E+02 ± 5.22E+01 +	1.45E+02 ± 2.98E+01 +	2.55E+01 ± 5.93E+00 +
	CEC2017 D30 F6	CEC2017 D30 F7	CEC2017 D30 F8	CEC2017 D30 F9	CEC2017 D30 F10
CEO	5.30E+01 ± 5.61E+00	**2.61E+01 ± 6.53E+00**	2.85E-01 ± 3.15E-01	**1.50E+03 ± 5.71E+02**	**3.66E+01 ± 2.86E+01**
GLPSO	1.62E+02 ± 5.41E+01 +	1.53E+02 ± 3.82E+01 +	1.41E+01 ± 9.25E+00 +	6.54E+03 ± 3.35E+02 +	1.32E+02 ± 6.01E+01 +
GWO	1.32E+02 ± 3.03E+01 +	7.65E+01 ± 2.34E+01 +	4.95E+02 ± 3.89E+02 +	3.18E+03 ± 9.33E+02 +	4.12E+02 ± 3.93E+02 +
IGSA	**4.48E+01 ± 5.71E+00** −	3.40E+01 ± 6.78E+00 +	**6.69E-15 ± 2.70E-14** −	2.68E+03 ± 5.12E+02 +	1.84E+02 ± 7.61E+01 +
SCA	4.24E+02 ± 3.85E+01 +	2.50E+02 ± 1.86E+01 +	4.32E+03 ± 9.18E+02 +	7.22E+03 ± 2.89E+02 +	1.03E+03 ± 4.36E+02 +
WFS	2.34E+02 ± 3.75E+01 +	1.35E+02 ± 3.10E+01 +	1.77E+03 ± 1.13E+03 +	4.60E+03 ± 6.50E+02 +	3.21E+02 ± 6.94E+01 +
	CEC2017 D30 F11	CEC2017 D30 F12	CEC2017 D30 F13	CEC2017 D30 F14	CEC2017 D30 F15
CEO	**5.46E+05 ± 5.03E+05**	4.54E+04 ± 2.28E+04	4.47E+04 ± 3.30E+04	**3.49E+03 ± 4.43E+03**	**5.25E+01 ± 9.51E+01**
GLPSO	7.84E+06 ± 1.33E+07 +	5.50E+04 ± 2.30E+05 −	**3.53E+04 ± 8.10E+04** −	8.49E+03 ± 8.31E+03 +	1.36E+03 ± 2.05E+02 +
GWO	3.28E+07 ± 3.58E+07 +	3.74E+06 ± 1.92E+07 +	1.38E+05 ± 2.83E+05 ≈	1.99E+05 ± 4.71E+05 +	7.02E+02 ± 2.69E+02 +
IGSA	1.66E+06 ± 1.38E+06 +	**2.88E+04 ± 8.87E+03** −	2.51E+05 ± 1.50E+05 +	1.21E+04 ± 4.31E+03 +	1.11E+03 ± 2.45E+02 +
SCA	1.04E+09 ± 2.39E+08 +	4.17E+08 ± 1.85E+08 +	1.37E+05 ± 7.29E+04 +	1.20E+07 ± 9.82E+06 +	2.00E+03 ± 2.17E+02 +
WFS	9.80E+07 ± 7.88E+07 +	7.47E+05 ± 8.14E+05 +	7.67E+03 ± 7.82E+03 −	1.55E+05 ± 1.58E+05 +	9.72E+02 ± 2.50E+02 +
	CEC2017 D30 F16	CEC2017 D30 F17	CEC2017 D30 F18	CEC2017 D30 F19	CEC2017 D30 F20
CEO	**3.72E+01 ± 5.03E+01**	**1.09E+05 ± 5.75E+04**	**7.02E+03 ± 1.15E+04**	**4.51E+01 ± 4.02E+01**	**2.21E+02 ± 5.08E+00**
GLPSO	2.78E+02 ± 1.65E+02 +	6.95E+05 ± 7.50E+05 +	9.55E+03 ± 1.39E+04 ≈	2.79E+02 ± 1.38E+02 +	3.74E+02 ± 2.34E+01 +
GWO	2.53E+02 ± 1.17E+02 +	5.46E+05 ± 7.05E+05 +	6.00E+05 ± 6.49E+05 +	3.47E+02 ± 1.27E+02 +	2.76E+02 ± 2.02E+01 +
IGSA	5.36E+02 ± 1.61E+02 +	3.33E+05 ± 2.20E+05 +	1.45E+04 ± 1.08E+04 +	4.51E+02 ± 1.69E+02 +	2.54E+02 ± 6.84E+00 +
SCA	6.99E+02 ± 1.65E+02 +	3.33E+06 ± 1.54E+06 +	2.47E+07 ± 1.33E+07 +	6.32E+02 ± 1.28E+02 +	4.56E+02 ± 1.68E+01 +
WFS	3.22E+02 ± 1.07E+02 +	2.16E+05 ± 1.48E+05 +	1.10E+06 ± 1.13E+06 +	3.92E+02 ± 1.08E+02 +	3.33E+02 ± 2.85E+01 +
	CEC2017 D30 F21	CEC2017 D30 F22	CEC2017 D30 F23	CEC2017 D30 F24	CEC2017 D30 F25
CEO	1.00E+02 ± 6.93E-01	**3.67E+02 ± 8.13E+00**	4.37E+02 ± 7.83E+00	**3.87E+02 ± 5.23E-01**	1.03E+03 ± 2.19E+02
GLPSO	1.02E+02 ± 2.32E+00 +	5.93E+02 ± 2.10E+01 +	6.56E+02 ± 2.18E+01 +	4.33E+02 ± 2.13E+01 +	2.94E+03 ± 9.36E+02 +
GWO	1.45E+03 ± 1.45E+03 +	4.38E+02 ± 2.75E+01 +	5.16E+02 ± 5.09E+01 +	4.52E+02 ± 2.27E+01 +	1.82E+03 ± 3.45E+02 +
IGSA	**1.00E+02 ± 1.00E-13** −	4.40E+02 ± 2.10E+01 +	**4.22E+02 ± 2.24E+01** −	4.20E+02 ± 8.29E+00 +	**2.37E+02 ± 4.88E+01** −
SCA	5.88E+03 ± 2.51E+03 +	6.84E+02 ± 2.39E+01 +	7.64E+02 ± 2.41E+01 +	7.05E+02 ± 7.41E+01 +	4.33E+03 ± 3.17E+02 +
WFS	3.03E+02 ± 8.44E+01 +	5.31E+02 ± 3.99E+01 +	5.79E+02 ± 3.66E+01 +	5.23E+02 ± 3.62E+01 +	2.54E+03 ± 7.02E+02 +
	CEC2017 D30 F26	CEC2017 D30 F27	CEC2017 D30 F28	CEC2017 D30 F29	w/t/l
CEO	**5.08E+02 ± 7.29E+00**	**4.19E+02 ± 1.91E+01**	**4.68E+02 ± 5.37E+01**	**1.43E+04 ± 7.59E+03**	
GLPSO	6.67E+02 ± 2.15E+01 +	5.46E+02 ± 7.72E+01 +	8.68E+02 ± 1.78E+02 +	9.17E+04 ± 1.54E+05 +	24/1/4
GWO	5.38E+02 ± 1.81E+01 +	5.59E+02 ± 5.61E+01 +	7.82E+02 ± 1.53E+02 +	5.65E+06 ± 5.05E+06 +	27/1/1
IGSA	6.57E+02 ± 5.12E+01 +	4.55E+02 ± 2.87E+01 +	1.07E+03 ± 1.90E+02 +	3.15E+05 ± 2.56E+05 +	21/0/8
SCA	7.05E+02 ± 4.12E+01 +	1.02E+03 ± 1.15E+02 +	1.75E+03 ± 2.52E+02 +	6.67E+07 ± 2.42E+07 +	28/1/0
WFS	6.19E+02 ± 3.09E+01 +	6.33E+02 ± 7.87E+01 +	1.01E+03 ± 1.44E+02 +	6.90E+06 ± 5.63E+06 +	27/0/2



(a) CEC2017 F4 D=30

(b) CEC2017 F7 D=30

(c) CEC2017 F15 D=30

(d) CEC2017 F19 D=30

(e) CEC2017 F26 D=30

(f) CEC2017 F28 D=30

Figure 3.11: Convergence graphs of CEO versus its peers on CEC2017 benchmark functions with 30 dimensions.

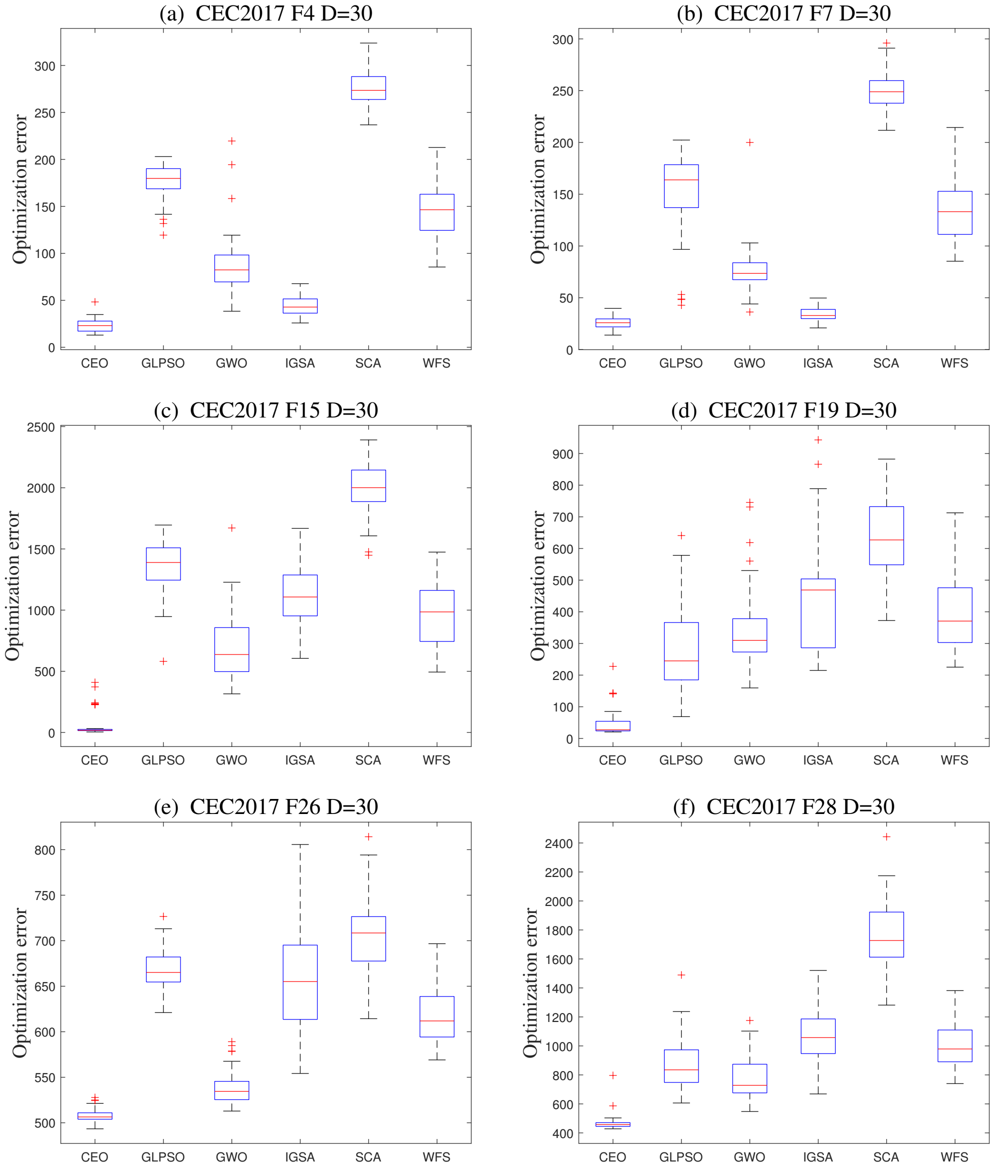


Figure 3.12: Box-and-whisker diagrams of CEO versus its peers on CEC2017 benchmark functions with 30 dimensions.

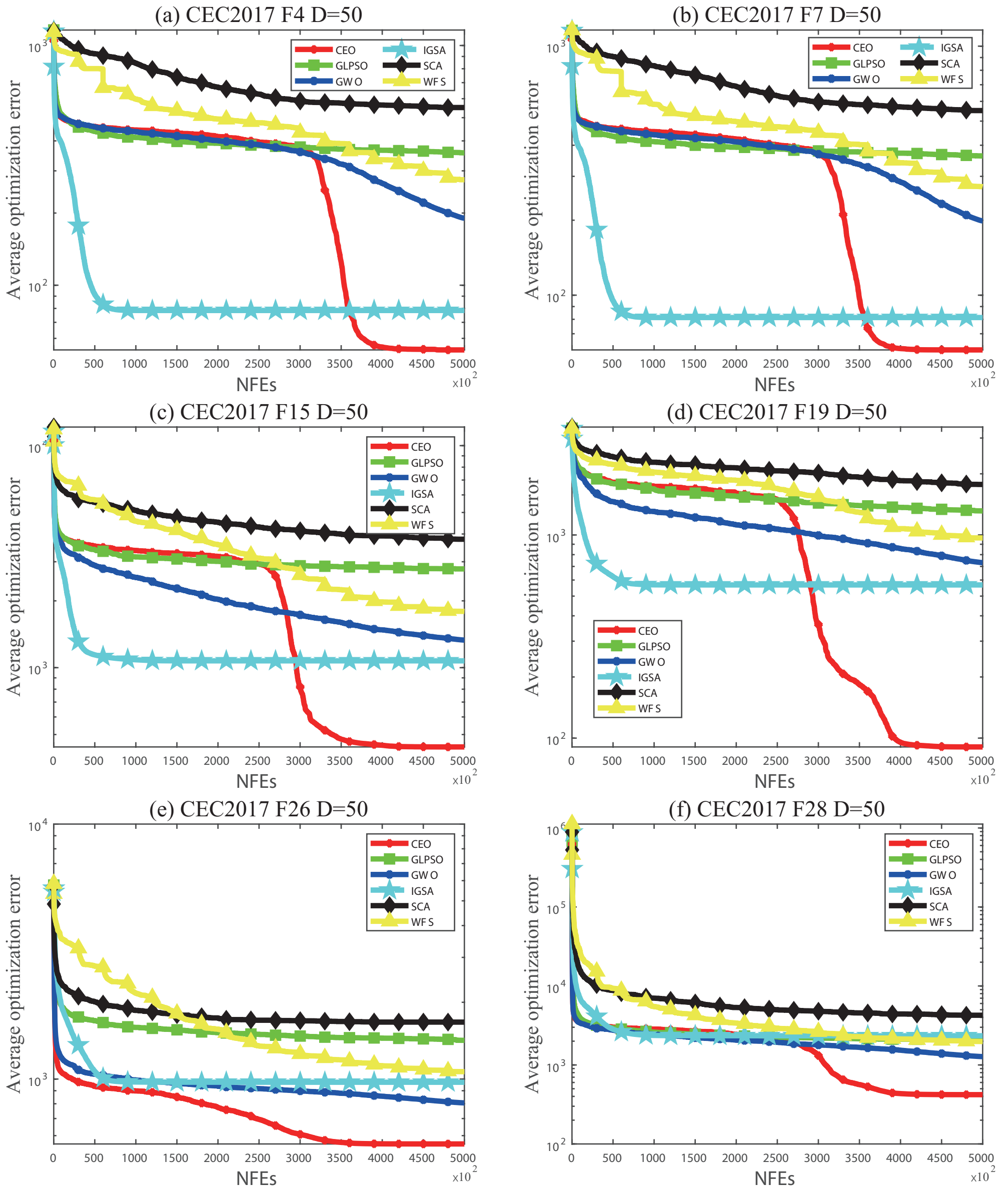
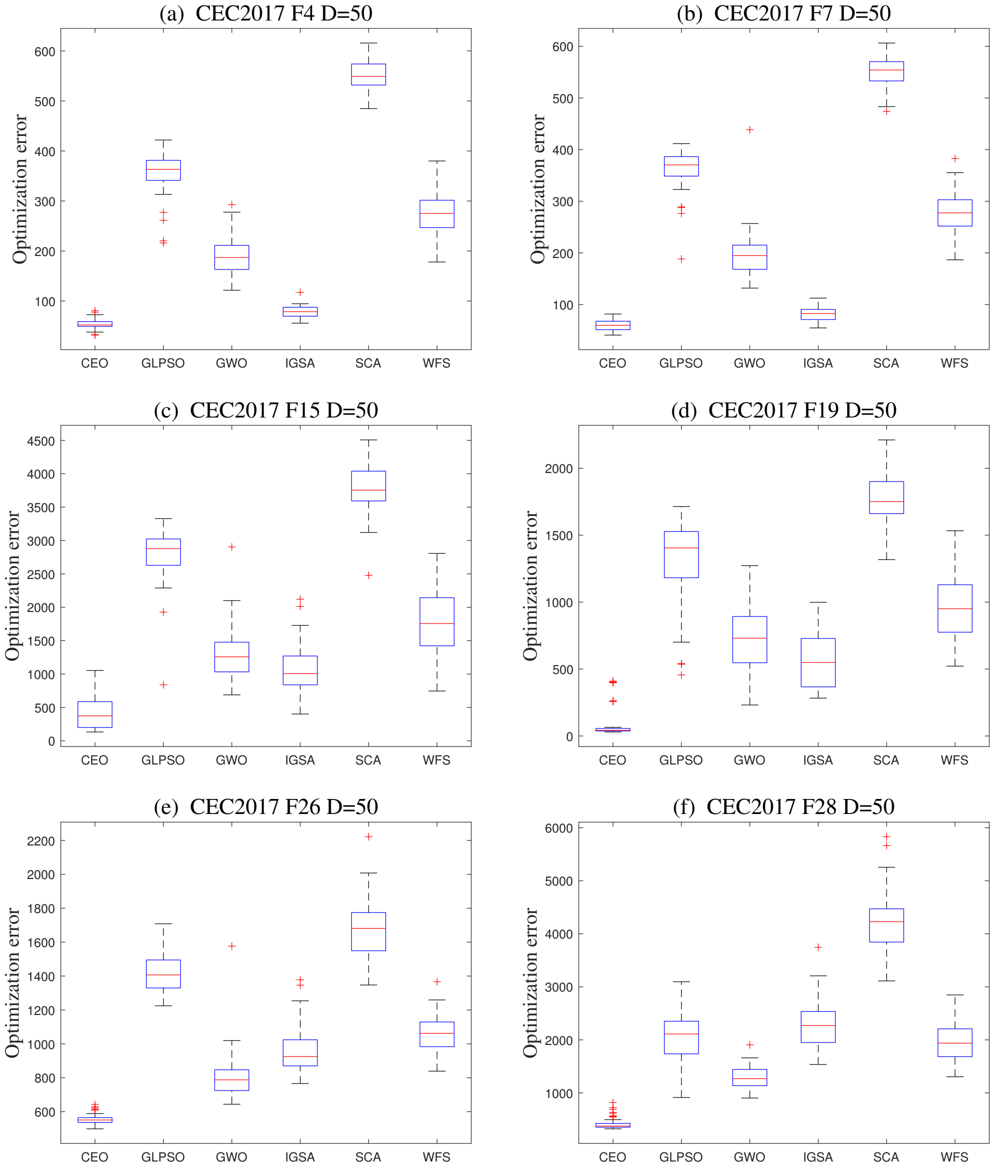


Figure 3.13: Convergence graphs of CEO versus its peers on CEC2017 benchmark functions with 50 dimensions.



(a) CEC2017 F4 D=50

(b) CEC2017 F7 D=50

(c) CEC2017 F15 D=50

(d) CEC2017 F19 D=50

(e) CEC2017 F26 D=50

(f) CEC2017 F28 D=50

Figure 3.14: Box-and-whisker diagrams of CEO versus its peers on CEC2017 benchmark functions with 50 dimensions.

Table 3.8: Experiment data of CEO versus its peers on CEC2017 benchmark functions with 50 dimensions.

Algorithm	CEC2017 D50 F1	CEC2017 D50 F2	CEC2017 D50 F3	CEC2017 D50 F4	CEC2017 D50 F5
CEO	5.39E+03 ± 5.22E+03	9.44E+04 ± 1.54E+04	**1.20E+02 ± 3.01E+01**	**5.37E+01 ± 9.38E+00**	7.52E-02 ± 3.34E-02
GLPSO	6.14E+06 ± 4.05E+07 –	**7.83E+04 ± 1.10E+04** –	8.64E+02 ± 2.49E+02 +	3.57E+02 ± 4.19E+01 +	1.47E+01 ± 2.96E+00 +
GWO	4.99E+09 ± 2.61E+09 +	7.19E+04 ± 1.58E+04 –	4.57E+02 ± 1.83E+02 +	1.90E+02 ± 3.75E+01 +	1.07E+01 ± 3.01E+00 +
IGSA	**2.36E+03 ± 1.44E+03** –	1.23E+05 ± 1.07E+04 +	2.03E+02 ± 3.51E+01 +	7.84E+01 ± 1.18E+01 +	**3.80E-03 ± 6.92E-03** –
SCA	3.84E+10 ± 5.66E+09 +	1.01E+05 ± 1.57E+04 +	5.66E+03 ± 1.35E+03 +	5.51E+02 ± 2.92E+01 +	6.87E+01 ± 4.96E+00 +
WFS	1.67E+09 ± 5.86E+08 +	3.40E+04 ± 5.75E+03 –	4.72E+02 ± 1.19E+02 +	2.75E+02 ± 4.78E+01 +	3.17E+01 ± 8.12E+00 +
	CEC2017 D50 F6	CEC2017 D50 F7	CEC2017 D50 F8	CEC2017 D50 F9	CEC2017 D50 F10
CEO	1.05E+02 ± 8.64E+00	**6.03E+01 ± 9.94E+00**	1.55E+00 ± 1.29E+00	**3.70E+03 ± 8.53E+02**	**1.61E+02 ± 4.03E+01**
GLPSO	3.68E+02 ± 8.17E+01 +	3.62E+02 ± 3.85E+01 +	1.49E+03 ± 1.10E+03 +	1.23E+04 ± 4.41E+02 +	7.27E+02 ± 6.01E+02 +
GWO	3.01E+02 ± 6.11E+01 +	1.99E+02 ± 4.68E+01 +	4.19E+03 ± 1.87E+03 +	5.79E+03 ± 8.45E+02 +	1.79E+03 ± 1.15E+03 +
IGSA	**7.98E+01 ± 1.18E+01** –	8.13E+01 ± 1.24E+01 +	**1.51E-01 ± 7.62E-01** –	4.25E+03 ± 6.75E+02 +	6.17E+02 ± 1.90E+02 +
SCA	9.07E+02 ± 6.45E+01 +	5.51E+02 ± 3.04E+01 +	2.08E+04 ± 3.71E+03 +	1.33E+04 ± 4.10E+02 +	4.80E+03 ± 1.21E+03 +
WFS	4.31E+02 ± 6.07E+01 +	2.73E+02 ± 4.18E+01 +	7.83E+03 ± 4.20E+03 +	8.58E+03 ± 1.04E+03 +	6.74E+02 ± 1.19E+02 +
	CEC2017 D50 F11	CEC2017 D50 F12	CEC2017 D50 F13	CEC2017 D50 F14	CEC2017 D50 F15
CEO	**3.42E+06 ± 2.02E+06**	**2.36E+04 ± 1.22E+04**	**7.84E+04 ± 6.63E+04**	1.68E+04 ± 6.47E+03	**4.40E+02 ± 2.54E+02**
GLPSO	1.25E+08 ± 3.15E+08 +	2.88E+06 ± 1.28E+07 –	2.81E+05 ± 4.38E+05 +	6.18E+03 ± 6.44E+03 –	2.79E+03 ± 4.03E+02 +
GWO	3.85E+08 ± 5.98E+08 +	9.06E+07 ± 1.31E+08 +	4.25E+05 ± 4.26E+05 +	6.73E+06 ± 1.31E+07 +	1.33E+03 ± 4.16E+02 +
IGSA	5.11E+06 ± 2.53E+06 +	3.88E+04 ± 1.65E+04 +	5.17E+05 ± 4.46E+05 +	**1.14E+04 ± 4.80E+03** –	1.07E+03 ± 3.71E+02 +
SCA	1.15E+10 ± 2.63E+09 +	2.66E+09 ± 8.45E+08 +	1.94E+06 ± 9.17E+05 +	3.40E+08 ± 1.46E+08 +	3.79E+03 ± 3.44E+02 +
WFS	3.32E+08 ± 1.77E+08 +	3.38E+06 ± 3.23E+06 +	1.11E+05 ± 9.38E+04 +	9.22E+05 ± 1.02E+06 +	1.79E+03 ± 4.81E+02 +
	CEC2017 D50 F16	CEC2017 D50 F17	CEC2017 D50 F18	CEC2017 D50 F19	CEC2017 D50 F20
CEO	**2.14E+02 ± 1.67E+02**	**2.95E+05 ± 1.89E+05**	1.87E+04 ± 1.19E+04	**9.05E+01 ± 1.16E+02**	**2.53E+02 ± 1.06E+01**
GLPSO	1.51E+03 ± 2.73E+02 +	3.87E+06 ± 4.31E+06 +	**6.48E+04 ± 3.81E+05** –	1.31E+03 ± 3.19E+02 +	5.76E+02 ± 2.58E+01 +
GWO	9.15E+02 ± 1.91E+02 +	2.18E+06 ± 2.22E+06 +	1.99E+06 ± 4.52E+06 +	7.31E+02 ± 2.54E+02 +	3.86E+02 ± 3.00E+01 +
IGSA	1.25E+03 ± 2.71E+02 +	1.64E+06 ± 6.72E+05 +	3.38E+04 ± 1.37E+04 +	5.67E+02 ± 2.03E+02 +	2.83E+02 ± 8.92E+00 +
SCA	2.58E+03 ± 2.44E+02 +	1.42E+07 ± 6.72E+06 +	2.20E+08 ± 1.06E+08 +	1.77E+03 ± 1.89E+02 +	7.62E+02 ± 3.22E+01 +
WFS	1.29E+03 ± 2.47E+02 +	1.60E+06 ± 9.38E+05 +	1.95E+06 ± 1.40E+06 +	9.67E+02 ± 2.35E+02 +	4.73E+02 ± 4.94E+01 +
	CEC2017 D50 F21	CEC2017 D50 F22	CEC2017 D50 F23	CEC2017 D50 F24	CEC2017 D50 F25
CEO	1.62E+03 ± 2.01E+03	**4.60E+02 ± 1.31E+01**	**5.38E+02 ± 1.25E+01**	**5.23E+02 ± 3.55E+01**	1.44E+03 ± 3.55E+02
GLPSO	1.12E+04 ± 3.84E+03 +	9.86E+02 ± 4.88E+01 +	1.06E+03 ± 4.61E+01 +	9.21E+02 ± 1.12E+02 +	5.96E+03 ± 6.94E+02 +
GWO	5.69E+03 ± 1.53E+03 +	6.27E+02 ± 5.96E+01 +	7.13E+02 ± 1.00E+02 +	8.80E+02 ± 1.88E+02 +	3.30E+03 ± 4.96E+02 +
IGSA	**1.00E+02 ± 1.97E-13** –	5.77E+02 ± 4.12E+01 +	5.50E+02 ± 2.55E+01 +	6.54E+02 ± 3.04E+01 +	**3.01E+02 ± 7.65E+00** –
SCA	1.37E+04 ± 3.71E+02 +	1.21E+03 ± 5.62E+01 +	1.26E+03 ± 5.27E+01 +	3.43E+03 ± 5.34E+02 +	9.06E+03 ± 6.19E+02 +
WFS	8.18E+03 ± 2.08E+03 +	8.06E+02 ± 6.41E+01 +	8.49E+02 ± 7.06E+01 +	9.04E+02 ± 9.02E+01 +	4.74E+03 ± 5.02E+02 +
	CEC2017 D50 F26	CEC2017 D50 F27	CEC2017 D50 F28	CEC2017 D50 F29	w/t/l
CEO	**5.57E+02 ± 3.06E+01**	**4.67E+02 ± 1.75E+01**	**4.21E+02 ± 1.12E+02**	**9.08E+05 ± 2.11E+05**	
GLPSO	1.42E+03 ± 1.13E+02 +	1.21E+03 ± 2.20E+02 +	2.04E+03 ± 4.92E+02 +	1.24E+07 ± 6.86E+06 +	24/0/3
GWO	8.06E+02 ± 1.35E+02 +	1.09E+03 ± 2.84E+02 +	1.27E+03 ± 2.17E+02 +	6.51E+07 ± 2.21E+07 +	28/0/1
IGSA	9.72E+02 ± 1.41E+02 +	6.09E+02 ± 8.17E+01 +	2.34E+03 ± 4.79E+02 +	3.59E+07 ± 7.50E+06 +	22/0/7
SCA	1.67E+03 ± 1.66E+02 +	3.53E+03 ± 4.53E+02 +	4.26E+03 ± 5.92E+02 +	5.83E+08 ± 1.84E+08 +	29/0/0
WFS	1.07E+03 ± 1.12E+02 +	1.26E+03 ± 3.10E+02 +	1.98E+03 ± 3.94E+02 +	2.00E+08 ± 3.75E+07 +	28/0/1

3.4.6 Comparison experiments with other algorithms on real-world optimization

The first test function of this experiment, i.e., the large scale transmission pricing problem (LSTPP) [55], is on transmission pricing. In modern, electricity has been deregulated, thus bringing about transmission owner as a separate identity. This has led to a division of users of electricity into two types: bilateral customers and pool customers. In this case, the pricing scheme becomes complex. This test function is used to help find the appropriate pricing scheme. It is a function with 126 dimensions constrained by a linear equality. Table 3.9 shows the results of the experiment, where the CEO achieved the best results. The convergence graph in Fig. 3.15 illustrates that CEO converges quickly at the beginning and accelerates again from about the 5000th iteration, and it always maintains a lower

Table 3.9: Experiment data of CEO versus its peers on LSTPP.

Algorithm	LSTPP	Significance
CEO	**3.16E+03 ± 8.38E+02**	
GLPSO	1.33E+06 ± 7.22E+04	+
GWO	5.94E+03 ± 2.95E+03	+
IGSA	7.90E+05 ± 1.11E+05	+
SCA	8.24E+05 ± 9.02E+04	+
WFS	8.01E+05 ± 1.00E+05	+

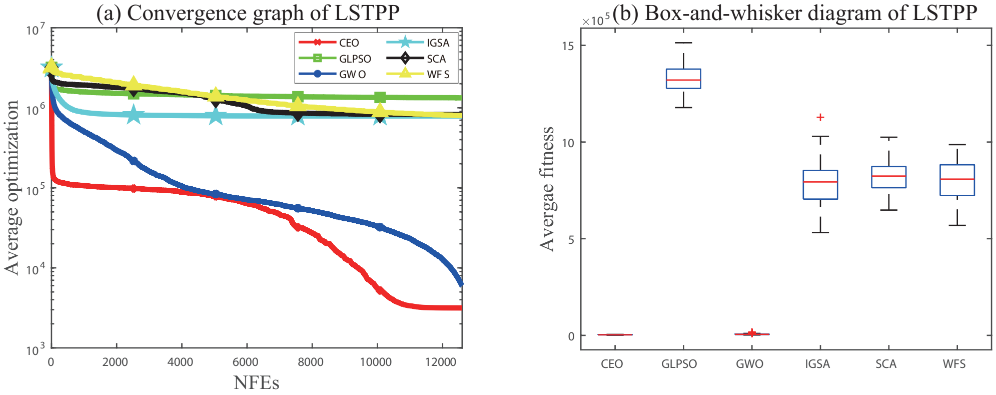


Figure 3.15: Convergence graph and box-and-whisker diagram of CEO and its peers for LSTPP.

curve than the other algorithms. The box-and-whisker diagram expresses that CEO is more stable than most of its peers. Although GWO also performs well, it is still slightly inferior to CEO.

The second test function, the spacecraft trajectory optimization problem (STOP) [55], is used to evaluate the distance values required for the spacecraft to reach Saturn in a particular fly-by sequence. The trajectory model of the spacecraft allows the use of deep space maneuvers between each planet making the problem complex. The problem has 22 dimensions. The results of the experiments are given in Table 3.10 and Fig. 3.16. It can be seen that the CEO has the best search ability and stability.

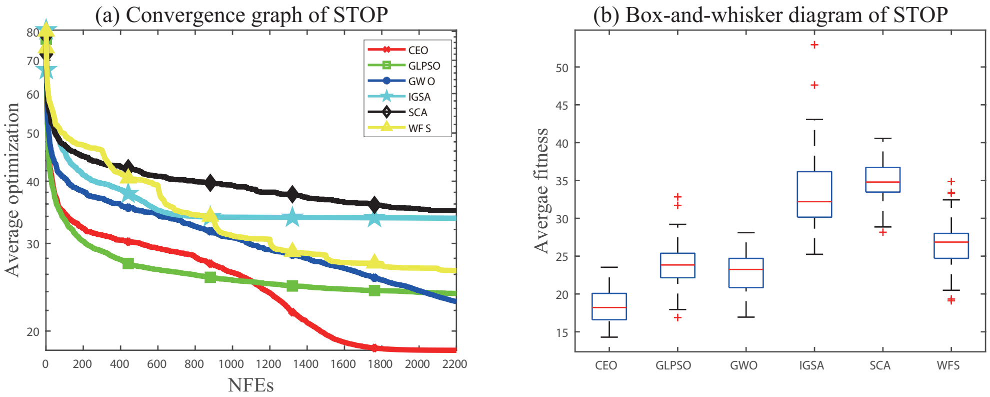


Figure 3.16: Convergence graph and box-and-whisker diagram of CEO and its peers for STOP.

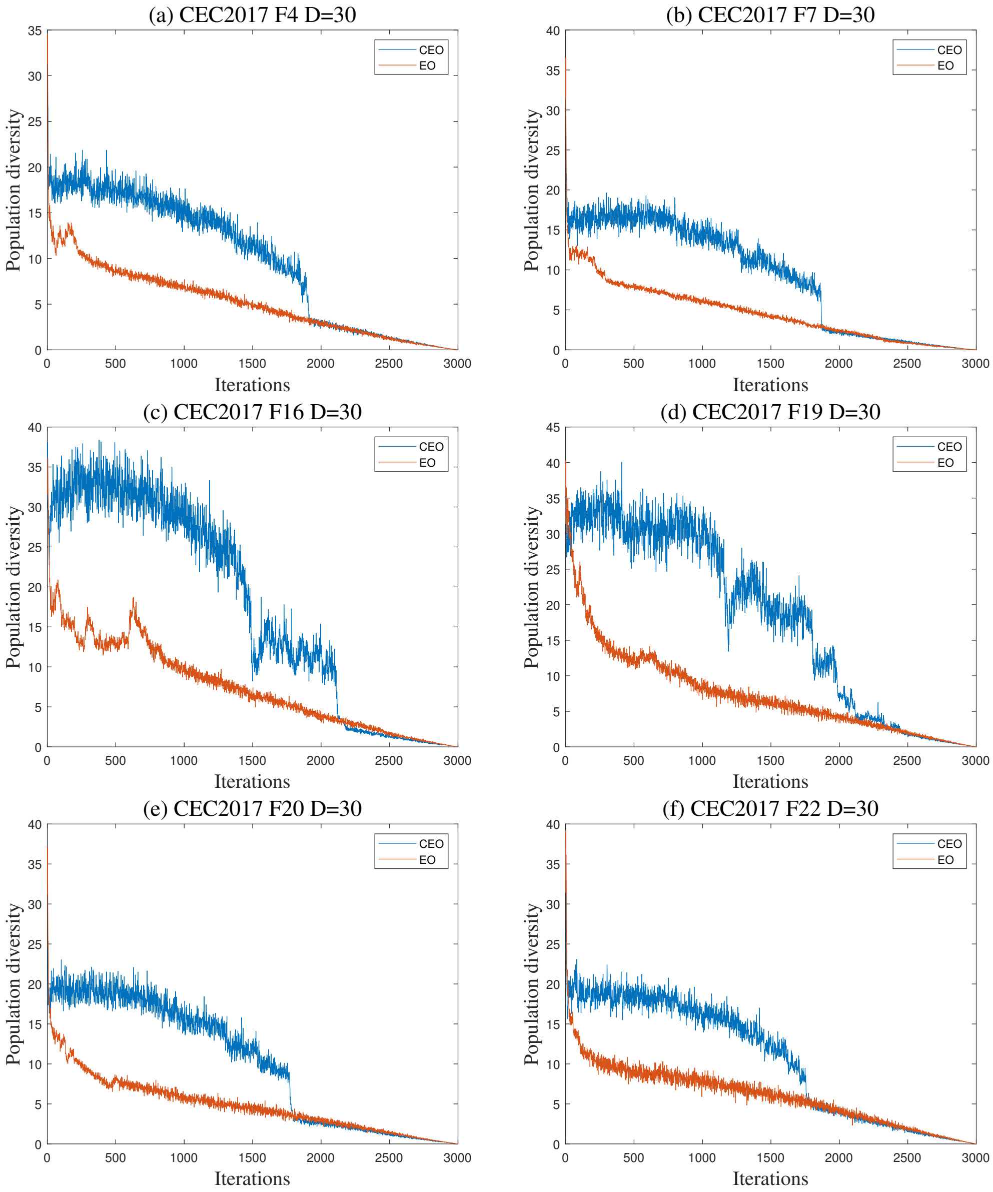
3.5 Discussion

3.5.1 Analysis of population diversity

Population diversity reflects the difference among individuals in a population, which is related to the search performance of the algorithm. The loss of algorithm diversity often leads to a lack of ability to jump out of the local optimum and thus convergence stagnates. In CEO, the differential search radius is a method to maintain population diversity. It combines the original feature of EO, which is inherited from four best particles in the equilibrium pool. These four particles are likely to be distributed in different regions when facing with multimodal functions or composition functions. This allows the local search to have a large radius, producing particles far from the current one, thus maintaining the differences among the particles. Moreover, in the later convergence phase of the algorithm,

Table 3.10: Experiment data of CEO versus its peers on STOP.

Algorithm	STOP	Significance
CEO	**1.83E+01 ± 2.41E+00**	
GLPSO	2.38E+01 ± 2.99E+00	+
GWO	2.29E+01 ± 2.80E+00	+
IGSA	3.37E+01 ± 5.49E+00	+
SCA	3.49E+01 ± 2.65E+00	+
WFS	2.65E+01 ± 3.39E+00	+


(a) CEC2017 F4 D=30

(b) CEC2017 F7 D=30

(c) CEC2017 F16 D=30

(d) CEC2017 F19 D=30

(e) CEC2017 F20 D=30

(f) CEC2017 F22 D=30

Figure 3.17: Population diversity analysis on CEC2017 30 dimensions.

when the particles are concentrated in a certain region, the search radius becomes smaller, which facilitates the exploitation of the current region.

Population diversity on CEC2017 benchmark functions with 30 dimensions was used as a reference for the analysis. A diversity equation based on the distance among individuals was used here, shown as:

$$\mu = \frac{1}{N} \cdot \sqrt{\sum_{i=1}^{N} (\left\| C_i - \bar{C} \right\|)^2} \tag{3.19}$$

$$\bar{C} = \frac{1}{N} \cdot \sum_{i=1}^{N} C_i \tag{3.20}$$

where μ denotes the diversity, $\bar{C}$ is the average of particles. Fig. 3.17 depicts the change process of diversity with the iteration of the algorithm. The commonality of these graphs is that the population diversity of the CEO is larger than that of the EO in the early stage, while the two tend to be similar in the late stage of convergence. In addition, the diversity curve of CEO has a larger amplitude, which may be due to the large search radius that makes the population more divergent.

3.5.2 Analysis of computational complexity

CEO has shown its effectiveness in benchmark functions and real-world problems. Its computational complexity also needs to be analyzed. Considering the size of the particle population N, the number of iterations K, the number of chaotic maps J, and using the generic notation O, the time complexity of CEO's key steps is shown as follows:

(1) Population initialization: $O(N)$

(2) Population boundary control: $O(N)$

(3) Population evaluation: $O(N)$

(4) Pick out the four best particles: $O(N) + O(N) + O(N) + O(N) = 4 \cdot O(N)$

(5) The chaotic map selection: $O(J)$

(6) CLS for one particle: $O(1)$

(7) CLS boundary control: $O(1)$

(8) CLS evaluation: $O(1)$

(9) Population update: $O(N)$

Thus the total time complexity is calculated as follows:

$$\begin{aligned}
&O(N) + K[O(N) + O(N) \\
&+ 4(O(N) + O(J) + O(1) + O(1) + O(1)) + O(N)] \\
&= O(N) + K(7 \cdot O(N) + 4 \cdot O(J) + 12 \cdot O(1)) \\
&= (7K + 1) \cdot O(N) + 4 \cdot O(J) + 12 \cdot O(1)
\end{aligned} \tag{3.21}$$

Since J is a constant, the total time complexity of CEO is $O(N)$. The operations brought by CLS are of constant level, which makes the time complexity of CEO comparable to that of EO.

3.6 Conclusion

This paper presents an adaptive algorithm CEO based on differential radius and chaotic local search. The differential radius allows the algorithm to be more divergent in the early stage and still possess convergence in the later stage. This endows the CEO with higher population diversity and a better balance of exploration and exploitation. The radius is adjusted by the values generated by the chaotic maps. Multiple chaotic maps are used here, and they are selected by an adaptive mechanism. This mechanism gives each chaotic map a variable called success intensity and uses the success intensity to play roulette. The success intensity records the improvement that each successful local search brings to the algorithm. The higher the cumulative success intensity, the larger the probability of a chaotic map being selected. All these allow the CEO to have the appropriate search radius at different stages to gain the ability to jump out of the local optimal or further exploitation. In addition, the results of the Friedman test show that multiple chaotic maps mechanism outperforms that of single chaotic map. Benchmark function set CEC2017 and several real-world problems were used to test the performance of CEO. The results show that CEO has a better ability to jump out of the local optimal in comparison with with EO and some other state-of-the-art algorithms.

However, there are some related issues that aroused our interest. For EO, it is valuable to study other selection criteria for the equilibrium pool, such as directly selecting a particle farthest from the population to control the population diversity. For CLS, other available information besides the success intensity, such as the number of successes, or even the use of multiple criteria for selection, might be beneficial to the selections. The success of CLS is inextricably linked to the equilibrium pool of EO, and whether this mechanism can be combined well with other MHAs is still a open question. In addition, the proposed CEO can also be extended to tackle multiple/many objective optimization problems arisen from Internet of things [82–84], bio-informatics [85], and so on.

Chapter 4

General conclusions and remarks

In this thesis, we introduced the reasons for the birth of metaheuristic algorithms and summarized four algorithm classes, including evolutionary algorithms, population intelligence algorithms, physics-based algorithms, and human-based algorithms. And the representative algorithms of these four classes, GA, PSO, GSA and BSO, are described in detail.

Then a success intensity-based chaotic local search is proposed to improve the performance of the metaheuristic algorithms. The key to this strategy lies in a differential search radius and a chaotic maps selection mechanism based on the success intensity. The differential radius is overall decreasing with iteration, which benefits exploration in the early stages and exploitation in the later stages. Since the chaotic sequence generated by the chaotic map has been proved to be better than the standard random number generator on some problems, in this paper, chaotic random values are used instead of standard random values to adjust the radius of the local search. In addition, there are references that show the use of multiple chaotic sequences is superior to that of single chaotic sequences. Therefore, the chaotic random values in this paper are derived from multiple chaotic sequences, and the selection of chaotic sequences depends on the success intensity. Success intensity is a quantity used to measure the magnitude of the improvement brought to the algorithm by each chaotic sequence. It accumulates the increment of the fitness of each chaotic sequence at the successful search. The higher the success intensity of a chaotic sequence, the higher the probability that the chaotic sequence will be selected to participate in the next iteration. This strategy of local search allows the algorithm to achieve a better balance between exploration and exploitation and improves the search performance.

We apply this strategy to two new algorithms, the spherical evaluation algorithm and the equilibrium optimizer, and proposed chaotic spherical evolution algorithm (CSE) and chaotic equilibrium optimizer (CEO). The improved algorithms perform better than the original algorithms on both benchmark functions and real-world optimization problems. And they are also competitive with some mainstream algorithms.

Success intensity as a kind of historical information is used in this paper to adjust the radius of local search. We also consider that there may be other useful historical information. In the future, finding other suitable historical information for local search will be one of the focus of work. This also includes strategies to improve the algorithm by using multiple historical information together. Moreover, other ways of using chaotic maps in metaheuristic algorithms are also part of our considerations.

Bibliography

[1] J. H. Holland, *Adaptation in natural and artificial systems: an introductory analysis with application to biology, control and artificial intelligence*. University of Michigan Press, 1975.

[2] J. Kennedy and R. Eberhart, “Particle swarm optimization,” in *Proceedings of ICNN’95-International Conference on Neural Networks*, vol. 4. IEEE, 1995, pp. 1942–1948.

[3] E. Rashedi, H. Nezamabadi-Pour, and S. Saryazdi, “GSA: a gravitational search algorithm,” *Information sciences*, vol. 179, no. 13, pp. 2232–2248, 2009.

[4] Y. Shi, “Brain storm optimization algorithm,” in *International conference in swarm intelligence*. Springer, 2011, pp. 303–309.

[5] J. Del Ser, E. Osaba, D. Molina, X.-S. Yang, S. Salcedo-Sanz, D. Camacho, S. Das, P. N. Suganthan, C. A. C. Coello, and F. Herrera, “Bio-inspired computation: Where we stand and what’s next,” *Swarm and Evolutionary Computation*, vol. 48, pp. 220–250, 2019.

[6] I. BoussaïD, J. Lepagnot, and P. Siarry, “A survey on optimization metaheuristics,” *Information Sciences*, vol. 237, pp. 82–117, 2013.

[7] J. H. Holland, “Genetic algorithms,” *Scientific American*, vol. 267, no. 1, pp. 66–73, 1992.

[8] Y. Sun, B. Xue, M. Zhang, G. G. Yen, and J. Lv, “Automatically designing CNN architectures using the genetic algorithm for image classification,” *IEEE Transactions on Cybernetics*, vol. 50, no. 9, pp. 3840 – 3854, 2020.

[9] S. Gao, W. Wang, H. Dai, F. Li, and Z. Tang, "Improved clonal selection algorithm combined with ant colony optimization," *IEICE Transactions on Information and Systems*, vol. 91, no. 6, pp. 1813–1823, 2008.

[10] S. Gao, Y. Wang, J. Cheng, Y. Inazumi, and Z. Tang, "Ant colony optimization with clustering for solving the dynamic location routing problem," *Applied Mathematics and Computation*, vol. 285, pp. 149–173, 2016.

[11] Y. Wang, S. Gao, Y. Yu, and Z. Xu, "The discovery of population interaction with a power law distribution in brain storm optimization," *Memetic Computing*, vol. 11, pp. 65–87, 2019.

[12] Y. Yu, S. Gao, S. Cheng, Y. Wang, S. Song, and F. Yuan, "CBSO: a memetic brain storm optimization with chaotic local search," *Memetic Computing*, vol. 10, no. 4, pp. 353–367, 2017.

[13] Y. Yu, S. Gao, Y. Wang, J. Cheng, and Y. Todo, "ASBSO: An improved brain storm optimization with flexible search length and memory-based selection," *IEEE Access*, vol. 6, pp. 36 977–36 994, 2018.

[14] X.-S. Yang, "Nature-inspired optimization algorithms: Challenges and open problems," *Journal of Computational Science*, vol. 46, p. 101104, 2020.

[15] A. Telikani, A. H. Gandomi, and A. Shahbahrami, "A survey of evolutionary computation for association rule mining," *Information Sciences*, vol. 524, pp. 318–352, 2020.

[16] T. Dokeroglu, E. Sevinc, T. Kucukyilmaz, and A. Cosar, "A survey on new generation metaheuristic algorithms," *Computers & Industrial Engineering*, vol. 137, p. 106040, 2019.

[17] Y. Wang, Y. Yu, S. Cao, X. Zhang, and S. Gao, "A review of applications of artificial intelligent algorithms in wind farms," *Artificial Intelligence Review*, vol. 53, no. 5, pp. 3447–3500, 2020.

[18] M. Črepinšek, S.-H. Liu, and M. Mernik, “Exploration and exploitation in evolutionary algorithms: A survey,” *ACM Computing Surveys (CSUR)*, vol. 45, no. 3, pp. 1–33, 2013.

[19] D. Tang, “Spherical evolution for solving continuous optimization problems,” *Applied Soft Computing*, vol. 81, p. 105499, 2019.

[20] N. Noman and H. Iba, “Accelerating differential evolution using an adaptive local search,” *IEEE Transactions on evolutionary Computation*, vol. 12, no. 1, pp. 107–125, 2008.

[21] Y. Lu, J. Zhou, H. Qin, Y. Wang, and Y. Zhang, “Chaotic differential evolution methods for dynamic economic dispatch with valve-point effects,” *Engineering Applications of Artificial Intelligence*, vol. 24, no. 2, pp. 378–387, 2011.

[22] A. R. Jordehi, “A chaotic artificial immune system optimisation algorithm for solving global continuous optimisation problems,” *Neural Computing and Applications*, vol. 26, no. 4, pp. 827–833, 2015.

[23] J. Sun, S. Gao, H. Dai, J. Cheng, M. Zhou, and J. Wang, “Bi-objective elite differential evolution for multivalued logic networks,” *IEEE Transactions on Cybernetics*, vol. 50, no. 1, pp. 233–246, 2020.

[24] R. Caponetto, L. Fortuna, S. Fazzino, and M. G. Xibilia, “Chaotic sequences to improve the performance of evolutionary algorithms,” *IEEE Transactions on Evolutionary Computation*, vol. 7, no. 3, pp. 289–304, 2003.

[25] G.-G. Wang, S. Deb, A. H. Gandomi, Z. Zhang, and A. H. Alavi, “Chaotic cuckoo search,” *Soft Computing*, vol. 20, no. 9, pp. 3349–3362, 2016.

[26] F. Han, X. Gu, Z. Wang, H. Fan, J. Cao, and Q. Lu, “Global firing rate contrast enhancement in e/i neuronal networks by recurrent synchronized inhibition,” *Chaos: An Interdisciplinary Journal of Nonlinear Science*, vol. 28, no. 10, p. 106324, 2018.

[27] F. Han, Z. Wang, Y. Du, X. Sun, and B. Zhang, "Robust synchronization of bursting hodgkin–huxley neuronal systems coupled by delayed chemical synapses," *International Journal of Non-Linear Mechanics*, vol. 70, pp. 105–111, 2015.

[28] B. Alatas, E. Akin, and A. B. Ozer, "Chaos embedded particle swarm optimization algorithms," *Chaos, Solitons & Fractals*, vol. 40, no. 4, pp. 1715–1734, 2009.

[29] A. H. Gandomi, G. J. Yun, X.-S. Yang, and S. Talatahari, "Chaos-enhanced accelerated particle swarm optimization," *Communications in Nonlinear Science and Numerical Simulation*, vol. 18, no. 2, pp. 327–340, 2013.

[30] S. Gao, K. Wang, S. Tao, T. Jin, H. Dai, and J. Cheng, "A state-of-the-art differential evolution algorithm for parameter estimation of solar photovoltaic models," *Energy Conversion and Management*, vol. 230, p. 113784, 2021.

[31] L. S. Coelho and V. C. Mariani, "Combining of chaotic differential evolution and quadratic programming for economic dispatch optimization with valve-point effect," *IEEE Transactions on Power Systems*, vol. 21, no. 2, pp. 989–996, 2006.

[32] A. H. Gandomi, X.-S. Yang, S. Talatahari, and A. H. Alavi, "Firefly algorithm with chaos," *Communications in Nonlinear Science and Numerical Simulation*, vol. 18, no. 1, pp. 89–98, 2013.

[33] A. H. Gandomi and X.-S. Yang, "Chaotic bat algorithm," *Journal of Computational Science*, vol. 5, no. 2, pp. 224–232, 2014.

[34] G.-G. Wang, L. Guo, A. H. Gandomi, G.-S. Hao, and H. Wang, "Chaotic krill herd algorithm," *Information Sciences*, vol. 274, pp. 17–34, 2014.

[35] M. Mitić, N. Vuković, M. Petrović, and Z. Miljković, "Chaotic fruit fly optimization algorithm," *Knowledge-Based Systems*, vol. 89, pp. 446–458, 2015.

[36] C. Choi and J.-J. Lee, "Chaotic local search algorithm," *Artificial Life and Robotics*, vol. 2, no. 1, pp. 41–47, 1998.

[37] S. Gao, C. Vairappan, Y. Wang, Q. Cao, and Z. Tang, “Gravitational search algorithm combined with chaos for unconstrained numerical optimization,” *Applied Mathematics and Computation*, vol. 231, pp. 48–62, 2014.

[38] S. Gao, Y. Yu, Y. Wang, J. Wang, J. Cheng, and M. Zhou, “Chaotic local search-based differential evolution algorithms for optimization,” *IEEE Transactions on Systems, Man and Cybernetics: Systems*, vol. 51, no. 6, pp. 3954–3967, 2021.

[39] J. Brest, S. Greiner, B. Boskovic, M. Mernik, and V. Zumer, “Self-adapting control parameters in differential evolution: A comparative study on numerical benchmark problems,” *IEEE Transactions on Evolutionary Computation*, vol. 10, no. 6, pp. 646–657, 2006.

[40] A. K. Qin, V. L. Huang, and P. N. Suganthan, “Differential evolution algorithm with strategy adaptation for global numerical optimization,” *IEEE Transactions on Evolutionary Computation*, vol. 13, no. 2, pp. 398–417, 2008.

[41] R. Tanabe and A. Fukunaga, “Success-history based parameter adaptation for differential evolution,” in *2013 IEEE Congress on Evolutionary Computation*. IEEE, 2013, pp. 71–78.

[42] J. Zhang and A. C. Sanderson, “JADE: adaptive differential evolution with optional external archive,” *IEEE Transactions on Evolutionary Computation*, vol. 13, no. 5, pp. 945–958, 2009.

[43] B. Alatas, “Chaotic bee colony algorithms for global numerical optimization,” *Expert Systems with Applications*, vol. 37, no. 8, pp. 5682–5687, 2010.

[44] R. Storn and K. Price, “Differential evolution–a simple and efficient heuristic for global optimization over continuous spaces,” *Journal of Global Optimization*, vol. 11, no. 4, pp. 341–359, 1997.

[45] S. Gao, Y. Wang, J. Wang, and J. Cheng, “Understanding differential evolution: A Poisson law derived from population interaction network,” *Journal of Computational Science*, vol. 21, pp. 140–149, 2017.

[46] Y. Yu, S. Gao, Y. Wang, and Y. Todo, “Global optimum-based search differential evolution,” *IEEE/CAA Journal of Automatica Sinica*, vol. 6, no. 2, pp. 379–394, 2018.

[47] Z.-H. Zhan, Z.-J. Wang, H. Jin, and J. Zhang, “Adaptive distributed differential evolution,” *IEEE Transactions on Cybernetics*, vol. 50, no. 11, pp. 4633–4647, 2019.

[48] Z.-J. Wang, Z.-H. Zhan, Y. Lin, W.-J. Yu, H. Wang, S. Kwong, and J. Zhang, “Automatic niching differential evolution with contour prediction approach for multimodal optimization problems,” *IEEE Transactions on Evolutionary Computation*, vol. 24, no. 1, pp. 114–128, 2019.

[49] J. J. Liang, A. K. Qin, P. N. Suganthan, and S. Baskar, “Comprehensive learning particle swarm optimizer for global optimization of multimodal functions,” *IEEE Transactions on Evolutionary Computation*, vol. 10, no. 3, pp. 281–295, 2006.

[50] Z.-H. Zhan, J. Zhang, Y. Li, and Y.-H. Shi, “Orthogonal learning particle swarm optimization,” *IEEE Transactions on Evolutionary Computation*, vol. 15, no. 6, pp. 832–847, 2010.

[51] Y.-J. Gong, J.-J. Li, Y. Zhou, Y. Li, H. S.-H. Chung, Y.-H. Shi, and J. Zhang, “Genetic learning particle swarm optimization,” *IEEE Transactions on Cybernetics*, vol. 46, no. 10, pp. 2277–2290, 2015.

[52] X.-F. Liu, Z.-H. Zhan, Y. Gao, J. Zhang, S. Kwong, and J. Zhang, “Coevolutionary particle swarm optimization with bottleneck objective learning strategy for many-objective optimization,” *IEEE Transactions on Evolutionary Computation*, vol. 23, no. 4, pp. 587–602, 2018.

[53] Z.-J. Wang, Z.-H. Zhan, S. Kwong, H. Jin, and J. Zhang, “Adaptive granularity learning distributed particle swarm optimization for large-scale optimization,” *IEEE Transactions on Cybernetics*, pp. 1175–1188, 2020.

[54] N. Awad, M. Ali, J. Liang, B. Qu, and P. Suganthan, “Problem definitions and evaluation criteria for the CEC 2017 special session and competition on single objective real-parameter numerical optimization,” *Tech. Rep.*, 2016.

[55] S. Das and P. N. Suganthan, “Problem definitions and evaluation criteria for CEC 2011 competition on testing evolutionary algorithms on real world optimization problems,” *Jadavpur University, Nanyang Technological University, Kolkata*, pp. 341–359, 2010.

[56] J. Carrasco, S. García, M. Rueda, S. Das, and F. Herrera, “Recent trends in the use of statistical tests for comparing swarm and evolutionary computing algorithms: Practical guidelines and a critical review,” *Swarm and Evolutionary Computation*, vol. 54, p. 100665, 2020.

[57] Y. Wang, Y. Yu, S. Gao, H. Pan, and G. Yang, “A hierarchical gravitational search algorithm with an effective gravitational constant,” *Swarm and Evolutionary Computation*, vol. 46, pp. 118–139, 2019.

[58] J. Ji, S. Gao, S. Wang, Y. Tang, H. Yu, and Y. Todo, “Self-adaptive gravitational search algorithm with a modified chaotic local search,” *IEEE Access*, vol. 5, pp. 17 881–17 895, 2017.

[59] Z. Cao, Y. Shi, X. Rong, B. Liu, Z. Du, and B. Yang, “Random grouping brain storm optimization algorithm with a new dynamically changing step size,” in *International Conference in Swarm Intelligence*. Springer, 2015, pp. 357–364.

[60] Y. Yu, S. Gao, Y. Wang, Z. Lei, J. Cheng, and Y. Todo, “A multiple diversity-driven brain storm optimization algorithm with adaptive parameters,” *IEEE Access*, vol. 7, pp. 126 871–126 888, 2019.

[61] S. Mirjalili, “SCA: a sine cosine algorithm for solving optimization problems,” *Knowledge-based systems*, vol. 96, pp. 120–133, 2016.

[62] Y. Wang, S. Gao, M. Zhou, and Y. Yu, “A multi-layered gravitational search algorithm for function optimization and real-world problems,” *IEEE/CAA Journal of Automatica Sinica*, vol. 8, no. 1, pp. 1–16, 2021.

[63] Z. Lei, S. Gao, S. Gupta, J. Cheng, and G. Yang, "An aggregative learning gravitational search algorithm with self-adaptive gravitational constants," *Expert Systems with Applications*, p. 113396, 2020.

[64] S. Gao, M. Zhou, Y. Wang, J. Cheng, H. Yachi, and J. Wang, "Dendritic neural model with effective learning algorithms for classification, approximation, and prediction," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 30, no. 2, pp. 601–614, 2019.

[65] S. Gao, S. Song, J. Cheng, Y. Todo, and M. Zhou, "Incorporation of solvent effect into multi-objective evolutionary algorithm for improved protein structure prediction," *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, vol. 15, no. 4, pp. 1365–1378, 2018.

[66] J. J. Cheng, G. Y. Yuan, M. C. Zhou, S. Gao, Z. H. Huang, and C. Liu, "A connectivity prediction-based dynamic clustering model for VANET in an urban scene," *IEEE Internet of Things Journal*, vol. 7, no. 9, pp. 8410–8418, 2020.

[67] J. Cheng, G. Yuan, M. Zhou, S. Gao, C. Liu, H. Duan, and Q. Zeng, "Accessibility analysis and modeling for IoV in an urban scene," *IEEE Transactions on Vehicular Technology*, vol. 69, no. 4, pp. 4246–4256, 2020.

[68] Z. Cui, J. Zhang, D. Wu, X. Cai, H. Wang, W. Zhang, and J. Chen, "Hybrid many-objective particle swarm optimization algorithm for green coal production problem," *Information Sciences*, vol. 518, pp. 256–271, 2020.

[69] N. Maleki, Y. Zeinali, and S. T. A. Niaki, "A k-nn method for lung cancer prognosis with the use of a genetic algorithm for feature selection," *Expert Systems with Applications*, vol. 164, p. 113981, 2021.

[70] J. Uthayakumar, N. Metawa, K. Shankar, and S. Lakshmanaprabu, "Financial crisis prediction model using ant colony optimization," *International Journal of Information Management*, vol. 50, pp. 538–556, 2020.

[71] C. Shilaja and T. Arunprasath, “Optimal power flow using moth swarm algorithm with gravitational search algorithm considering wind power,” *Future Generation Computer Systems*, vol. 98, pp. 708–715, 2019.

[72] C. Narmatha, S. M. Eljack, A. A. R. M. Tuka, S. Manimurugan, and M. Mustafa, “A hybrid fuzzy brain-storm optimization algorithm for the classification of brain tumor mri images,” *Journal of Ambient Intelligence and Humanized Computing*, pp. 1–9, 2020.

[73] B. Morales-Castañeda, D. Zaldivar, E. Cuevas, F. Fausto, and A. Rodríguez, “A better balance in metaheuristic algorithms: Does it exist?” *Swarm and Evolutionary Computation*, vol. 54, p. 100671, 2020.

[74] A. Faramarzi, M. Heidarinejad, B. Stephens, and S. Mirjalili, “Equilibrium optimizer: A novel optimization algorithm,” *Knowledge-based Systems*, vol. 191, p. 105190, 2020.

[75] H. D. Nguyen, I. Yoshihara, K. Yamamori, and M. Yasunaga, “Implementation of an effective hybrid GA for large-scale traveling salesman problems,” *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 37, no. 1, pp. 92–99, 2007.

[76] P. Merz and B. Freisleben, “Fitness landscape analysis and memetic algorithms for the quadratic assignment problem,” *IEEE Tactions on Evolutionary Computation*, vol. 4, no. 4, pp. 337–352, 2000.

[77] M. Tang and X. Yao, “A memetic algorithm for VLSI floorplanning,” *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 37, no. 1, pp. 62–69, 2007.

[78] R. Senkerik, I. Zelinka, and M. Pluhacek, “Chaos-based optimization-a review,” *Journal of Advanced Engineering and Computation*, vol. 1, no. 1, pp. 68–79, 2017.

[79] Z. Song, S. Gao, Y. Yu, J. Sun, and Y. Todo, “Multiple chaos embedded gravitational search algorithm,” *IEICE Transactions on Information and Systems*, vol. 100, no. 4, pp. 888–900, 2017.

[80] S. Mirjalili, S. M. Mirjalili, and A. Lewis, “Grey wolf optimizer,” *Advances in Engineering Software*, vol. 69, pp. 46–61, 2014.

[81] N. Covic and B. Lacevic, “Wingsuit flying search—a novel global optimization algorithm,” *IEEE Access*, vol. 8, pp. 53 883–53 900, 2020.

[82] X. Cai, S. Geng, D. Wu, J. Cai, and J. Chen, “A multicloud-model-based many-objective intelligent algorithm for efficient task scheduling in internet of things,” *IEEE Internet of Things Journal*, vol. 8, no. 12, pp. 9645–9653, 2020.

[83] X. Cai, S. Geng, J. Zhang, D. Wu, Z. Cui, W. S. Zhang, and J. Chen, “A sharding scheme based many-objective optimization algorithm for enhancing security in blockchain-enabled industrial internet of things,” *IEEE Transactions on Industrial Informatics*, 2021, doi: 10.1109/TII.2021.3051607.

[84] Z. Cui, Z. Zhang, Z. Hu, S. Geng, and J. Chen, “A many-objective optimization based intelligent high performance data processing model for cyber-physical-social systems,” *IEEE Transactions on Network Science and Engineering*, 2021, doi: 10.1109/TNSE.2021.3073911.

[85] Z. Lei, S. Gao, Z. Zhang, M. Zhou, and J. Cheng, “MO4: A many-objective evolutionary algorithm for protein structure prediction,” *IEEE Transactions on Evolutionary Computation*, 2021, doi: 10.1109/TEVC.2021.3095481.

Acknowledgements

Throughout the research and writing process for my PhD, I received a great deal of guidance, help and encouragement.

I would deepest like to thank my supervisors, Prof. Tang and Associate Prof. Gao of Toyama University, for guiding and helping me find the suitable topic and showing me the way with their extensive knowledge and experience when I was confused. Associate prof. Gao told me that there is no need to be inferior because of coming from other majors, and the knowledge from other majors is a valuable asset for our research, which gave me a lot of confidence.

I sincerely thank all the members of the research lab for their patient discussions and serious cooperation, as well as for their support and help in my life. They are the ones who made me not lonely in a foreign country.

I would particularly like to thank my parents and my girlfriend for their great support in life and for understanding and supporting my research. They are the ones who made it all started and supported it to the end.

I hope this thesis is the beginning of my research career rather than the end, and I hope the first half of this sentence is not just a hope.