IEEE Access
Multidisciplinary : Rapid Review : Open Access Journal

Received 21 October 2022, accepted 9 December 2022, date of publication 12 December 2022, date of current version 21 December 2022.

Digital Object Identifier 10.1109/ACCESS.2022.3228782

RESEARCH ARTICLE

An Improved Lion Swarm Optimization Algorithm With Chaotic Mutation Strategy and Boundary Mutation Strategy for Global Optimization

JUNFENG LIU[1] **AND YUN WU**[2]
[1]State Key Laboratory of Industrial Control Technology, Control Department, Zhejiang University, Hangzhou 310027, China
[2]College of Science, Jiujiang University, Jiujiang 332005, China

Corresponding author: Junfeng Liu (junfeng_liu@zju.edu.cn)

This work was supported in part by the Scientific and Technological Project of Jiangxi Education Office under Grant GJJ211823, in part by the National Key Research and Development Program of China under Grant 2021YFC2101100, in part by the National Natural Science Foundation of China under Grant 62073288 and Grant 12075212, and in part by the Zhejiang University NGICS Platform.

ABSTRACT Lion swarm optimization (LSO) inspired by the natural division of labor among lion king, lionesses and lion cubs in a lion group, i.e., lion king guarding, lionesses hunting and lion cubs following, is a relatively novel swarm intelligent optimization technique. Due to its remarkable performance, the canonical LSO has been extensively researched. However, how to balance contradictions between the exploration and the exploitation and alleviate the premature convergence are two critical concerns that need to be dealt with in the LSO study. To address these two drawbacks, enhance the optimization performance, and broaden its application domain, an improved lion swarm optimization algorithm with chaotic mutation strategy and a boundary mutation strategy (CBLSO) is proposed in this paper. In the proposed algorithm, a chaotic mutation strategy based on chaotic cubic mapping is designed to enhance the exploration ability of the algorithm, while the boundary mutation strategy based on the concept of multilevel parallel is adopted to manage boundary constraint violations, which is beneficial for improving the exploitation ability of the algorithm. The proposed CBLSO is evaluated on 56 classic test functions and 30 CEC2014 benchmark functions, and is compared with quite a few state-of-the-art algorithms regarding often-used performance metrics. The experimental results demonstrate the superior performance of the embedded strategies on balancing the exploration and the exploitation. Furthermore, the proposed CBLSO is applied to the optimal dispatch problem of cascade hydropower stations based on a novel constraints handling method designed in this paper to validate its good practicability and performance. The experimental results of a case study on the optimal dispatch problem of China's Wujiang cascade hydropower stations indicate that the proposed CBLSO can produce better and more reliable optimal results than the canonical LSO and other comparison algorithms with competitive speed. Thus, we can conclude that the proposed CBLSO is a competitive and effective alterative tool to solve complex numerical optimization problems and real-world optimization with complicated constraints.

INDEX TERMS Meta-heuristic optimization algorithm, lion swarm optimization (LSO) algorithm, swarm intelligence, optimal dispatch, cascade hydropower stations.

I. INTRODUCTION

Global optimization problems (GOPs) that concern a large number of decision variables and complex constraints exist extensively in various fields of science [1], [2], engineering design [3], [4], business and economics [5], which cannot be solved well within adequate time or specific accuracy by traditional deterministic methods. In order to address such optimization problems, a lot of computational intelligence methods, especially the meta-heuristic optimization algorithms that inspired natural organisms, social behaviors, biological behaviors and physical phenomena have emerged

The associate editor coordinating the review of this manuscript and approving it for publication was Frederico Guimarães.

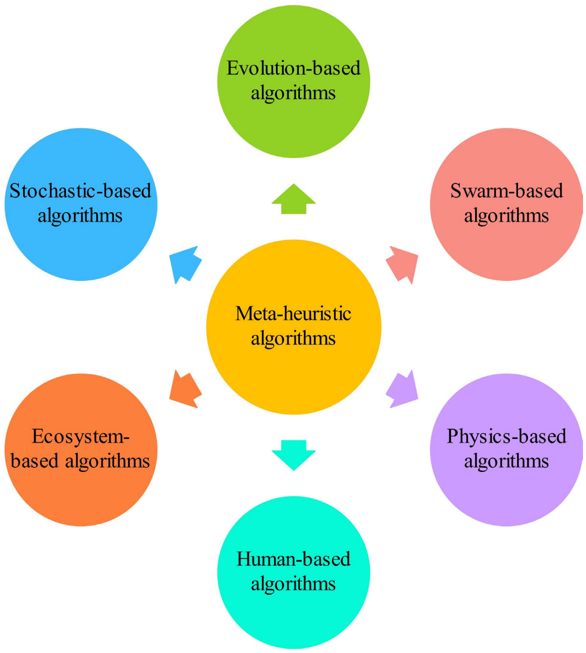


FIGURE 1. Classification of meta-heuristic algorithms.

in the past few decades. Roughly, the main branch of meta-heuristic algorithms can be broadly classified into six main categories: stochastic-based algorithms, evolution-based algorithms, swarm-based algorithms, physics-based algorithms, human-based algorithms, and ecosystem-based algorithms [6], [7], [8], [9], [10], [11], [12], [13], [14], which are shown in Fig. 1.

Stochastic-based algorithms use randomness to perform non-deterministic behaviors, usually different from the purely deterministic procedures. The most famous stochastic-based algorithms are Adaptive Random Search (ARS) [15], Random Search (RS) [16], Tabu Search (TS) [17], Hill Climbing (HC) [18], Local Search (LS) [19], and Guided Search (GLS) [19]. Evolution-based algorithms imitate the evolutionary behaviors, i.e., reproduction, mutation, recombination and selection of creatures found in nature. The search process in these algorithms usually starts with a randomly generated population, which further evolves over successive generations. For evolution-based algorithms, the main strength point is the best individuals are always combined together to form the new generation since it can promote the improvement of population over the course of iterations. Evolution Strategy (ES) [20], [21], Evolutionary Programming (EP) [22], Genetic Algorithm (GA) [23], Genetic Programming (GP) [24], Evolutionary Algorithm (EA) [25], and Differential Evolution (DE) [26], [27] can be considered as the most standard form of evolution-based algorithms. Swarm-based algorithms are swarm intelligence-based techniques, which simulate the intelligent social and individual behaviors of swarms, animals, herds, teams, or any group of creatures. Ant Colony Optimization (ACO) [28], Particle Swarm Optimization (PSO) [29], [30], Artificial Bee Colony (ABC) [31], Cuckoo Search (CS) [32], and Bat Algorithm (BA) [33] can be described as representative algorithms in Swarm-based algorithms. Some of the recent Swarm-based algorithms are Grey Wolf Optimizer (GWO) [34], Lion Swarm Optimization (LSO) [35], Squirrel Search Algorithm (SSA) [6], and Harris Hawks Optimization (HHO) [36]. Physics-based algorithms are motivated by the basic physical rules that exist in universe. The inspiring physical systems range from metallurgy, music, the interplay between culture and evolution, science (chemistry, physics, mathematics), and complex dynamic systems. Generally, a physics-based algorithm is a combination of local search techniques and global search techniques. Some of the prevailing algorithms of this category are Simulated Annealing (SA) [37], [38], Big Bang-Big Crunch (BB-BC) [39], Gravitational Search Algorithm (GSA) [40], and Atom Search Optimization (ASO) [41], [42]. Human-based algorithms are inspired by the human behaviors, while ecosystem-based algorithms are mainly based on the natural ecosystem. The most state-of-the-art Human-based algorithms are Harmony Search (HS) [43], Imperialist Competitive Algorithm (ICA) [44], League Championship Algorithm (LCA) [45] and Teaching-Learning-Based Optimization (TLBO) [46], [47], while the most well-known ecosystem-based algorithms are Invasive Weed Optimization (IWO) [48], Biogeography-Based Optimization (BBO) [49], and Multi-species optimizer based on PSO (PS^2O) [50].

Apart from the aforementioned algorithms, various improved versions based on the basic versions of existing meta-heuristic optimization algorithms are also proposed and have been successfully applied to almost all areas of operational research, data mining, neural networks, image and video processing, machine intelligence, and many other fields of knowledge. Thousands of research papers and dozens of books have been published [51], [52].

In spite of the rapid development of meta-heuristic optimization algorithms, to learn from the nature for developing a better algorithm to solve new and more complex GOPs is still in progress and new meta-heuristic optimization algorithms are still emerging, such as Marine Predators Algorithm (MPA) [53], Slime Mould Algorithm (SMA) [54], Golden Eagle Optimizer (GEO) [55], Artificial Hummingbird Algorithm (AHA) [56], and so on. In addition, according to the "no free lunch" (NFL) theory [57], there is no single meta-heuristic optimization algorithms best suited for dealing with all GOPs. This means that a particular optimization method may show competitive outcomes on a set of optimization problems, but may not provide promising results on the other set of optimization problems. The NFL theory, certainly, forms the basis for the researchers to improve the existing algorithms or design new and powerful algorithms for better optimization now and then.

Lion swarm optimization (LSO) proposed by Liu et al. [35], is a new entrant in the domain of meta-heuristic algorithms based on the natural division of labor among lion king, lionesses and lion cubs in a lion group, i.e., lion king guarding, lionesses hunting and lion cubs following. Due to advantages, e.g., simple structure, few control parameter, ease of implementation, good robustness, and fast convergence speed, LSO has been widely used in solving various real-world optimization problems and many variants of LSO are also developed. Some excellent works have been reported: Yang and Wei [58] proposed an improved LSO algorithm for optimizing the long short-term memory recurrent neural network problem. In this algorithm, the chaos theory was introduced for accelerating the global convergence speed of the algorithm. Zhang and Jiang [59] presented a parallel discrete lion swarm optimization (PDSLO) algorithm for solving the traveling salesman problem (TSP). In the PDLSO, discrete coding and order crossover operators were firstly used to form a discrete lion swarm optimization (DLSO). And then, the complete 2-opt (C2-opt) algorithm was employed to enhance the local search ability of the PDLSO algorithm. In addition, the PDLSO has multiple populations and each sub-population independently ran the DLSO algorithm in parallel as well as transferred information among them by using the ring topology. Qiao et al. [60] proposed a novel hybrid lion swarm optimization algorithm to optimize the traditional least squares support vector machine model by combing lion swarm optimization algorithm and genetic algorithm. Guo and Jiang [61] presented an improved lion swarm optimization algorithm for solving the job-shop scheduling problem. In this algorithm, chaos search and gaussian perturbation strategy were added to the position of lions in the past dynasties, which could improve the optimization efficiency of the algorithm in the optimization process. Although these researches demonstrate that LSO has greater application potential, LSO stills has some noticeable deficiencies in solving some GOPs. Specially, when tackling the complex GOPs that have a high eccentric ellipse, a narrow curving valley, or GOPs characterized with multimodality, high-dimension and the existence of several local minima and many global minima, like most nature-inspired meta-heuristic optimization algorithms, LSO fails to effectively find near-optimal solutions. This is mainly because of their stochastic nature, which may erect barriers for them to tackle some types of complex GOPs.

To address this weakness, chaotic mutation strategy based on chaotic cubic mapping and boundary mutation strategy based on the concept of multilevel parallel maybe two alternative technologies as they have been studied experimentally in multi-objective PSO for handling multi-objective optimization problems (MOPs) [62] and PSO for handling GOPs [63], respectively, and proven to be two effective and efficient technologies to improve the performance of PSO when tackling various categories of GOPs. Hence, it is reasonable to believe that the chaotic mutation strategy and the boundary mutation strategy can be introduced into LSO to address its premature convergence problem and improve the robustness as well as population diversity when tackling different types of GOPs.

In this paper, a novel LSO algorithm with a chaotic mutation strategy and a boundary mutation strategy (CBLSO) is proposed for GOPs and optimal dispatch problem of cascade hydropower stations. Compared with the standard LSO and other meta-heuristic optimization algorithms, the main features of our CBLSO algorithm can be stated as follows:

(a) The CBLSO can achieve good performance with various updating strategies of the population which inherit from the standard LSO. Broadly, adaptability and choice of the fittest are two distinct features of nature that imitated by all meta-heuristic optimization algorithms. The population is recursively updated at each iteration according to the suitable updating mechanism. As previously stated, there are many meta-heuristic optimization algorithms inspired by animal social behaviors. Like PSO, ACO, ABC, BA, GWO, and SSA et al, imitating animal foraging behavior is also the solution updating strategy of the CBLSO. However, there are some evident differences between CBLSO and other algorithms. Similar to PSO and BA, a new direction for movement of lion individual is generated by considering the local best position (*pbest*) and the global best position (*gbest*) obtained by lion individual and lion group so far, while other algorithms (ACO, ABC, GWO, and SSA) do not adopt. Besides, PSO and BA update the population by using a single strategy, while CBLSO utilizes different strategies for different lion group. Thus, the position updating mechanism of CBLSO differs from PSO and BA. Although CBLSO, ABC, GWO, and SSA work on the effective division of labor, i.e., the whole population is divided into various regions. Different from ABC and GWO, CBLSO initially sorts the whole population in ascending order of fitness values and then divides it, which is controlled by the user. Although the dividing mode of CBLSO and SSA looks quite similar, they technically present some differences. In CBLSO, the highest hierarchy is the lion king and the number of lion king is only one, while the number of the highest hierarchy in SSA is the least ones. Besides, in CBLSO, for a certain hierarchy (lion cubs), the updating strategy is diversified, while the updating strategy used by SSA for a certain hierarchy is single. Hence, as discussed previously, we are confident that our CBLSO will be a good competitor for existing nature-inspired meta-heuristic optimization algorithms.

(b) The introduced chaotic mutation strategy. In the standard LSO, both lionesses and lion cubs use *gbest* information possessed by the lion king to update the hunting behavior, directly or indirectly. Through this mechanism, the whole lion group may be easily trapped into a poor zone once the lion king is located around a local optimum. However, a chaotic mutation strategy based on chaotic cubic mapping is designed herein could improve the exploration ability of the CBLSO, since the regions where the lion king is located will be explored many times by lionesses and lion cubs. Hence, the whole lion group may be pulled toward the sparsest region. Its execution is determined by a pre-defined threshold, i.e., when the lion king's fitness value does not change or changes little in three consecutive generations, the chaotic mutation strategy is executed. The advantages of this lie in two aspects: (1) exploring more near-optimal solutions and (2) retaining the balance of the exploration ability and the exploitation ability of the algorithm. The efficiency of the chaotic mutation strategy is experimentally tested in Section 4.2.

(c) The introduced boundary mutation strategy. Different from the common boundary constraint approach used in the standard LSO, a boundary mutation strategy based on the concept of multilevel parallel is introduced in CBLSO for managing boundary constraint violations, which is beneficial for the maintaining of the population diversity and improving the global search ability. The advantage of the boundary mutation strategy is experimentally invested in Section 4.2.

Based on the above three main features, we are confident that the CBLSO will be effective and competitive. To evaluate the performance of the proposed algorithm, a comprehensive experiment study is conducted to compare the CBLSO with various state-of-the-art optimization algorithms on fifty-six classic test functions, encompassing Uni-modal, Multi-modal, Separable, Non-separable and Multi-dimension problems, and thirty 30-dimensional benchmark functions from the CEC 2014 test suite. The experiment results indicate that the CBLSO can provide better global optimal solutions with high quality. Besides, the results of non-parametric statistical tests demonstrate that the CBLSO performs significantly better than or at least comparable to the well-known algorithms and can be taken as a promising alternative optimization tool to solve GOPs. Finally, to validate the great potential of the CBLSO for real-world application, based on the novel constraints handling method designed in this paper, the CBLSO is applied to solve the optimal dispatch problem of cascade hydropower stations in Section 6. The obtained results show that the proposed algorithm can produce competitive results when compared with the standard LSO and other comparison algorithms.

The remainder of this paper is organized as follows. In Section 2, the mathematical model of the canonical LSO is presented. Details of the CBLSO are described in Section 3. In Section 4, the effects of the chaotic mutation strategy and the boundary mutation strategy are first well studied through a comparative experiment, and then a series of comparative experiments between the CBLSO and other state-of-the-art optimization algorithms, comparative statistical analysis of simulation results on 56 test problems are presented and discussed. Thereafter, In Section 5, the CBLSO is further compared with other well-known optimizers on 56 classic functions as well as other competitive optimization algorithms on the benchmark functions from the CEC 2014 test suite. Section 6 provides the application on real-world optimization problems. Finally, in Section 7, discussions, conclusions and some future works are given. All classic test functions are listed in the Appendix.

II. CANONICAL LSO

Lion swarm optimization (LSO) algorithm is a Swarm-based optimization algorithm inspired by the hunting behavior of the lion group. In LSO, just as in real life, there are three kinds of lions: lion king, lionesses and lion cubs. Starting from an initial position of the search space to be optimized, the lion with the best fitness value is assumed to be the lion king. Then, a certain proportion of lions are selected as hunting lions (usually are lionesses) and they hunt together with other lionesses of the group. Once a prey found is better than the prey currently occupied by the lion king, the position of the prey will be possessed by the lion king. The lion cubs follow the lionesses to learn how to hunt or forage for food near the lion king. After reaching sexual maturity and they are not stronger than the lion king, the lion cubs will be driven out of the group and become nomadic lions. In order to survive, the nomadic lions will try to approach the best position in his or her memory. According to the natural division of labor and cooperate with each other, the lions of the group repeatedly search for the optimal of the objective function.

A. RANDOM INITIALIZATION

In LSO, each lion position represents a potential solution to the considered problem, and the quality of prey corresponds to the quality (fitness) of the associated solution. Specifically, for a D-dimensional GOP, an initial population P called lion group containing n solutions (lion positions) will be randomly generated in search space in the beginning. The position of the ith lion will be generated through the following equation:

$$x_{i,j} = x_{min,j} + \text{rand}\,(0, 1) \cdot \left(x_{max,j} - x_{min,j}\right) \tag{1}$$

where $x_{i,j}$ represents the jth dimension of the ith lion; $i = 1, 2, \cdots, n; j = 1, 2, \cdots, D$ and rand (0, 1) is a random number that distributed uniformly in the range [0,1]. $x_{min,j}$ and $x_{max,j}$ are the lower and upper bounds of the jth dimension, respectively. Up to now, the initialization of the population is completed.

Thereafter, the number of adult lions, i.e., lion king and lioness, $nLeader(2 \leq nLeader \leq n/2)$ will be defined through the following equation:

$$nLeader = \lceil n \cdot \beta \rceil \tag{2}$$

where β is a random number in the range [0,1] that is acknowledged as the proportion factor of the adult lions (including lion king and lionesses). In LSO, the proportion of the adult lions in a group has an important effect on the final optimization result. The bigger the proportion of the adult lions, the fewer the number of the lion cubs. However, the updating pattern of the lion cubs' position is various, which may increase the otherness of the group and improve the exploration of the algorithm. To balance the exploitation and the exploration of LSO, the proportion factor of adult lions β is considered to be less than 0.5 for all cases in this paper [35]. Here, β is set to be 0.2 suggested by literature [64].

The number of lion cubs is $n - nLeader$.

After the population initialization, the fitness value of position for each lion is calculated by putting the values of decision variable (solution vector) into the user-defined fitness functions. The fitness value of each lion's position represents the quality of prey searched by it, i.e., optimal prey (possessed by lion king), normal prey (possessed by lioness) and little prey (possessed by lion cubs) and hence their probability of survival also.

B. SORTING AND DECLARATION

After calculating the fitness values of each lion's position, we sort the fitness values in ascending order. The lion with minimal fitness value is considered to be the lion king. The next best $nLeader - 1$ lions are declared as lionesses. The remaining lions are supposed to be lion cubs. Next, all the lions can conduct hunting behavior by adopting the position updating mechanism.

C. HUNTING BEHAVIORS OF LIONS

As mentioned previously, the LSO works on the effective division of the natural labor, i.e., the whole group is divided into various regions (including lion king, lioness and lion cubs). In each generation, it is assumed that different lions conduct different hunting behaviors to update their position. The dynamic hunting behavior can be mathematically modeled as follows:

Lion king: To ensure the priority for prey than the other lions, the lion king may move in the range of the best food, i.e., the position that has the minimal fitness value. In this case, the new position of lion king can be obtained as follows:

$$x_i(t+1) = gbest(t) \cdot (1 + \gamma \cdot \|pbest_i(t) - gbest(t)\|) \tag{3}$$

where γ is normally distributed random number in the range [0,1], t is the current iteration value, $pbest_i(t)$ is the historically best position of the ith lion at the current iteration, $gbest(t)$ is the global best position of the lion group at the current iteration.

Lionesses: Recognizing the position of prey, encircling them, and then attacking towards the prey is the usually hunting behavior of lioness. When lioness conducts the hunting behavior, they usually cooperate with another lioness. In this case, the new position of lionesses can be obtained as follows:

$$x_i(t+1) = \frac{pbest_i(t) + pbest_c(t)}{2} \cdot \left(1 + \alpha_f \cdot \gamma\right) \tag{4}$$

where $pbest_c(t)$ is the historically best position of the cooperation lioness at the current iteration, α_f is moving range disturbance factor of lionesses, defined as follows:

$$\alpha_f = step \cdot \exp\left(-30 \cdot \frac{t}{t_{max}}\right)^{10} \tag{5}$$

where t_{max} is the maximum iteration value, $step$ denotes the maximal step value of the lioness' activity range computed as follows:

$$step = a \cdot (\bar{x}_{max} - \bar{x}_{\min}) \tag{6}$$

where $\bar{x}_{max}$ and $\bar{x}_{min}$ are maximal mean value and minimal mean value of each dimension, respectively.

Lion cubs: As discussed above, three situations may occur during the dynamic hunting of the lion cubs. Note that the lioness followed by lion cubs is randomly selected from the lioness group, which means that the followed relationship between the lioness and the lion cubs is randomly established. In this case, the new position of lion cubs can be obtained as follows:

$$x_i(t+1) = \begin{cases} \frac{gbest(t) + pbest_i(t)}{2} \cdot (1 + \alpha_c \cdot \gamma), & q \le \frac{1}{3} \\ \frac{pbest_m(t) + pbest_i(t)}{2} \cdot (1 + \alpha_c \cdot \gamma), & \frac{1}{3} < q < \frac{2}{3} \\ \frac{\overline{gbest}(t) + pbest_i(t)}{2} \cdot (1 + \alpha_c \cdot \gamma), & \frac{2}{3} \le q \le 1 \end{cases} \tag{7}$$

where q is a random number in the range [0,1], $pbest_m(t)$ is the historically best position followed by lion cubs at the current iteration, α_c is the moving range disturbance factor of lion cubs, $\overline{gbest}(t)$ is the position that ith lion cub is driven out of the scope of hunting. At the position that far from lion king, it is an elite opposition-based learning (OBL) strategy that has been verified to be an effective method for solving the optimization problems [65], [66]. The introduction of OBL can enlarge the search space of nomadic lion cubs and is helpful to improve the global exploration of the LSO.

α_c is defined as follows:

$$\alpha_c = step \cdot \frac{t_{max} - t}{t_{max}} \tag{8}$$

where $step$ denotes the maximal step value of lion cubs' activity range computed by Eq. (6):

$\overline{gbest}(t)$ is defined as follows:

$$\overline{gbest}(t) = \bar{x}_{max} + \bar{x}_{min} - gbest(t) \tag{9}$$

III. PROPOSED CBLSO ALGORITHM

In meta-heuristic optimization algorithms, generally, the optimization process may be divided into two conflicting milestones: the exploration and the exploitation, where enhancing one may result in degrading the other. In the early phase of the optimization process, the candidate solutions should be encouraged to explore the whole search space instead of clustering around the local optima, which is beneficial for improving the population diversity and causing a high exploration of the whole search space. In the later phase, the candidate solutions have to exploit the information gathered to converge towards the global optima, which is aimed at enhancing the quality of the solutions. Although there are various improvement methods to balance the two milestones and promote local optima avoidance, the works of literatures [62], [63] indicate that the chaotic mutation strategy and the boundary mutation strategy are better in tackling this issue. In the following subsections, the chaotic mutation strategy and the boundary mutation strategy are firstly described in detail, respectively. Then the CBLSO algorithm is proposed.

A. CHAOTIC MUTATION STRATEGY

Recently, following various realm of humans, a large number of chaotic maps developed by researchers, physicians and mathematicians are available in the optimization field. Out of all these available chaotic maps, the bulk of them has been applied widely in optimization algorithms for enhancing the local search ability of the algorithms and avoiding premature convergence effectively [67], [68], [69], [70]. To avoid the degeneration of the candidate solutions caused by an irregular random variation, a chaotic mutation strategy based on chaotic cubic mapping with exquisite internal structure is introduced to the LSO as the local exploitation, defined as follows [67], [68]:

$$\boldsymbol{x}_i^{new}(t) = \boldsymbol{x}_i(t) \cdot (1 + \eta \cdot z_i) \tag{10}$$

where $\boldsymbol{x}_i^{new}(t)$ and $\boldsymbol{x}_i(t)$ are the state of the ith candidate solutions before and after mutation at the current iteration, respectively; η denotes the control factor of mutation, defined as follows:

$$\eta = 1 - \frac{t-1}{t_{max}-1} \tag{11}$$

It should be noted that the value of η significantly affects the effectiveness of the chaotic mutation strategy. From Eq. (11), we can conclude that the value of η constructs an arithmetic progression with 1 as the first term, 0 as the last term, and $-1/(t_{max}-1)$ as the tolerance. $\eta = 1$ represents chaotic mutation plays the most vital role, while $\eta = 0$ represents chaotic mutation plays the least vital role.

The chaotic sequences of z_i are calculated as follows:

$$z_1 = 1 - 2 \cdot \text{rand}(1, D) \tag{12}$$

$$z_{i+1} = 4 \cdot z_i^3 - 4 \cdot z_i, -1 \le z_i \le 1 \tag{13}$$

B. BOUNDARY MUTATION STRATEGY

Each GOP has various boundary constraints that limit the search space. When an individual moves out of the search space, an approach to managing boundary constraint violations will be employed. However, different boundary constraint approaches on the performance of the algorithm are different, which have been validated in works of literatures [71], [72]. When individuals move outside the predefined bounds of the multi-dimensional asymmetric search space, using the unified boundary constraint approaches to manage boundary constraint violations may lead to new boundary constraint violations or weak the search ability of individuals. To solve this problem, based on the idea of a hierarchical GA, introduce the concept of the multilevel parallel, set the asymmetrical search spaces and the layer's parameters separately, and operate them in parallel style. If an individual's position violates the asymmetrical boundary for a specific dimension, it conducts mutation near the respond boundary and maintains the individual among the effective search space. Mathematically, the boundary mutation strategy is defined as:

$$if\ x_{i,j} > x_{max,j}, \quad x_{i,j} = x_{max,j} - c \cdot \text{rand}(0,1) \tag{14}$$

$$if\ x_{i,j} < x_{min,j}, \quad x_{i,j} = x_{min,j} + c \cdot \text{rand}(0,1) \tag{15}$$

where $x_{max} = [x_{max,1}, x_{max,2}, \cdots, x_{max,D}]$ is the upper bounds of decision variables, while $x_{min} = [x_{min,1}, x_{min,2}, \cdots, x_{min,D}]$ is the lower bounds of decision variables, respectively. c is the parameter in the range $[10^{-2}, 10^{-4}]$.

C. FRAMEWORK OF CBLSO

The aforementioned subsections have described the main procedures of the LSO, the chaotic mutation strategy, and the boundary mutation strategy in detail, which compose the main components of the CBLSO. The main steps of the CBLSO are summarized as follows:

In the initialization stage, a population with n lions are randomly generated by using Eq. (1). Then, set the numbers of the lion king, the lionesses, and the lion cubs. After evaluating the fitness value of each lion's position, the sorting, declaration, and random selection procedures are performed to sort the positions of lions in ascending order based on their fitness value and declare the lion king, the lionesses and the lion cubs. Then, the lionesses' $pbest_c$ and $pbest_m$ as the cooperation partner and the mother lioness followed by lion cub are selected, respectively. And the CBLSO turns to the main loop of the evolutionary process until the iteration value t or the number of function evaluations (denote as $NFEs$) reaches the predefined maximum iteration value t_{max} or the maximum $NFEs$ (denote as $maxNFEs$), defined as $maxNFEs = n \times t_{max}$.

During the evolutionary stage, the LSO search is first conducted. The values of the related parameters, i.e., α_f, $step$, and α_c are calculated according to Eq. (5), Eq. (6), and Eq. (8), respectively. The hunting behaviors of the lion king, the lionesses, and the lion cubs are updated by using Eq. (3), Eq. (4), Eq. (7), and the boundary mutation strategy illustrated in Section 3.2 is adopted at the same time. After the positional information for each lion is renewed, the fitness values of new lions are evaluated. Once the absolute value of difference for the lion king between tth iteration and $t+3$th iteration is smaller than the predefined value (denote as u), usually is considered in the range $[10^{-6}, 10^{-4}]$. The chaotic mutation strategy and the boundary mutation strategy introduced in Section 3.1 and Section 3.2 are called, respectively. Then, the fitness values of the mutant solutions are computed and the lionesses $pbest_c$ and $pbest_m$, the values of related parameters, i.e., α_f, $step_1$, α_c, and $step_2$ are updated again. Moreover, the lion king, the lionesses, and the lion cubs are sorted with 10 increments in the iteration. The above evolutionary phase will be repeated until the predefined termination condition is satisfied. At the end of the algorithm, the lion is reported as the final near-optimal solution. For the sake of clarity, the pseudo-code of the CBLSO is outlined in Fig. 2. and the complete flowchart of the CBLSO is described as Fig. 3.

D. EQUATIONSTIME COMPLEXITY ANALYSIS

Any meta-heuristic optimization algorithms should have less computational complexity so that the real-world optimization problems can be solved in less computational efforts.

Algorithm 1: Pseudo-codes of the CBLSO

Initialize the parameters values
$t = 0$;
for $i = 1$ to n **do**
  Randomly initialize the position x_i of lion i by using Eq. (1);
  Evaluate the fitness of x_i;
end for
Sort the positions of lions in ascending order depending on their fitness value;
Declare the lion king, lionesses and lion cubs based on the proportion factor of adult lions;
Record the $pbest_i$ of lion i as noted by and the $gbest$ of the lion group;
while ($t <= t_{max}$) **do**
  $t = t + 1$;
  Calculate the value of α_f, α_c, η;
  Randomly generate the value of γ;
  for $i = 1$ to n **do**
    if $i == 1$ (lion king) **then**
      Update the position of lion king by using Eq. (3);
    else if $i == 2$ to $nLeader$ (lionesses) **then**
      Select the lionesses $pbest_c$ as the cooperation partner;
      Update the position of lionesses using Eq. (4);
    else $i == nLeader + 1$ to n (lion cubs) **then**
      Randomly generate the value of q;
      Select the lionesses $pbest_m$ as the mother lioness followed by lion cub;
      Update the position of lion cubs using Eq. (7);
      Execute the boundary mutation strategy described in Section 3.2;
      Evaluate the fitness value of the new solution;
    end if
  end for
  if abs (fit^{t+3}(lion king) - fit^{t}(lion king)) $< u$ **then**
    Execute the chaotic mutation strategy described in Section 3.1;
    Execute the boundary mutation strategy described in Section 3.2;
    Evaluate the fitness value of the new solution;
  end if
  Update the $pbest_i$ of lion i as noted by and the $gbest$ of the lion group;
  if t mod $10 == 0$ **then**
    Sort the positions of lions in ascending order depending on their fitness value;
    Declare the lion king, lionesses and lion cubs;
  end if
end while
Report the position of $gbest$ as the final optimal solution.

FIGURE 2. **The pseudo code of the CBLSO algorithm.**

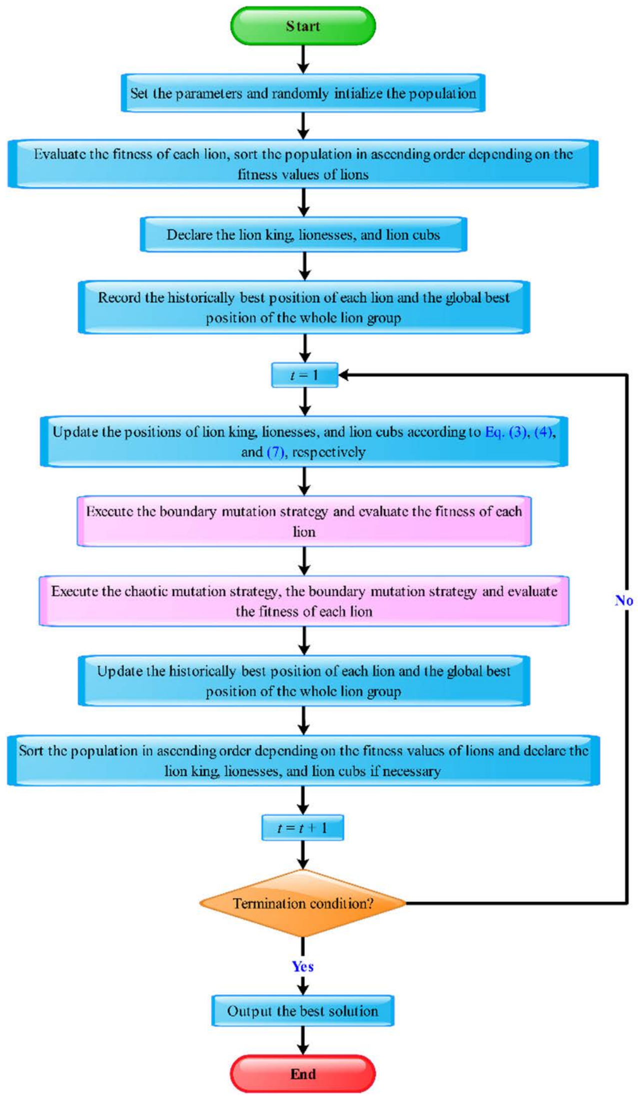


FIGURE 3. **The flowchart of the CBLSO algorithm.**

To investigate the effectiveness of the CBLSO on optimization problems, the time complexity analysis between the CBLSO and the LSO is carried out. The population size n is analyzed in the time complexity. The corresponding time complexity of each procedure in the CBLSO in terms of the worst-case of the computational time can be calculated as follows:

(1) Initializing the parameters of the CBLSO takes the time complexity $O(1)$.

(2) Initializing the population needs $O(n)$.

(3) Calculating the fitness value of all individuals requires $O(n)$.

(4) Sorting the population into a three-layered hierarchical population structure (i.e., the lion king, the lionesses and the lion cubs) costs $O(n\log n)$.

(5) The *pbest* individual and the *gbest* individual are recorded. The time complexity of this operation at most is $2O(n)$.

(6) The cooperation partner and the mother lioness followed by lion cub are selected, respectively. This process at most requires $2O(n)$.

(7) The values of the α_f, α_c, η, γ and q are determined, which costs $O(1)$.

(8) The positions of all individuals are updated, which requires $O(n)$.

(9) Justifying the boundary of population needs $O(n)$.

(10) Calculating the fitness value of all individuals requires $O(n)$.

(11) Executing the chaotic mutation strategy and the boundary mutation strategy and calculating the fitness value of all individuals again. This operation at most requires $2O(n) + O(n) + O(n)$.

(12) Updating the *pbest* individual and the *gbest* individual as well as the cooperation partner and the mother lioness followed by lion cub, which needs $2O(n)$ and $2O(n)$.

(13) Sorting the population into a three-layered hierarchical population structure (i.e., lion king, lionesses and lion cubs) again. This process costs $O(n\log n)$.

Consequently, the total time complexity of the CBLSO under the termination criterion is calculated as follows:

$$\begin{aligned} &O(1) + O(n) + O(n) + O(n\log n) + 2O(n) + 2O(n) \\ &\quad + t_{max} \\ &\quad \cdot \begin{bmatrix} O(1) + O(n) + O(n) + O(n) + 2O(n) + O(n) \\ +O(n) + 2O(n) + 2O(n) + O(n\log n) \end{bmatrix} \\ &= (t_{max} + 1) \cdot O(n\log n) + (11t_{max} + 6) \\ &\quad \cdot O(n) + (t_{max} + 1) \cdot O(1) \end{aligned} \tag{16}$$

To be simplified, the total time complexity can be regarded as $O(T \cdot n \log n) + O(t_{max} \cdot n)$.

After calculating the time complexity of the LSO, we find that it possesses the same time complexity $O(T \cdot n \log n) + O(t_{max} \cdot n)$. The results show that the time complexity of the LSO and the CBLSO is identical, indicating that the chaotic mutation strategy and the boundary mutation strategy not only improve the performance of the CBLSO but also does not decrease its computational efficiency.

IV. EXPERIMENTAL RESULTS AND ANALYSIS

In this section, a series of experimental studies are performed for rigorous assessing the performance of the CBLSO. Firstly, the related preparation works about the simulations are summarized, including the well-known classic test functions and the corresponding parameter settings. Secondly, to show the effect of the chaotic mutation strategy and the boundary mutation strategy introduced in CBLSO algorithm. The performance of the CBLSO is compared against three LSO variants: (1) the CLSO that only uses the chaotic mutation strategy and (2) the BLSO that replaces the unified boundary constraint approach with the boundary mutation strategy, while (3) the LSO uses no strategy at all, making it a standard LSO. Thirdly, the performance of the CBLSO is compared with some state-of-the-art optimization algorithms (GA [23], PSO [29], [30], ABC [31], CS [32], GSA [40], GWO [34], ASO [41], [42], and SMA [54]). Finally, in order to verify the optimization performance of the CBLSO further, more state-of-the-art algorithms are compared with CBLSO on CEC2014 test suite.

A. TEST FUNCTIONS AND EXPERIMENTAL SETUP

From the NFL theory [57], we can conclude that if we compare two searching methods or algorithms with all possible functions, the performance of any two algorithms will be, on average, the same. However, if the size of the functions used for a test is too small, it will be impossible to give a generalized and meaningful conclusion. Meanwhile, it also has a potential risk that the method is biased (optimized) towards the small size of test functions used for a test while such bias might be meaningless for other problems of interest [73]. In this study, a large enough set of test functions without any equality or inequality constraints, including 56 test functions collected from the related works of literature, are used for evaluating the performance of CBLSO. The detailed description of the set is given in the Appendix. These test functions can be classified into four categories on the basis of their functional features: Uni-modal (U), Multi-modal (M), Separable (S), and Non-separable (N). Functions F01 to F04 are US functions. Functions F05 to F17 are UN functions. Functions F18 to F27 are MS functions. Functions F28 to F56 are MN functions.

The values of the common control parameters for the CBLSO and other algorithms used for comparison are population size, *maxNFEs* or t_{max}. Different population size and *maxNFEs* or t_{max} are set as for different experiments. Moreover, to balance the exploration and the exploitation of the CBLSO, $c = 10^{-2}$ and $u = 10^{-5}$ are adopted for all experiments as in [63] and [68]. In the following subsections, the effects of the chaotic mutation strategy and the boundary mutation strategy are shown first, and then, to validate the performance of the CBLSO in a convincing way, we compare the CBLSO with some other state-of-the-art algorithms. For each experiment, the best results are identified with boldfaced through the paper.

All the experiments are implemented on the environment of Microsoft Windows 10 (64 bit) on Intel (R) Core (TM) i7-8700K CPU with 3.20 GHz 3.19 GHz and 64.0 GB (RAM). The programming software is MATLAB R2018b, and the software for Wilcoxon Signed Ranks Test (WSRT) is IBM SPSS Statistics 23.

B. EFFECTS OF THE IMPROVED STRATEGY

In this section, the advantages of the chaotic mutation strategy and boundary mutation strategy introduced in the CBLSO algorithm are investigated by a simulation experiment. Three variants of the CBLSO called CLSO, BLSO, and LSO are adopted for comparison. Due to the space limitation, only 23 test functions shown in the Appendix is tested in this section, these functions include three US functions, i.e., F01, F02, and F04, four UN functions, i.e., F13, F14, F15, and F16, three MS functions, i.e., F18, F21, and F23, thirteen MN functions, i.e., F28, F30, F34, F35, F36, F37, F38, F41, F42, F43, F44, F45, and F46. For all algorithms, to ensure the convergence rate of CBLSO, CLSO, BLSO and LSO, the population size is respectively set to be 50 for all experiments. The maximum number of iterations is fixed to 4,000.

To make a comparison, all experiments are repeated for 30 independent runs and the results, including mean best solutions (Mean) and standard deviations (Std), are all summarized in Table 1.

TABLE 1. Experimental results of 30 independent runs obtained by LSO, CLSO, BLSO, and CBLSO on 23 test functions.

Function	Metrics	LSO	CLSO	BLSO	CBLSO
F01	Mean	**0.0000E+00**	**0.0000E+00**	**0.0000E+00**	**0.0000E+00**
	Std	**0.0000E+00**	**0.0000E+00**	**0.0000E+00**	**0.0000E+00**
F02	Mean	**0.0000E+00**	**0.0000E+00**	**0.0000E+00**	**0.0000E+00**
	Std	**0.0000E+00**	**0.0000E+00**	**0.0000E+00**	**0.0000E+00**
F04	Mean	5.3878E−06	3.3087E−05	6.4666E−06	**2.7587E−06**
	Std	4.8861E−06	3.0648E−05	5.8589E−06	**2.2407E−06**
F13	Mean	**0.0000E+00**	**0.0000E+00**	**0.0000E+00**	**0.0000E+00**
	Std	**0.0000E+00**	**0.0000E+00**	**0.0000E+00**	**0.0000E+00**
F14	Mean	**0.0000E+00**	**0.0000E+00**	**0.0000E+00**	**0.0000E+00**
	Std	**0.0000E+00**	**0.0000E+00**	**0.0000E+00**	**0.0000E+00**
F15	Mean	**0.0000E+00**	**0.0000E+00**	**0.0000E+00**	**0.0000E+00**
	Std	**0.0000E+00**	**0.0000E+00**	**0.0000E+00**	**0.0000E+00**
F16	Mean	2.3477E−03	**1.4981E−03**	3.0963E−03	1.7710E−03
	Std	3.0247E−03	**1.4405E−03**	3.5549E−03	1.5095E−03
F18	Mean	0.39809	**0.39789**	0.39874	**0.39789**
	Std	4.8371E−04	3.4144E−08	2.5916E−03	**2.6179E−08**
F21	Mean	**0.0000E+00**	**0.0000E+00**	**0.0000E+00**	**0.0000E+00**
	Std	**0.0000E+00**	**0.0000E+00**	**0.0000E+00**	**0.0000E+00**
F23	Mean	−12555.0769	−12561.2726	−12561.8512	**−12565.9686**
	Std	37.5656	23.1461	25.1102	**7.3258**
F28	Mean	3.8121E−11	**0.998**	**0.998**	**0.998**
	Std	1.6218E−07	6.0559E−12	2.5583E−10	**4.2229E−12**
F30	Mean	**−1.0316**	**−1.0316**	**−1.0316**	**−1.0316**
	Std	6.4556E−10	**8.7435E−11**	2.7427E−08	4.0552E−10
F34	Mean	**3**	**3**	**3**	**3**
	Std	2.4886E−05	**1.3081E−07**	5.4345E−05	1.9048E−06
F35	Mean	0.00039585	0.00030753	0.00045092	**0.00030754**
	Std	2.9895E−04	6.4651E−08	3.7761E−04	**5.7674E−08**
F36	Mean	**−10.1532**	**−10.1532**	**−10.1532**	**−10.1532**
	Std	**1.1794E−06**	1.8906E−06	1.7410E−06	1.5725E−06
F37	Mean	**−10.4029**	**−10.4029**	**−10.4029**	**−10.4029**
	Std	1.321E−05	1.3285E−05	2.1912E−05	**1.1777E−05**
F38	Mean	**−10.5364**	**−10.5364**	**−10.5364**	**−10.5364**
	Std	1.3112E−05	**1.0617E−05**	2.4531E−05	1.8364E−05
F41	Mean	−3.8129	**−3.8627**	−3.8131	**−3.8627**
	Std	3.0271−02	7.2212E−05	4.2866E−02	**5.7317E−05**
F42	Mean	−2.7839	−3.3101	−2.8694	**−3.3143**
	Std	2.4182E−01	3.6835E−02	3.0304E−01	**3.0057E−02**
F43	Mean	**0.0000E+00**	**0.0000E+00**	**0.0000E+00**	**0.0000E+00**
	Std	**0.0000E+00**	**0.0000E+00**	**0.0000E+00**	**0.0000E+00**
F44	Mean	**8.8818E−16**	**8.8818E−16**	**8.8816E−16**	**8.8818E−16**
	Std	**0.0000E+00**	**0.0000E+00**	**0.0000E+00**	**0.0000E+00**
F45	Mean	5.1098E−03	1.5179E−04	4.9903E−03	**1.4808E−04**
	Std	6.8793E−03	1.0642E−04	4.9817E−03	**8.0745E−05**
F46	Mean	1.1856E−03	4.5598E−04	1.1357E−03	**3.9092E−04**
	Std	1.4669E−03	3.7230E−04	1.3245E−03	**3.0625E−04**

From Table 1, on the US functions, we can find that CBLSO provides the best results in terms of the Mean values and the Std values, followed by CLSO and BLSO, suggesting the positive effect of introducing the chaotic mutation strategy and the boundary mutation strategy in our CBLSO framework. In detail, all of the four algorithms have equal performance on functions F01 and F02. On F04, all algorithms cannot obtain a global optimum, but the Mean value provided by CBLSO outperforms the results of the other three algorithms, while the results obtained by the LSO and the BLSO are better than the results of the CLSO. The reason may be that F04 (Quartic function) is padded with noise and the random noise makes sure that the algorithm never gets the same value on the same point. For this function, the chaotic mutation may weaken the exploration ability of algorithm but combining the chaotic mutation and the boundary mutation can enhance the exploration ability of the algorithm. On the UN functions, we can see that on functions F13, F14, and F15, all algorithms have equal performance, whereas CBLSO cannot perform the best for F16. For this function, CLSO performs the best both in terms of the Mean and the Std. The reason may be that F16 (Rosenbrock function) is a function with a narrow curving valley. For this function, the boundary mutation is unfavorable for the algorithm to explore the space properly and keep up the direction changes, while the randomness of the chaotic system makes it more favorable for improving the exploration ability of the algorithm. However, it demonstrates that the chaotic mutation strategy and the boundary mutation strategy used in CBLSO indeed improves the performance of LSO. It also manifests that CLSO performs better than LSO and BLSO on F16 in terms of the Mean, which shows that the chaotic mutation strategy achieves better influences than the boundary mutation strategy. On the MS functions, all of the four algorithms show equal performance on F21, while CBLSO outperforms LSO, CLSO, and BLSO on all the remaining functions. On the MN functions, the results reported in Table 1 show that CBLSO performs the best for all MS problems. For all algorithms, there is no difference on functions, i.e., F30, F34, F36, F37, F38, F43, and F44 in terms of the Mean values. Besides, when we pay attention to Table 1, it is apparent that CBLSO has a good performance in comparison with the other algorithms on US functions but its performance decreases on the UN, MS, and MN functions. This is because US functions and UN functions are the easiest test problems that have no local optimum and there is only one global solution, and they are often used to examine the convergence rate of optimization algorithms. However, MS functions and MN functions usually have many or a few local optima and they have often employed the ability of the algorithms to escape from poor local optima and obtain the near-global optimum. Different algorithms have different advantages in dealing with these functions. Thus, the performance of the algorithm on solving these functions is different.

C. FIXED-ITERATION RESULTS BETWEEN CBLSO AND OTHER NATURE-INSPIRED ALGORITHMS

In this part, the CBLSO is compared with eight meta-heuristic optimization algorithms, namely, GA [23], PSO [29], [30], ABC [31], CS [32], GSA [40] and GWO [34], ASO [41], [42], and SMA [54] on all 56 functions listed in the Appendix. For all algorithms, the population size is set to 50. In GA, the crossover, mutation probabilities, and generation gap value are set to 0.8, 0.01, and 0.9, respectively [74]. In PSO, $c_1 = c_2 = 2$ and inertia weight ω is 0.9 [4]. In CS, $p_a = 0.25$ as recommended in [32]. In ABC, $limit = n \times D$ [74]. In GSA, G_0 is set to 100, α is set to 20, and$K_0$is set to$n$and is decreased linearly to 1 [40]. In GWO, $r_1 = r_2 = rand, a = 2 - 2^*t/t_{max}, A = 2^*a^*r_1 - a$, and 2^*r_2. In ASO, $\alpha = 50$ and $\beta = 0.2$ [41], [42]. In SMA, $z = 0.03$ [54].

The maximum number of iterations is set to be 6,000. For every benchmark function, all algorithms are repeated for 30 independent runs. The experimental results of Mean, Std, and standard errors of means (SEM) are presented in Tables 2-5.

Firstly, we describe the results for US functions. It is apparent from Table 2 that the CBLSO is the most competitive algorithm that provides the best Mean values for all test functions, and then they are GWO and SMA that obtain the best Mean values on 3 of 4 test functions. GA, ABC, CS, GSA, and ASO perform similarly on function F01, while PSO gives poor results in terms of this indicator. It is also shown that, on F04, all algorithms cannot give a global optimum but the results provided by CBLSO outperform the results of the other six algorithms.

Next, we analyze the simulation results for UN functions (Table 3). In this case, CBLSO and SMA are the best algorithms that provide the best Mean values on 8 of 13 test functions. PSO, CS, and GWO obtain the best Mean values on 5 of 13 test functions. Roughly, ABC and GSA produce competitive results regarding this indicator only on 3 functions, while ASO provides the best Mean values on 2 of 13 test functions. GA gives worse results compared with the other peer algorithms. It is also observed from Table 3 that CBLSO performs better than GA on all test functions. On 3 functions (F06, F07, and F14) CBLSO and PSO show equal performance but on 7 functions (F05 and F11 to F17) CBLSO outperforms PSO, while PSO outperforms CBLSO only on 3 functions (F08 to F10). Only on 1 function (F06), there is no significant difference between CBLSO and CS but on 8 functions CBLSO outperforms CS, while CS outperforms CBLSO only on 4 functions. The result between CBLSO and ABC is identical to the result between CBLSO and CS. On F14, CBLSO and GSA show equal performance but on 9 functions CBLSO is better than GSA while GSA is better than CBLSO only on 3 functions. The CBLSO algorithm and GWO show equal performance on 4 functions (F06, F07, F11, and F14) but on 8 functions CBLSO outperforms GWO, while GWO outperforms CBLSO only on 1 function. There are no functions that CBLSO and ASO show equal performance, however, CBLSO outperforms ASO on 9 functions (F06, F07, and F11 to F17), while ASO outperforms CBLSO only on 4 functions (F05, F08, F09, and F10). On functions (F07, F09, F11 to F15), CBLSO and SMA show equal performance, and on functions (F06, F10, and F16) CBLSO outperforms SMA, while SMA outperforms CBLSO only on 3 functions (F05, F08, F09, F10, and F17).

We now pay attention to MS functions (Table 4). Regarding the best Mean values, SMA is the best algorithm providing the best Mean values on 7 out of 10 functions, followed by CS, GWO, and CBLSO, which obtains the best Mean values on 5 functions. PSO and ABC can be the third-best algorithm, which performs the best Mean values on 4 functions. And then GSA, ASO, and GA are the worst best algorithm which give the best Mean values only on 3, 3, and 2 functions, respectively. For all algorithms, it is clear from the results listed in Table 4 that there is no evident difference on F25. Only on 3 functions (F23, F26, and F27) that GA outperforms CBLSO but CBLSO outperforms GA on 6 functions (F18 to F22 and F24). On 3 functions (F18, F19, and F25) CBLSO and PSO show equal performance but on 5 of the remaining 7 functions CBLSO outperforms PSO while PSO outperforms CBLSO only on 2 functions (F20 and F27). The number of functions that CBLSO and CS show equal performance is the same as the number of CBLSO and CS show equal performance, but on 4 of the remaining 7 functions CBLSO outperforms CS while CS outperforms CBLSO only on 3 functions (F20, F26, and F27). The results between CBLSO and ABC as well as the results between CBLSO and GSA are identical to the results between CBLSO and CS as well as the results between CBLSO and PSO, respectively. It is evident from the results that CBLSO and GWO show equal performance on 5 functions but 4 of the remaining 5 functions CBLSO performs better than GWO while GWO performs better than CBLSO only on 1 function (F20). On functions (F18, F19, and F25) CBLSO and ASO show equal performance, however, on functions (F21 to F24 and F26) CBLSO outperforms ASO, while on functions (F20 and F27) ASO outperforms CBLSO. On functions F18, F19, F21, F22, and F25, CBLSO and SMA can show equal performance, and CBLSO outperforms SMA on function F26, while SMA outperforms CBLSO on functions F20, F22, F23, and F27.

TABLE 2. Experimental results obtained by GA, PSO, ABC, CS, GSA, GWO, ASO, SMA, and CBLSO on US test functions.

Function		GA	PSO	ABC	CS	GSA	GWO	ASO	SMA	CBLSO
F01	Mean	**0.0000E+000** (=)	3.3333E−002 (+)	**0.0000E+000** (=)	**0.0000E+000** (=)	**0.0000E+000** (=)	**0.0000E+000** (=)	**0.0000E+000** (=)	**0.0000E+000** (=)	**0.0000E+000**
	Std	**0.0000E+000**	1.7951E−001	**0.0000E+000**	**0.0000E+000**	**0.0000E+000**	**0.0000E+000**	**0.0000E+000**	**0.0000E+000**	**0.0000E+000**
	SEM	**0.0000E+000**	1.8257E−001	**0.0000E+000**	**0.0000E+000**	**0.0000E+000**	**0.0000E+000**	**0.0000E+000**	**0.0000E+000**	**0.0000E+000**
F02	Mean	1.1995E−001 (+)	2.7980E−118 (+)	5.1185E−028 (+)	3.7243E−035 (+)	8.1403E−018 (+)	**0.0000E+000** (=)	1.5086E−026 (+)	**0.0000E+000** (=)	**0.0000E+000**
	Std	4.4828E−002	1.3640E−117	2.5786E−027	2.7528E−035	1.6344E−018	**0.0000E+000**	2.1096E−026	**0.0000E+000**	**0.0000E+000**
	SEM	1.2805E−001	1.3924E−117	2.6289E−027	4.6313E−035	8.3028E−018	**0.0000E+000**	2.5935E−026	**0.0000E+000**	**0.0000E+000**
F03	Mean	1.8335E−002 (+)	1.6736E−120 (+)	3.5101E−030 (+)	7.2206E−036 (+)	7.4250E−017 (+)	**0.0000E+000** (=)	4.8977E−025 (+)	**0.0000E+000** (=)	**0.0000E+000**
	Std	6.5971E−003	4.0971E−120	8.2097E−030	6.8698E−036	2.5358E−017	**0.0000E+000**	2.0794E−024	**0.0000E+000**	**0.0000E+000**
	SEM	1.9485E−002	4.4257E−120	8.9286E−030	9.9666E−036	7.8461E−017	**0.0000E+000**	2.1363E−024	**0.0000E+000**	**0.0000E+000**
F04	Mean	1.9875E−002 (+)	2.1160E−003 (+)	2.8609E−002 (+)	6.5289E−003 (+)	1.2969E−002 (+)	6.9173E−005 (+)	1.3580E−002 (+)	1.5959E−005 (+)	**2.2957E−006**
	Std	6.5016E−003	7.1960E−004	5.8639E−003	1.8134E−003	2.8669E−003	3.3174E−005	5.7535E−003	1.4114E−005	**1.9012E−006**
	SEM	2.0911E−002	2.2350E−003	2.9204E−002	6.7761E−003	1.3283E−002	7.6716E−005	1.4748E−002	2.1305E−005	**2.9807E−006**
+/−/=		3/0/1	4/0/0	3/0/1	3/0/1	3/0/1	1/0/3	3/0/1	1/0/3	

TABLE 3. Experimental results obtained by GA, PSO, ABC, CS, GSA, GWO, ASO, SMA, and CBLSO on UN benchmark functions.

Function		GA	PSO	ABC	CS	GSA	GWO	ASO	SMA	CBLSO
F05	Mean	2.0033E−001 (+)	5.0805E−002 (−)	2.5204E−012 (−)	**0.0000E+000** (−)	9.6271E−022 (−)	5.0515E−010 (+)	3.3979E−031 (−)	5.2059E−013 (−)	5.3139E−011
	Std	3.6944E−001	1.9009E−001	2.4626E−012	**0.0000E+000**	1.1672E−021	5.0971E−010	7.8670E−031	7.0941E−013	9.1496E−011
	SEM	4.2026E−001	1.9677E−001	3.5238E−012	**0.0000E+000**	1.5130E−021	7.1762E−010	8.5694E−031	8.7992E−013	1.0581E−010
F06	Mean	−0.5644 (+)	**−1** (−)	**−1** (−)	**−1** (−)	−0.98705 (−)	**−1** (−)	−0.068773 (+)	−0.99535 (+)	**−1**
	Std	4.9350E−001	**0.0000E+000**	**0.0000E+000**	**0.0000E+000**	6.9763E−002	1.3799E−009	2.2206E−001	1.3073E−02	2.2298E−008
	SEM	6.5825E−001	**0.0000E+000**	**0.0000E+000**	**0.0000E+000**	7.0956E−002	2.3351E−009	9.5734E−001	1.3875E−02	2.4866E−008
F07	Mean	1.9745E−003 (+)	**0.0000E+000** (=)	4.4127E−015 (+)	2.8314E−130 (+)	7.9593E−023 (+)	**0.0000E+000** (=)	7.1227E−033 (+)	**0.0000E+000** (=)	**0.0000E+000**
	Std	1.6359E−003	**0.0000E+000**	4.6895E−015	1.4367E−129	6.2243E−023	**0.0000E+000**	1.3526E−032	**0.0000E+000**	**0.0000E+000**
	SEM	2.5642E−003	**0.0000E+000**	6.4392E−015	1.4643E−129	1.0104E−022	**0.0000E+000**	1.5287E−032	**0.0000E+000**	**0.0000E+000**
F08	Mean	4.8984E+000 (+)	1.1318E−016 (−)	1.5209E−005 (−)	**0.0000E+000** (−)	7.1192E−004 (−)	2.4398E−001 (=)	1.4867E−022 (−)	1.0787E−005 (−)	6.2741E−005
	Std	5.9690E+000	1.8174E−016	2.0710E−005	**0.0000E+000**	3.8338E−003	5.4759E−001	3.0506E−022	2.4126E−005	1.1968E−004
	SEM	7.7216E+000	2.1410E−016	2.5695E−005	**0.0000E+000**	3.8993E−003	5.9949E−001	3.3936E−022	2.6428E−005	1.3513E−004
F09	Mean	−49.8717 (+)	**−50** (−)	**−50** (−)	**−50** (−)	**−50** (−)	**−50** (−)	**−50** (−)	**−50** (−)	−49.9999
	Std	6.6131E−002	2.9296E−014	3.0644E−014	3.5527E−014	3.0644E−014	5.8860E−007	**2.6395E−014**	7.7352E−006	6.5361E−005
	SEM	1.4434E−001	**1.3531E−013**	1.3729E−013	1.7053E−013	1.5217E−013	1.0029E−006	1.5739E−013	9.6689E−006	1.2723E−004
F10	Mean	−196.0927 (+)	**−210** (−)	**−210** (−)	**−210** (−)	**−210** (−)	−194.7858 (=)	**−210** (−)	−209.9992 (−)	−209.9067
	Std	7.7300E+000	4.4946E−012	4.1246E−013	**0.0000E+000**	4.5070E−013	3.3499E+001	4.2884E−013	4.3330E−004	6.7367E−002
	SEM	1.5911E+001	1.1523E−011	**8.6282E−013**	2.7285E−012	2.3775E−012	3.6792E+001	2.4629E−012	9.0276E−004	1.1509E−001
F11	Mean	1.7215E+000 (+)	6.5616E−199 (+)	5.0728E−009 (+)	4.9007E−059 (+)	1.6566E−018 (+)	**0.0000E+000** (=)	4.1373E−028 (+)	**0.0000E+000** (=)	**0.0000E+000**
	Std	1.2504E+000	**0.0000E+000**	4.1515E−009	5.5441E−059	5.3076E−019	**0.0000E+000**	3.7132E−028	**0.0000E+000**	**0.0000E+000**
	SEM	2.1277E+000	**0.0000E+000**	6.5550E−009	7.3996E−059	1.7395E−018	**0.0000E+000**	5.5592E−028	**0.0000E+000**	**0.0000E+000**
F12	Mean	5.6065E−002 (+)	6.5655E−006 (+)	1.4189E−003 (+)	1.6370E−006 (+)	4.7948E−006 (+)	3.9101E−007 (+)	2.7013E−006 (+)	**0.0000E+000** (=)	**0.0000E+000**
	Std	2.3576E−002	3.0647E−006	2.8694E−004	7.5655E−007	2.2361E−006	5.4400E−007	9.2033E−007	**0.0000E+000**	**0.0000E+000**
	SEM	6.0821E−002	7.2456E−006	1.4476E−003	1.8034E−006	5.2906E−006	6.6995E−007	2.8538E−006	**0.0000E+000**	**0.0000E+000**
F13	Mean	7.4688E+003 (+)	2.5103E+003 (+)	7.3283E+003 (+)	7.3989E+003 (+)	1.2244E+003 (+)	2.0789E+002 (+)	3.4566E+003 (+)	**0.0000E+000** (=)	**0.0000E+000**
	Std	3.8436E+003	1.1554E+003	1.3663E+003	1.6659E+003	4.3436E+002	4.2950E+002	1.1989E+003	**0.0000E+000**	**0.0000E+000**
	SEM	8.3998E+003	2.7634E+003	7.4546E+003	7.5841E+003	1.2992E+003	4.7717E+002	3.6586E+003	**0.0000E+000**	**0.0000E+000**
F14	Mean	4.3878E−001 (+)	**0.0000E+000** (=)	2.8917E−292 (+)	1.1028E−089 (+)	**0.0000E+000** (=)	**0.0000E+000** (=)	2.4182E−301 (=)	**0.0000E+000** (=)	**0.0000E+000**
	Std	3.5844E−002	**0.0000E+000**	**0.0000E+000**	4.9217E−089	**0.0000E+000**	**0.0000E+000**	**0.0000E+000**	**0.0000E+000**	**0.0000E+000**
	SEM	5.6657E−002	**0.0000E+000**	**0.0000E+000**	5.0438E−089	**0.0000E+000**	**0.0000E+000**	**0.0000E+000**	**0.0000E+000**	**0.0000E+000**
F15	Mean	1.3989E−001 (+)	2.8883E−068 (+)	6.6502E−024 (+)	1.5092E−014 (+)	1.4801E−008 (+)	9.4688E−253 (+)	9.4463E−013 (+)	**0.0000E+000** (=)	**0.0000E+000**
	Std	1.8882E−002	5.5719E−068	1.1916E−023	9.0339E−015	2.2046E−009	**0.0000E+000**	1.0089E−012	**0.0000E+000**	**0.0000E+000**
	SEM	1.4115E−001	6.2760E−068	1.3646E−023	1.7589E−014	1.4965E−008	**0.0000E+000**	1.3821E−012	**0.0000E+000**	**0.0000E+000**
F16	Mean	1.0196E+002 (+)	1.3750E+001 (+)	2.4691E+001 (+)	4.5483E+000 (+)	1.9101E+001 (+)	2.6008E+001 (+)	1.5338E+001 (+)	3.9313E−003 (+)	**1.2973E−003**
	Std	3.8712E+001	1.3961E+001	1.0733E+001	2.4774E+000	2.8623E−001	7.7327E−001	4.0408E−001	2.7648E−003	**2.3935E−003**
	SEM	1.0906E+002	1.9595E+001	2.6923E+001	5.1792E+000	1.9103E+001	2.6019E+001	1.5343E+001	4.8062E−003	**2.7224E−003**
F17	Mean	2.6010E+000 (+)	6.6667E−001 (+)	6.6667E−001 (+)	6.6667E−001 (+)	6.6667E−001 (+)	6.6667E−001 (+)	6.6667E−001 (+)	**1.3216E−002** (−)	2.1610E−001
	Std	1.4524E+000	8.5998E−017	4.0825E−008	8.6381E−008	1.3896E−008	3.6565E−009	**2.8666E−017**	6.5583E−003	2.2768E−002
	SEM	2.9791E+000	6.6667E−001	6.6667E−001	6.6667E−001	6.6667E−001	6.6667E−001	6.6667E−001	**1.4753E−002**	2.1730E−001
+/−/=		13/0/0	6/5/2	8/5/0	8/5/0	7/5/1	6/2/5	8/4/1	2/5/6	

Finally, we compare CBLSO with GA, PSO, ABC, CS, GSA, and GWO on MN functions (Table 5). Definitely, in terms of best Mean values, CS performs the best on 24 functions. Other promising algorithms are CBLSO, SMA, ASO, and ABC, obtaining the best on 16, 16, 15, and 14 functions, respectively. At last, GWO, PSO, GSA, and GA are four algorithms performing the worst results regarding this indicator which gives the best results only on 9, 7, 7, and 3 functions, respectively. By observing the results listed in Tables 5, we can conclude that all algorithms achieve success in finding the best Mean values on F30. On F28, only GA, ABC, CS, ASO, SMA, and CBLSO can reach the global optimum region and other algorithms fail. On F29, the results provided by CS, GWO, SMA, and CBLSO are better than the other three algorithms. On F31 and F32, PSO, CS, GSA, GWO, ASO, SMA, and CBLSO are superior to both GA and ABC. On F33, only PSO, CS, ASO, SMA, and CBLSO achieve success in reaching the global optimum and other algorithms fail. On F34, the performance of GA is far worse than other peer algorithms. Moreover, the same pattern occurs

TABLE 4. Experimental results obtained by GA, PSO, ABC, CS, GSA, GWO, ASO, SMA, and CBLSO on MS benchmark functions.

Function		GA	PSO	ABC	CS	GSA	GWO	ASO	SMA	CBLSO
F18	Mean	0.39822 (+)	**0.39789** (−)	**0.39789** (−)	**0.39789** (−)	**0.39789** (−)	**0.39789** (=)	**0.39789** (−)	**0.39789** (−)	**0.39789**
	Std	5.6391E−004	**0.0000E+000**	5.9024E−013	**0.0000E+000**	**0.0000E+000**	3.2201E−009	**0.0000E+000**	1.2692E−010	2.5267E−008
	SEM	6.5665E−004	**3.5773E−007**	**3.5773E−007**	**3.5773E−007**	**3.5773E−007**	3.6032E−007	**3.5773E−007**	3.5776E−007	3.6744E−007
F19	Mean	1.5325E−001 (+)	**0.0000E+000** (=)	**0.0000E+000** (=)	**0.0000E+000** (=)	**0.0000E+000** (=)	**0.0000E+000** (=)	**0.0000E+000** (=)	**0.0000E+000** (=)	**0.0000E+000**
	Std	1.5328E−001	**0.0000E+000**	**0.0000E+000**	**0.0000E+000**	**0.0000E+000**	**0.0000E+000**	**0.0000E+000**	**0.0000E+000**	**0.0000E+000**
	SEM	2.1675E−001	**0.0000E+000**	**0.0000E+000**	**0.0000E+000**	**0.0000E+000**	**0.0000E+000**	**0.0000E+000**	**0.0000E+000**	**0.0000E+000**
F20	Mean	2.2203E−003 (+)	**0.0000E+000** (−)	3.4935E−017 (−)	**0.0000E+000** (−)	2.3323E−021 (−)	8.6939E−010 (=)	1.5777E−031 (−)	8.4417E−013 (−)	1.3930E−009
	Std	2.7178E−003	**0.0000E+000**	5.2873E−017	**0.0000E+000**	2.2940E−021	9.2937E−010	3.1554E−031	1.2590E−012	2.5772E−009
	SEM	3.5094E−003	**0.0000E+000**	6.3372E−017	**0.0000E+000**	3.2714E−021	1.2726E−009	3.5279E−031	1.5158E−012	2.9296E−009
F21	Mean	6.1261E−002 (+)	3.6449E+001 (+)	1.7474E+002 (+)	2.6984E+001 (+)	1.1741E+001 (+)	**0.0000E+000** (=)	1.9966E+001 (+)	**0.0000E+000** (=)	**0.0000E+000**
	Std	1.9602E−002	1.1412E+001	1.3355E+001	5.9131E+000	2.5741E+000	**0.0000E+000**	5.0140E+000	**0.0000E+000**	**0.0000E+000**
	SEM	6.4321E−002	3.8194E+001	1.7525E+002	2.7624E+001	1.2019E+001	**0.0000E+000**	2.0586E+001	**0.0000E+000**	**0.0000E+000**
F22	Mean	6.1008E−002 (+)	4.7000E+000 (+)	1.8354E+002 (+)	2.2011E+001 (+)	1.8388E+001 (+)	**0.0000E+000** (=)	2.7967E+001 (+)	**0.0000E+000** (=)	**0.0000E+000**
	Std	2.2589E−002	4.5982E+000	1.6190E+002	2.7284E+000	4.3250E+000	**0.0000E+000**	7.8845E+000	**0.0000E+000**	**0.0000E+000**
	SEM	6.5056E−002	6.5752E+000	1.3138E+001	2.2180E+001	1.8890E+001	**0.0000E+000**	2.9057E+001	**0.0000E+000**	**0.0000E+000**
F23	Mean	−12569.1321 (=)	−7101.717 (+)	−6469.6355 (+)	−10628.3791 (+)	−2704.2987 (+)	−6468.9457 (+)	−7463.0405 (+)	**−12569.4861** (−)	−12562.8834
	Std	1.3425E−001	6.1536E+002	3.3860E+002	2.1770E+002	4.4836E+002	5.7428E+002	6.3980E+002	**3.7911E−004**	2.4517E+001
	SEM	3.9167E−001	5.5023E+003	6.1093E+003	1.9533E+003	9.8754E+003	6.1275E+003	5.1464E+003	**1.3908E−002**	2.5394E+001
F24	Mean	3.8546E−001 (+)	5.4593E+003 (+)	6.3319E+003 (+)	2.0276E+003 (+)	9.9001E+003 (+)	6.2288E+003 (+)	4.9754E+003 (+)	**5.7605E−004** (−)	3.4081E−001
	Std	1.7629E−001	7.0274E+002	3.7671E+002	2.4644E+002	4.2672E+002	7.1331E+002	5.9839E+002	**4.1107E−004**	9.0185E−001
	SEM	4.2386E−001	5.5043E+003	6.3431E+003	2.0425E+003	9.9093E+003	6.2695E+003	5.0113E+003	**7.0768E−004**	9.6409E−001
F25	Mean	**−1.8013** (+)	**−1.8013** (−)	**-1.8013** (−)	**−1.8013** (−)	**−1.8013** (−)	**−1.8013** (=)	**−1.8013** (−)	**−1.8013** (−)	**−1.8013**
	Std	6.3960E−005	**8.8818E−016**	**8.8818E−016**	**8.8818E−016**	**8.8818E−016**	8.2762E−009	**8.8818E−016**	1.0372E−011	2.6672E−008
	SEM	7.3242E−005	3.4101E−006	3.4101E−006	3.4101E−006	3.4101E−006	3.4008E−006	3.4101E−006	3.4101E−006	**3.3981E−006**
F26	Mean	−4.6873 (−)	−4.5417 (+)	**−4.6877** (−)	**−4.6877** (−)	−4.5695 (+)	−4.4107 (+)	−4.5450 (+)	−4.6228 (=)	−4.6283
	Std	4.5923E−004	1.0672E−001	2.1143E−015	**1.8130E−015**	7.4435E−002	3.4949E−001	7.7494E−002	1.4477E−002	6.5202E−002
	SEM	5.9070E−004	1.8085E−001	**1.7909E−007**	**1.7909E−007**	1.3968E−001	4.4590E−001	1.6235E−001	1.5862E−002	8.8200E−002
F27	Mean	**−9.6577** (−)	−9.0577 (−)	−9.6241 (−)	−9.6432 (−)	−9.2857 (−)	−8.1586 (=)	−9.0184 (−)	−8.8822 (−)	−8.2197
	Std	**2.3394E−003**	3.1848E−001	6.5841E−002	2.0432E−001	2.1364E−001	7.2776E−001	2.5701E−001	5.1944E−001	6.1022E−001
	SEM	**3.3677E−003**	6.8149E−001	7.5050E−002	2.6565E−002	4.3111E−001	1.6686E+000	6.9126E−001	9.3540E−001	1.5644E+000
+/−/=		7/2/1	5/4/1	4/5/1	4/5/1	5/4/1	3/0/7	5/4/1	0/6/4	

for GSA on F41. All algorithms could not find the global optimum on F35, F39, F40, F46, F52, F53, and F56, but results provided by CS are better than other algorithms. The reason is that the Lévy flight employed by CS to generate exemplars for an evolving population can effectively enhance the exploration strength and local optimum avoidance ability of the algorithm. Besides, the same pattern occurs for SMA and CBLSO on F44, ABC on F45, CS and ABC on F49, ASO on F50 and F55, and PSO and CS on F54. On F36, ABC and CS are two best algorithms in terms of the best Mean values. On F37, the worst algorithms are GA and PSO, while on F38, the worst algorithms are GA, PSO, and GSA. On F42, ABC, CS, GSA, and ASO are four algorithms providing the best Mean values, while ABC, CS, ASO, SMA, and CBLSO are five competitive algorithms obtaining the best Mean values on F42, F48, and F51. On F47, the results provided by ABC, GWO, SMA, and CBLSO are obviously better than other peer algorithms.

To intuitively analysis the performance of the CBLSO algorithm for solving Uni-modal problems and Multi-modal problems, the convergence progress of the average best-so-far solutions over 30 independent runs are plotted in Figs. 4, 6, 8, and 10, while the box-and-whisker diagrams of obtained optimal solutions over 30 independent runs are depicted in Figs. 5, 7, 9, and 11. Due to the space limitation, some representative curves of them are selected for illustration.

We first take a look at the convergence process graphs of average best-so-far solutions (Figs. 4, 6, 8, and 10). For Uni-modal problems, which usually have no local optimums, the convergence rate of the algorithms is more meaningful than the final optimization results, because there are some other methods or algorithms that designed for optimizing Uni-modal problems, specifically. From Fig. 4 plotted on US problems (including F02, F03, and F04), we can see that CBLSO converges quickly thorough the whole stage of search process towards the best solution, while other eight algorithms converge slowly and finally traps into the local minimum in the late stage of search process, indicating that CBLSO possesses superior performance on such optimization problems. Different from the convergence characteristic shown in Fig. 4, according to Fig. 6 plotted on UN problems (including F06, F08, F10, F12, F13, and F16), it is observed that the convergence speed in the early stage of search process of CBLSO is not remarkable among all the nine algorithms, but it finally obtains the global optimal minimum in the late stage of search process, implying that CBLSO can effectively avoid a premature convergence and significantly improve the accuracy of solution. Thus, we can conclude that the chaotic mutation strategy and the boundary mutation strategy used in CBLSO indeed improve the algorithm's exploration and exploitation abilities. For multi-modal problems, which usually have many local optimums and almost are often difficult to be optimized, the final optimization results are usually more significant since the optimization results are the standard to measure the exploration and exploitation abilities of the algorithm in getting rid of local optimums and finding a better solution in a short time. From Fig. 8 depicted on MS problems (including F21, F22, F23, F24, F26, and F27),

TABLE 5. Experimental results obtained by GA, PSO, ABC, CS, GSA, GWO, ASO, SMA, and CBLSO on MN benchmark functions.

Function		GA	PSO	ABC	CS	GSA	GWO	ASO	SMA	CBLSO
	Mean	**0.998** (+)	2.5425 (+)	**0.998** (+)	**0.998** (−)	1.4493 (+)	2.2442 (+)	**0.998** (−)	**0.998** (−)	**0.998**
F28	Std	4.4928E−007	2.3316E+000	1.2259E−010	**0.0000E+000**	5.1539E−001	2.4465E+000	**0.0000E+000**	4.8223E−016	2.8564E−012
	SEM	4.5222E−007	2.7968E+000	**1.6212E−007**	1.6221E−007	6.8508E−001	2.7456E+000	1.6221E−007	1.6221E−007	1.6220E−007
	Mean	1.6717E−002 (+)	1.9432E−003 (+)	1.1211E−004 (+)	**0.0000E+000** (=)	1.1757E−002 (+)	**0.0000E+000** (=)	6.4773E−004 (=)	**0.0000E+000** (=)	**0.0000E+000**
F29	Std	1.6355E−002	3.8864E−003	1.1388E−004	**0.0000E+000**	6.8497E−003	**0.0000E+000**	2.4236E−003	**0.0000E+000**	**0.0000E+000**
	SEM	2.3387E−002	4.3451E−003	1.5980E−004	**0.0000E+000**	1.3607E−002	**0.0000E+000**	2.5086E−003	**0.0000E+000**	**0.0000E+000**
	Mean	**−1.0316** (+)	**−1.0316** (−)	**−1.0316** (−)	**−1.0316** (−)	**−1.0316** (−)	**−1.0316** (=)	**−1.0316** (−)	**−1.0316** (−)	**−1.0316**
F30	Std	8.7199E−005	**4.4409E−016**	**4.4409E−016**	**4.4409E−016**	**4.4409E−016**	3.9376E−011	4.4409E−016	7.4043E−014	1.1551E−010
	SEM	1.1503E−004	**4.6510E−008**	**4.6510E−008**	**4.6510E−008**	**4.6510E−008**	4.6555E−008	4.6510E−008	4.6510E−008	4.6590E−008
	Mean	1.6197E−001 (+)	**0.0000E+000** (=)	3.7007E−018 (=)	**0.0000E+000** (=)	**0.0000E+000** (=)	**0.0000E+000** (=)	**0.0000E+000** (=)	**0.0000E+000** (=)	**0.0000E+000**
F31	Std	1.0121E−001	**0.0000E+000**	1.3847E−017	**0.0000E+000**	**0.0000E+000**	**0.0000E+000**	**0.0000E+000**	**0.0000E+000**	**0.0000E+000**
	SEM	1.9099E−001	**0.0000E+000**	1.4333E−017	**0.0000E+000**	**0.0000E+000**	**0.0000E+000**	**0.0000E+000**	**0.0000E+000**	**0.0000E+000**
	Mean	9.9344E−002 (+)	**0.0000E+000** (=)	2.5274E−012 (+)	**0.0000E+000** (=)	**0.0000E+000** (=)	**0.0000E+000** (=)	**0.0000E+000** (=)	**0.0000E+000** (=)	**0.0000E+000**
F32	Std	7.7771E−002	**0.0000E+000**	2.5450E−012	**0.0000E+000**	**0.0000E+000**	**0.0000E+000**	**0.0000E+000**	**0.0000E+000**	**0.0000E+000**
	SEM	1.2617E−001	**0.0000E+000**	3.5868E−012	**0.0000E+000**	**0.0000E+000**	**0.0000E+000**	**0.0000E+000**	**0.0000E+000**	**0.0000E+000**
	Mean	−186.7214 (+)	**−186.7309** (−)	−186.7306 (+)	**−186.7309** (−)	−186.3725 (+)	−186.7305 (=)	**−186.7309** (−)	**−186.7309** (−)	**−186.7309**
F33	Std	1.1589E−002	3.7057E−014	3.3261E−004	**1.7210E−014**	3.5348E−001	1.3079E−003	3.1988E−014	1.1595E−008	3.7438E−006
	SEM	1.4967E−002	8.8310E−006	4.4141E−004	8.8310E−006	5.0342E−001	1.3606E−003	8.8310E−006	8.8267E−006	**7.8155E−006**
	Mean	5.8642 (+)	**3** (−)	**3** (−)	**3** (−)	**3** (−)	**3** (+)	**3** (−)	**3** (−)	**3**
F34	Std	8.6177E+000	2.0431E−015	2.0431E−015	**3.5342E−016**	9.4901E−016	7.9312E−008	1.2871E−015	1.2827E−014	3.4759E−007
	SEM	9.0812E+000	7.8504E−014	**7.5555E−014**	8.0543E−014	7.7913E−014	1.3116E−007	7.8339E−014	4.8190E−014	3.6019E−007
	Mean	0.0096435 (+)	0.0011781 (=)	0.00060429 (+)	**0.00030749** (−)	0.001064 (+)	0.0023514 (=)	0.00059419 (+)	0.00032443 (+)	0.0003075
F35	Std	1.6954E−002	3.5915E−003	6.3233E−005	**1.6383E−019**	2.1710E−004	6.0063E−003	1.4056E−004	5.8244E−005	1.2551E−008
	SEM	1.9355E−002	3.6955E−003	3.0347E−004	**1.1219E−010**	7.8701E−004	6.3445E−003	3.1931E−004	6.0658E−005	1.9217E−008
	Mean	−5.302 (+)	−5.9006 (+)	**−10.1532** (−)	**−10.1532** (−)	−7.8838 (+)	−9.6449 (+)	−9.9848 (−)	**−10.1532** (+)	**−10.1532**
F36	Std	3.0898E+000	3.3572E+000	**7.1054E−015**	**7.1054E−015**	2.6202E+000	1.5249E+000	9.0694E−001	1.3612E−006	7.0982E−007
	SEM	5.7516E+000	5.4181E+000	**3.2094E−007**	**3.2094E−007**	3.4664E+000	1.6073E+000	9.2244E−001	2.3649E−006	1.3269E−006
	Mean	−5.0891 (+)	−7.8028 (=)	**−10.4029** (−)	**−10.4029** (−)	**−10.4029** (−)	**−10.4029** (−)	**−10.4029** (−)	**−10.4029** (−)	**−10.4029**
F37	Std	3.0544E+000	3.4855E+000	**0.0000E+000**	3.2432E−016	1.0256E−015	1.2367E−006	3.2432E−016	1.8486E−006	1.2053E−005
	SEM	6.1291E+000	4.3485E+000	**2.4663E−005**	**2.4663E−005**	**2.4663E−005**	2.7795E−005	**2.4663E−005**	2.6978E−005	3.8766E−005
	Mean	−5.2716 (+)	−7.396 (+)	**−10.5364** (−)	**−10.5364** (−)	−7.7861 (+)	**−10.5364** (−)	**−10.5364** (−)	**−10.5364** (−)	**−10.5364**
F38	Std	3.0269E+000	3.4109E+000	3.9721E−015	**3.5527E−015**	1.8440E+000	1.4856E−006	4.2903E−015	1.2914E−006	7.2077E−006
	SEM	6.0729E+000	4.6364E+000	3.3153E−005	3.3153E−005	3.3113E+000	3.0662E−005	3.3153E−005	3.1242E−005	**2.6616E−005**
	Mean	1.1445E+001 (+)	5.4394E−002 (+)	2.6239E−002 (+)	**1.7766E−003** (=)	1.9889E+000 (+)	2.2619E−001 (=)	3.5108E−002 (+)	3.3471E−002 (=)	2.0576E−003
F39	Std	1.5580E+001	9.5554E−002	1.8339E−002	2.2342E−003	1.5629E+000	5.5216E−001	9.5036E−002	8.2367E−002	**1.7234E−003**
	SEM	1.9332E+001	1.0995E−001	3.2012E−002	2.8545E−003	2.5295E+000	5.9669E−001	1.0131E−001	8.8908E−002	**2.6840E−003**
	Mean	8.2427E−001 (+)	1.8532E−004 (−)	2.7513E−003 (+)	**4.9219E−005** (−)	1.5796E−002 (+)	5.2936E−002 (=)	1.4655E−004 (−)	2.0318E−004 (−)	9.4025E−004
F40	Std	1.0245E+000	1.7726E−004	2.1419E−003	**5.6481E−005**	1.1542E−002	1.9494E−001	1.7307E−004	1.6169E−004	1.5109E−003
	SEM	1.3149E+000	2.5645E−004	3.4868E−003	**7.4917E−005**	1.9564E−002	2.0200E−001	2.2678E−004	2.5966E−004	1.7795E−003
	Mean	**−3.8628** (=)	**−3.8628** (−)	**−3.8628** (−)	**−3.8628** (−)	**−3.8628** (−)	−3.8625 (−)	**−3.8628** (−)	**−3.8628** (−)	**−3.8628**
F41	Std	9.8200E−006	2.9823E−015	2.9098E−015	**2.8654E−015**	3.1086E−015	1.3853E−003	3.0810E−015	1.8874E−010	2.5370E−005
	SEM	1.2550E−005	**2.1267E−007**	**2.1267E−007**	**2.1267E−007**	**2.1267E−007**	1.4091E−003	**2.1267E−007**	**2.1267E−007**	3.0136E−005
	Mean	−3.2628 (=)	−3.2906 (=)	**−3.3224** (−)	**−3.3224** (−)	**−3.3224** (−)	−3.2620 (=)	**−3.3224** (−)	−3.2230 (+)	−3.3063
F42	Std	5.9605E−002	5.2715E−002	8.5422E−016	**8.4260E−016**	8.8818E−016	6.6566E−002	8.6569E−016	4.4425E−002	4.0848E−002
	SEM	8.4303E−002	6.1559E−002	**1.9886E−006**	**1.9886E−006**	**1.9886E−006**	8.8987E−002	**1.9886E−006**	1.0882E−001	4.3909E−002
	Mean	1.7221E−001 (+)	1.6225E−002 (+)	**0.0000E+000** (=)	**0.0000E+000** (=)	3.2858E−004 (=)	2.5049E−004 (=)	**0.0000E+000** (=)	**0.0000E+000** (=)	**0.0000E+000**
F43	Std	4.3531E−002	1.6570E−002	**0.0000E+000**	**0.0000E+000**	1.7694E−003	1.3489E−003	**0.0000E+000**	**0.0000E+000**	**0.0000E+000**
	SEM	1.7763E−001	2.3191E−002	**0.0000E+000**	**0.0000E+000**	1.7997E−003	1.3720E−003	**0.0000E+000**	**0.0000E+000**	**0.0000E+000**
	Mean	9.9868E−002 (+)	4.4681E−002 (+)	7.6383E−015 (+)	1.6517E−010 (+)	2.2949E−009 (+)	7.7568E−015 (+)	8.4140E−014 (+)	**8.8818E−016** (=)	**8.8818E−016**
F44	Std	1.9674E−002	2.4061E−001	1.4062E−015	7.3517E−010	2.9784E−010	8.8620E−016	4.8985E−014	**0.0000E+000**	**0.0000E+000**
	SEM	1.0179E−001	2.4473E−001	7.7667E−015	7.5349E−010	2.3141E−009	7.8072E−015	9.7361E−014	**8.8818E−016**	**8.8818E−016**
	Mean	1.8041E−003 (+)	9.1147E−002 (=)	**1.5306E−022** (−)	6.4272E−018 (−)	3.0426E−019 (−)	9.6024E−002 (+)	2.8504E−028 (−)	4.4584E−005 (−)	2.0642E−004
F45	Std	6.3112E−004	1.8958E−001	**3.0328E−022**	2.9307E−017	8.4230E−020	1.8938E−002	2.4388E−028	7.6939E−005	8.0646E−004
	SEM	1.9113E−003	2.1035E−001	**3.3972E−022**	3.0004E−017	3.1570E−019	9.7873E−002	3.7514E−028	8.8923E−005	8.3246E−004
	Mean	9.8688E−003 (+)	4.3949E−003 (=)	1.7829E−026 (−)	**4.8668E−032** (−)	8.8745E−019 (−)	2.7011E−001 (+)	1.1910E−027 (−)	1.1022E−005 (−)	2.5358E−004
F46	Std	5.4956E−003	8.7898E−003	8.9899E−026	**1.7716E−032**	2.5469E−019	1.6571E−001	1.9502E−027	6.8267E−006	2.3252E−004
	SEM	1.1296E−002	9.8273E−003	9.1650E−026	**5.1792E−032**	9.2327E−019	3.1689E−001	2.2851E−027	1.2965E−005	3.4404E−004
	Mean	8.0292E−001 (+)	8.2465E−001 (+)	**0.0000E+000** (=)	3.7493E−002 (+)	2.0184E−004 (+)	**0.0000E+000** (=)	6.5815E−012 (+)	**0.0000E+000** (=)	**0.0000E+000**
F47	Std	1.0749E−001	7.9896E−001	**0.0000E+000**	1.4595E−002	2.0031E−005	**0.0000E+000**	9.7676E−012	**0.0000E+000**	**0.0000E+000**
	SEM	8.1008E−001	1.1482E+000	**0.0000E+000**	4.0233E−002	2.0284E−004	**0.0000E+000**	1.1778E−011	**0.0000E+000**	**0.0000E+000**
	Mean	−1.0333 (+)	−1.0674 (−)	**−1.0809** (−)	**−1.0809** (−)	−1.0784 (+)	**−1.0809** (−)	**−1.0809** (−)	**−1.0809** (−)	**−1.0809**
F48	Std	1.1564E−001	4.0593E−002	**3.1402E−016**	4.4409E−016	2.6398E−003	2.2184E−009	4.4409E−016	9.4594E−012	7.5439E−009
	SEM	1.2507E−001	4.2777E−002	3.8442E−005	3.8442E−005	3.6090E−003	3.8441E−005	3.8442E−005	3.8442E−005	**3.8436E−005**
	Mean	−0.72303 (+)	−1.1237 (+)	**−1.5** (−)	**−1.5** (−)	−0.82997 (+)	−1.0967 (+)	−1.4268 (−)	−0.53983 (+)	−1.4812
F49	Std	2.9729E−001	3.2473E−001	**6.6613E−016**	**6.6613E−016**	1.0504E−001	2.6887E−001	1.8693E−001	4.7999E−001	1.0123E−001
	SEM	8.3190E−001	4.9704E−001	**7.7665E−007**	**7.7665E−007**	6.7821E−001	4.8469E−001	4.9817E−001	6.4122E−001	1.0296E−001
	Mean	−0.35474 (+)	−0.58783 (=)	−1.4893 (−)	−1.001 (−)	−0.24423 (+)	−0.59633 (=)	**−0.63802** (=)	−0.00031006 (+)	−0.53132
F50	Std	1.6705E−001	2.6764E−001	**1.2893E−002**	2.9541E−001	1.3174E−001	2.8286E−001	2.9399E−001	1.0390E−003	1.9078E−001
	SEM	6.3271E−001	**4.6249E−001**	5.2449E−001	2.9759E−001	7.3271E−001	4.6468E−001	9.1074E−001	1.4997E+000	4.7379E−001
	Mean	−10.3351 (+)	−8.9162 (+)	**−12.1190** (−)	**−12.1190** (−)	−11.9574 (+)	−9.1957 (+)	**−12.1190** (−)	**−12.1190** (−)	**−12.1190**
F51	Std	2.9660E+000	3.2091E+000	1.5515E−012	6.8951E−015	1.5910E−001	3.1251E+000	**6.8415E−015**	3.8891E−010	5.2666E−007
	SEM	3.4612E+000	4.5340E+000	**9.1620E−005**	**9.1620E−005**	2.2687E−001	4.2793E+000	**9.1620E−005**	9.1621E−005	9.2016E−005
	Mean	−3.2418 (+)	−2.3754 (+)	−3.6525 (=)	**−8.0712** (−)	−2.3545 (+)	−3.1816 (=)	−6.1788 (=)	−2.7824 (=)	−4.5640

TABLE 5. ***(Continued.)*** **Experimental results obtained by GA, PSO, ABC, CS, GSA, GWO, ASO, SMA, and CBLSO on MN benchmark functions.**

F52	Std	2.4584E+000	5.4582E−001	9.8587E−001	3.3024E+000	6.9956E−001	**3.9404E−001**	3.7124E+000	4.8892E−001	3.2123E+000
	SEM	7.5738E+000	8.0487E+000	6.8247E+000	**4.0442E+000**	8.0814E+000	7.2347E+000	5.6256E+000	7.6388E+000	6.6666E+000
F53	Mean	−1.4837 (−)	−1.4701 (−)	−1.4774 (−)	**−7.8114** (−)	−1.4774 (−)	−1.8397 (−)	−1.4774 (−)	−0.36296 (+)	−1.6967
	Std	1.0162E−001	3.9269E−002	6.5994E−016	3.4021E+000	8.8354E−016	1.5872E+000	**2.4324E−016**	7.1226E−003	1.0035E+000
	SEM	8.7257E+000	8.7388E+000	8.7314E+000	**4.1619E+000**	8.7314E+000	8.5183E+000	8.7314E+000	9.8458	8.5710E+000
F54	Mean	1.2061E−002 (+)	**0.0000E+000** (−)	1.2622E−014 (−)	**0.0000E+000** (−)	2.0477E−018 (−)	1.9449E−004 (−)	1.1309E−027 (−)	5.9862E−011 (−)	2.3510E+001
	Std	1.3686E−002	**0.0000E+000**	1.5307E−014	**0.0000E+000**	1.7257E−018	8.4069E−004	2.6267E−027	1.1434E−010	1.2661E+002
	SEM	1.8242E−002	**0.0000E+000**	1.9839E−014	**0.0000E+000**	2.6779E−018	8.6289E−004	2.8598E−027	1.2907E−010	1.2877E+002
F55	Mean	1.5090E+002 (−)	5.4882E+002 (=)	6.7275E−005 (−)	1.6718E−018 (−)	8.0111E−001 (−)	1.1378E+002 (=)	**7.3266E−020** (−)	9.7594E+001 (=)	3.2264E+002
	Std	5.1591E+002	9.9014E+002	3.0311E−004	8.2351E−018	2.4294E+000	1.0859E+002	**2.0049E−019**	1.1063E+002	6.6015E+002
	SEM	5.3752E+002	1.1321E+003	3.1049E−004	8.4031E−018	2.5581E+000	1.5728E+002	**2.1346E−019**	1.4752E+002	7.3477E+002
F56	Mean	8.0910E+002 (−)	1.9777E+003 (−)	2.8225E+001 (−)	**1.1001E+000** (−)	3.5735E+003 (−)	2.3010E+003 (−)	2.0558E+002 (−)	1.3228E+003 (−)	1.2289E+004
	Std	1.4668E+003	2.7065E+003	3.9108E+001	**2.3165E+000**	5.5238E+003	2.4798E+003	9.2845E+002	1.6073E+003	9.0234E+003
	SEM	1.6751E+003	3.3521E+003	4.8230E+001	**2.5644E+000**	6.5790E+003	3.3829E+003	9.5094E+002	2.0816E+003	1.5246E+004
+/−/=		24/3/2	11/9/9	8/17/4	2/22/5	15/11/3	8/7/14	4/19/6	6/14/9	

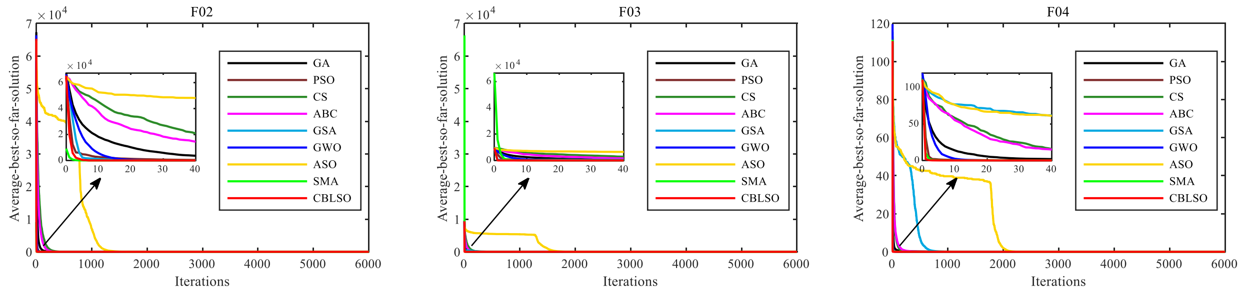


FIGURE 4. **The convergence graphs of average best-so-far solutions obtained by CBLSO and other nature-inspired algorithms on US problems.**

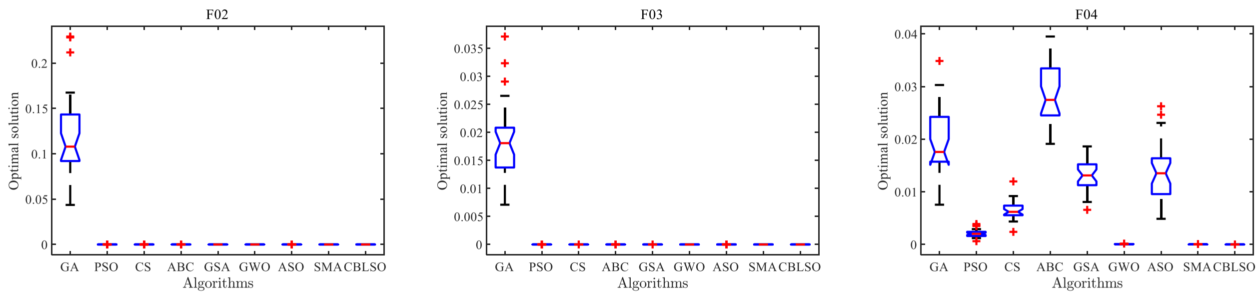


FIGURE 5. **The box plots of optimal solutions obtained by CBLSO and other nature-inspired algorithms on US problems.**

we can see that the convergence speed of CBLSO is superior on F21, F22, F23, and F24. GA and PSO defeat CBLSO and other algorithms on F26 and F27 in the early stage of the search process but the solutions obtained by CBLSO are better than or similar to the solutions obtained by GA and PSO. And then according to Fig. 10 depicted on MN problems (including F29, F36, F42, F43, F45, F47, F49, F51, and F55), CBLSO defeats other eight algorithms in case of F36, F45, and F47 by providing very fast convergence speed and finally offers the best solutions. Although the convergence speed offered by CBLSO for other functions is not remarkable among all the nine algorithms, the solutions obtained by CBLSO are comparable.

We now analysis the box plots of optimal solutions obtained by each algorithm (Figs. 5, 7, 9, and 11). From Fig. 5 plotted on US problems (including F02, F03, and F04), it is observed that CBLSO maintains the fewer values and the shorter distribution of optimal solutions compared with the other peer algorithms on all problems, indicating its great superiority for solving US problems. Next, it can be seen from Fig. 7 plotted on UN problems (including F06, F08, F10, F12, F13, and F16) that the performance of CBLSO is satisfactory for all problems, especially on F16, CBLSO can provide a better optimal solution than other peer algorithms. Thus, based on the aforementioned studies, we can conclude that CBLSO is a more effective algorithm for optimizing Uni-modal problems. According to Fig. 9 on MS problems (including F21, F22, F23, F24, F26, and F27), we can find that CBLSO maintains the fewer values and the shorter distribution of optimal solutions compared with the other nine algorithms on F21, F22, F23, and F24. GA, CS, and ABC defeat CBLSO, PSO, GSA, GWO, ASO, and SMA in the case of F26 while GA defeats other eight algorithms in the case of F27. From Fig. 11 depicted on MN problems (including F29,

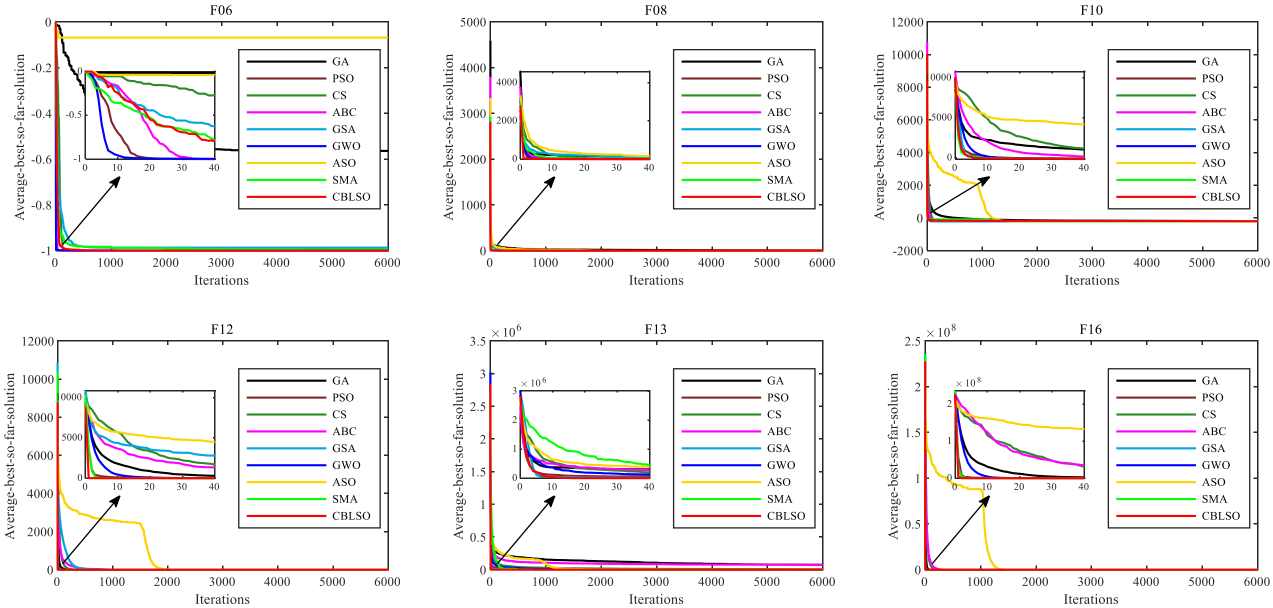


FIGURE 6. The convergence graphs of average best-so-far solutions obtained by CBLSO and other nature-inspired algorithms on UN problems.

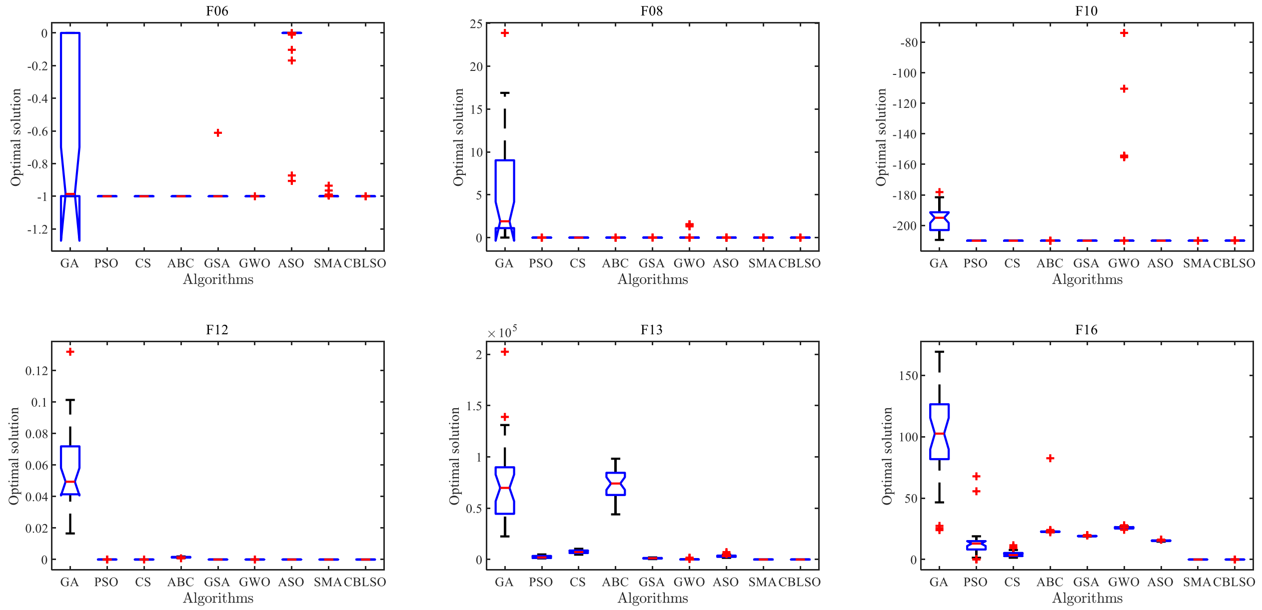


FIGURE 7. The box plots of solutions obtained by CBLSO and other nature-inspired algorithms on UN problems.

F36, F42, F43, F45, F47, F49, F51, and F55), the complexity of the test problems is enhanced. We can find that CBLSO is superior on F29, F36, F42, F43, F45, F47, F49, and F51 whereas its performance is comparable to GA, CS, ABC, GSA, ASO, and SMA on F55, denoting a good and steady performance of CBLSO.

D. ROBUSTNESS ANALYSIS BETWEEN CBLSO AND OTHER NATURE-INSPIRED ALGORITHMS

The *maxNFEs* is set as 500,000 and other parameters are set to be the same as Section C. A trial is considered to be "successful" if the following inequality is satisfied [75]:

$$|\mathrm{FOBJ}_{\mathrm{BEST}} - \mathrm{FOBJ}_{\mathrm{ANAL}}| < \varepsilon_{\mathrm{rel}} \, |\mathrm{FOBJ}_{\mathrm{ANAL}}| + \varepsilon_{\mathrm{abs}} \tag{17}$$

where $\mathrm{FOBJ}_{\mathrm{ANAL}}$ indicates the known analytical minima, $\mathrm{FOBJ}_{\mathrm{BEST}}$ indicates the best function value provided by the method or algorithm. Control parameters of accuracy are considered to be $\varepsilon_{\mathrm{rel}} = 10^{-4}$ and $\varepsilon_{\mathrm{abc}} = 10^{-6}$ in all cases for the present work.

To compare the convergence speeds, we use the acceleration rate (AR) [76] which is defined as follows, based on the

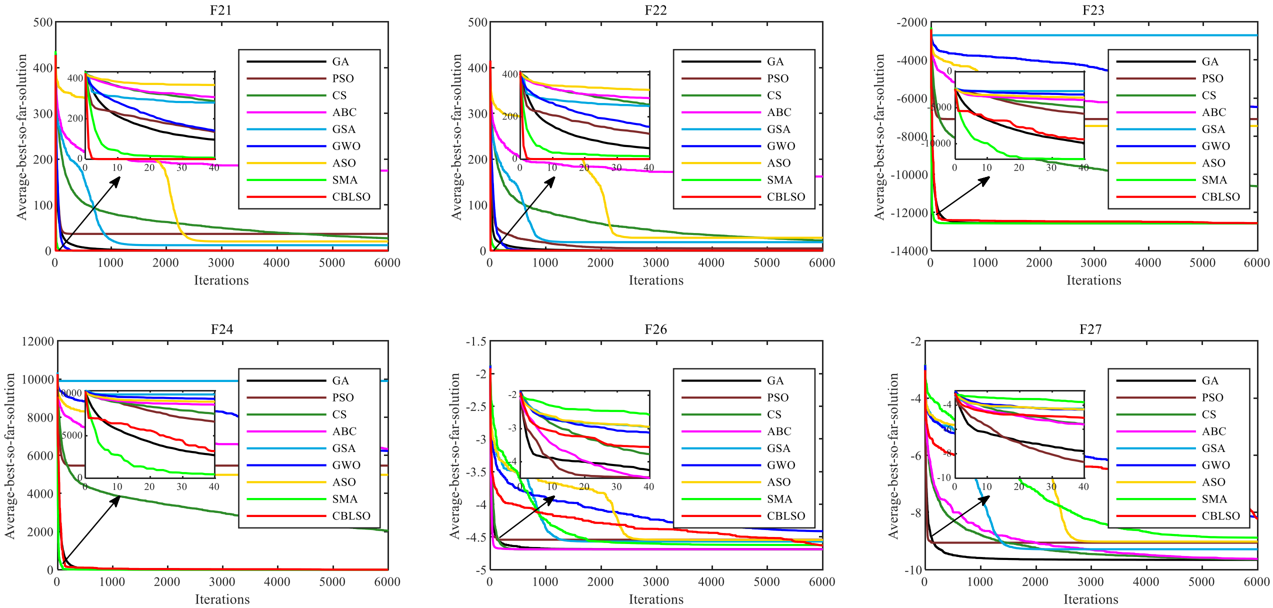


FIGURE 8. The convergence graphs of average best-so-far solutions obtained by CBLSO and other nature-inspired algorithms on MS problems.

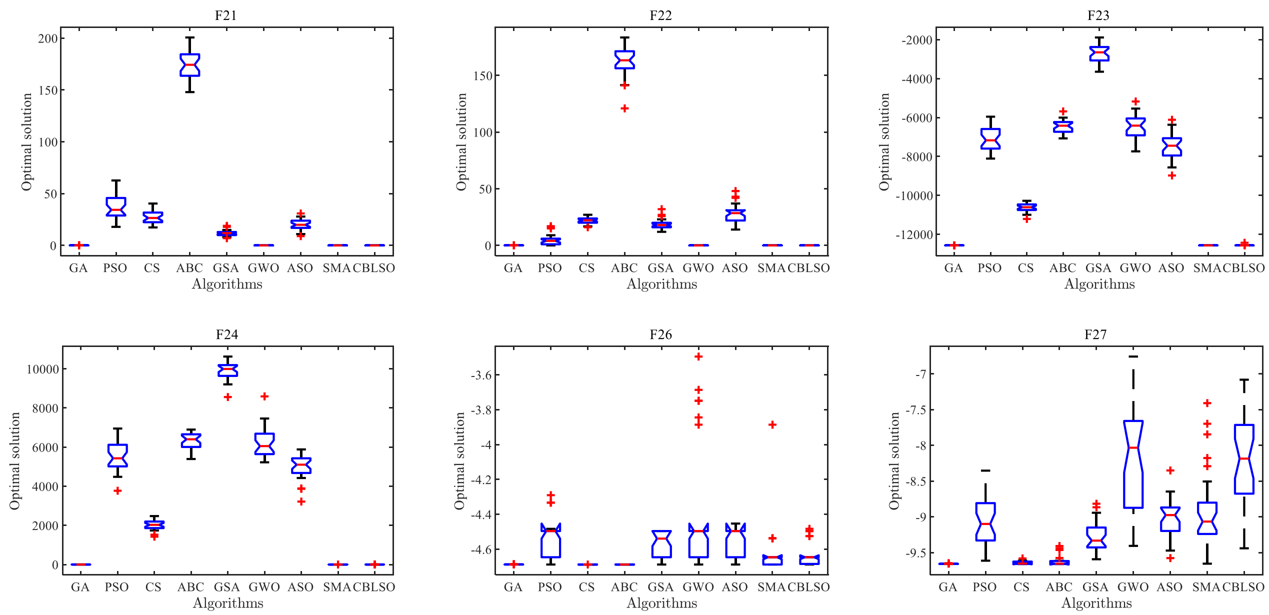


FIGURE 9. The box plots of optimal solutions obtained by CBLSO and other nature-inspired algorithms on MS problems.

number of function evaluations (NFEs) for the two algorithms CBLSO algorithm and one comparative algorithm (OCA):

$$AR = \frac{NFE_{OCA}}{NFE_{cbLSO}} \tag{18}$$

where AR > 1 means CBLSO is faster. Note that, to minimize the effect of the stochastic nature of the algorithms on this metric, the reported NFEs for each problem is the mean value over 50 runs.

The number of runs, for which the algorithm successfully reaches the accuracy of each test function is measured as the success rate (SR):

$$SR = \frac{runs_{Success}}{runs_{Total}} \tag{19}$$

where $runs_{Success}$ represents the number of runs that success, $runs_{Total}$ represents the total number of the independent runs.

Also, the average acceleration rate (AR_{ave}) and the average success rate (SR_{ave}) over m test functions are calculated as follows:

$$AR_{ave} = \frac{1}{m}\sum_{i=1}^{m} AR_i \tag{20}$$

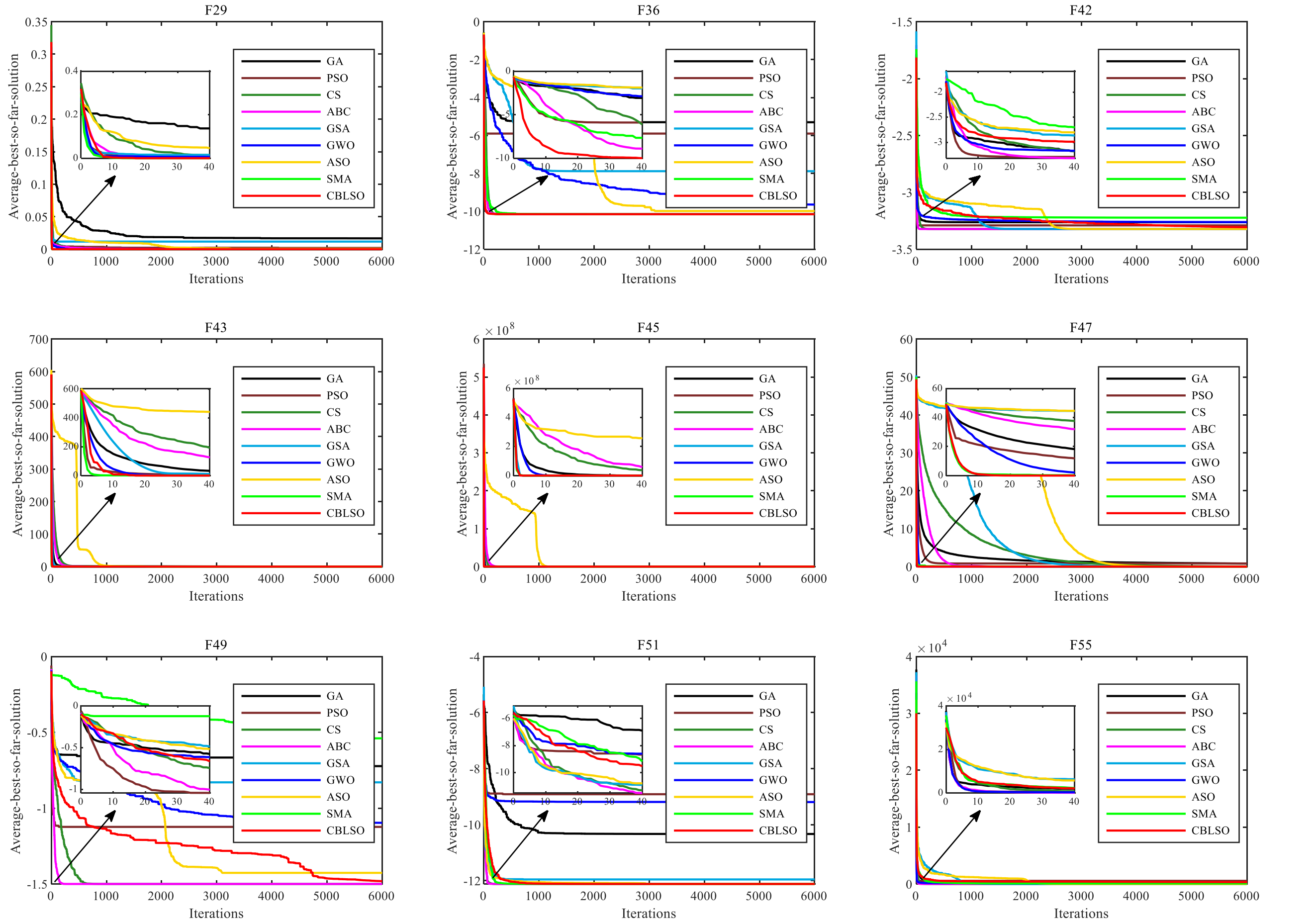


FIGURE 10. The convergence graphs of average best-so-far solutions obtained by CBLSO and other nature-inspired algorithms on MN problems.

$$SR_{ave} = \frac{1}{m}\sum_{i=1}^{m} SR_i \tag{21}$$

We compare the CBLSO algorithm with GA, PSO, CS, ABC, GSA, GWO, ASO, and SMA, in terms of convergence speed and robustness, and the results (i.e., NEFs, SR, AR, AR_{ave}, and SR_{ave}) of 56 test problems are presented in Tables 6-7. Besides, to check whether CBLSO saves computation time, the results of the runtime (T) over 50 runs are also recorded in Table 7.

From the results in Table 6, it is clear that CS ranks the first in terms of average NFEs for all test problems, and then it is ABC. The CBLSO ranks the third. Although CS and ABC outperform our algorithm in terms of average NFEs for all test problems, the CBLSO ranks the first best results than the other six algorithms on NFEs for all test problems. As can be inferred from Table 6, CBLSO converges much faster than GA, PSO, CS, ABC, GSA, GWO, ASO, and SMA on 41, 26, 20, 21, 34, 33, 36, and 33 test functions, while GA, PSO, CS, ABC, GSA, and GWO outperforms CBLSO on 5, 24, 33, 26, 14, 11, 14, and 16 test problems, respectively. Besides, CBLSO and GA, CBLSO, and PSO, CBLSO and CS, CBLSO and ABC, CBLSO and GSA, CBLSO and GWO show the same performance on 10, 6, 3, 9, 8, 12, 6, and 7 test functions, respectively. Although the CBLSO cannot surpass its competitors on all test functions, The overall average acceleration rates of CBLSO to GA, PSO, CS, ABC, GSA, GWO, ASO, and SMA are 347.4107, 118.9409, 127.0125, 133.1074, 210.5751, 21.5300, 239.9045, and 20.3052, respectively, which means CBLSO is faster than its competitors regarding to the overall average acceleration rates. Next, we pay attention to the SR and the execution time of 50 runs. As can be seen from Table 7 that GWO is the best algorithm in the case of average execution time and it also gets the first best results on execution time whereas our CBLSO ranks the sixth and the third, respectively. Moreover, when regarding the average SR and the best results on SR, CS performs the best (0.76 and 45), followed by our CBLSO algorithm (0.71 and 39). It also can be seen that there are 36 functions that CBLSO can solve 100% successfully (92.3% of the best results on SR), while the number for GA is 5 (71.4% of the best results on SR), for PSO is 23 (88.5% of the best results on SR), for CS is 40 (88.9% of the best results on SR), for ABC is 36 (94.7% of the best results on SR), for GSA is 27 (90.0% of the best results on SR), for GWO is 28 (96.5% of the best results on SR), for

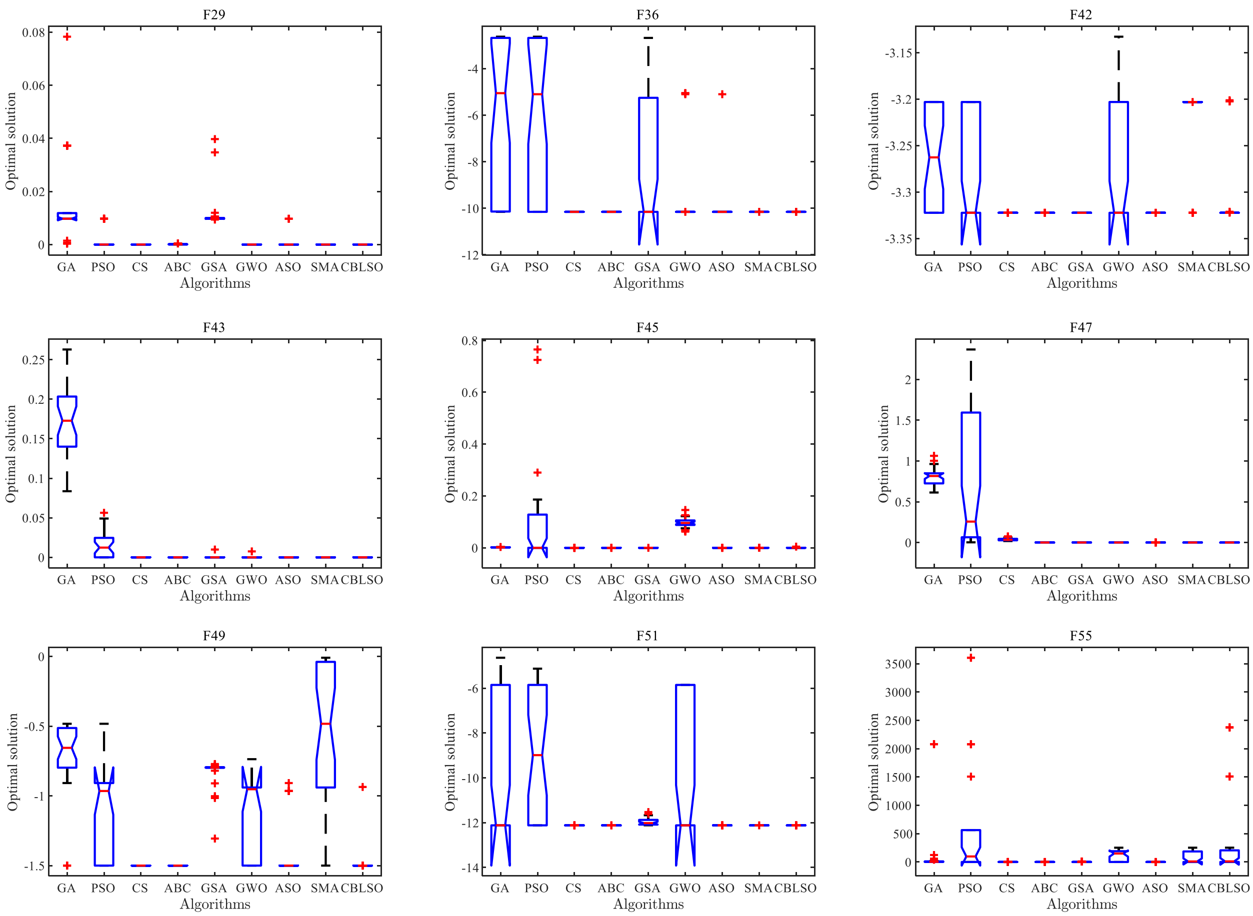


FIGURE 11. The box plots of average best-so-far solutions obtained by CBLSO and other nature-inspired algorithms on MN problems.

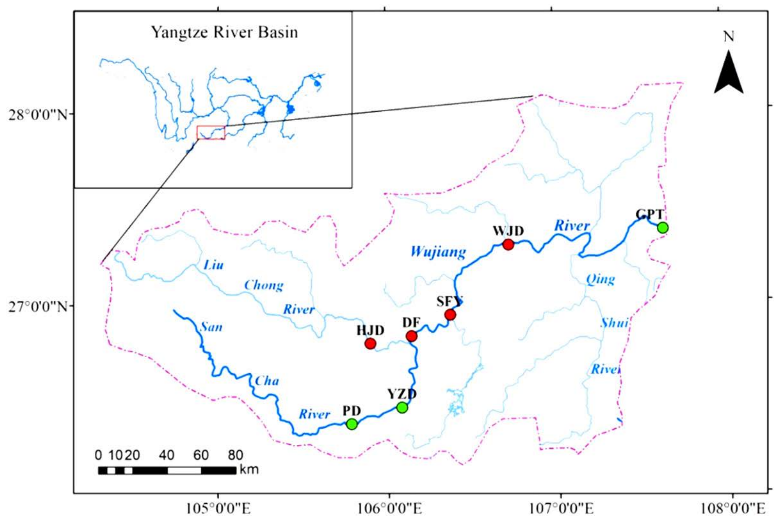


FIGURE 12. The index map of the Wujiang River basin.

ASO is 33 (82.5% of the best results on SR), and for SMA is 37 (92.5% of the best results on SR),. Thus, although GWO is better than our algorithm in terms of execution time, CBLSO achieves much better results than GWO in terms of Mean values, which can be seen from Tables 2-5. Besides, although CS has a superior performance in terms of SR and execution time, the percent of solving 100% successfully on problems of CBLSO is higher than CS. On the whole, the CBLSO is

TABLE 6. Experimental results in terms of NFEs obtained by GA, PSO, CS, ABC, GSA, GWO, ASO, SMA, and CBLSO.

Function	GA	GA/CBLSO	PSO	PSO/CBLSO	CS	CS/CBLSO	ABC	ABC/CBLSO	GSA	GSA/CBLSO	GWO	GWO/CBLSO	ASO	ASO/CBLSO	SMA	SMA/CBLSO	CBLSO
	NFEs	AR	NFEs	AR	NFEs	AR	NFEs	AR	NFEs	AR	NFEs	AR	NFEs	AR	NFEs	AR	NFEs
F01	115,329	364.9652	24,332	77.0000	45,146	142.8671	30,068	95.1519	42,187	133.5032	2,424	7.6719	161,983	512.6044	873	2.7627	**316**
F02	500,000	1179.2453	22,962	54.1557	80,215	189.1863	71,492	168.6132	216,951	511.6769	5,012	11.8208	296,980	700.4245	27,631	65.1675	**424**
F03	500,000	1440.9222	21,048	60.6571	73,276	211.1700	65,829	189.7089	245,896	708.6340	4,581	13.2017	315,767	909.9914	15,396	44.3689	**347**
F04	500,000	1.3651	500,000	1.3651	500,000	1.3651	500,000	1.3651	500,000	1.3651	500,000	1.3651	500,000	1.3651	484,661	1.3232	**366,285**
F05	500,000	14.8697	21,624	0.6427	5,965	0.1773	**3,066**	0.0911	145,616	4.3276	263,837	7.8411	229,164	6.8106	84,324	2.5061	33,648
F06	499,493	3.5868	**1,676**	0.0120	8,930	0.0641	1,901	0.0137	103,479	0.7431	41,528	0.2982	500,000	3.5904	397,646	2.8554	139,260
F07	500,000	2347.4178	1,282	6.0188	2,895	13.5915	1,540	7.2300	88,884	417.2958	498	2.3380	188,299	884.0329	385	1.8075	**213**
F08	500,000	1.0082	86,714	0.1748	**41,897**	0.0845	488,496	0.9850	174,199	0.3513	499,822	1.0078	300,149	0.6052	453,409	0.9142	495,938
F09	494,591	1.0091	**2,420**	0.0049	9,203	0.0188	10,649	0.0217	81,405	0.1661	471,567	0.9622	191,111	0.3899	215,139	0.4390	490,117
F10	500,000	1.0019	**10,782**	0.0216	20,060	0.0402	57,340	0.1149	78,104	0.1565	492,455	0.9867	188,635	0.3780	333,383	0.6680	499,070
F11	500,000	919.1176	10,375	19.0717	41,880	76.9853	67,119	123.3805	219,484	403.4632	3,083	5.6673	292,222	537.1728	15,874	29.1801	**544**
F12	500,000	959.6929	489,470	939.4818	349,080	670.0192	500,000	959.6929	446,667	857.3263	190,257	365.1766	415,030	796.6027	9,028	17.3282	**521**
F13	500,000	490.6771	500,000	490.6771	500,000	490.6771	500,000	490.6771	500,000	490.6771	500,000	490.6771	500,000	490.6771	283,646	278.3572	**1,019**
F14	500,000	781.2500	2,696	4.2125	5,099	7.9672	1,020	1.5938	12,155	18.9922	**636**	0.9938	81,235	126.9297	657	1.0266	640
F15	500,000	1216.5450	31,363	76.3090	158,842	386.4769	74,634	181.5912	414,156	1007.6788	6,718	16.3455	421,660	1025.9367	50,349	122.5036	**411**
F16	500,000	1.0000	500,000	1.0000	**463,050**	0.9261	500,000	1.0000	500,000	1.0000	500,000	1.0000	500,000	1.0000	490,052	0.9801	500,000
F17	**500,000**	1.0000	**500,000**	1.0000	**500,000**	1.0000	**500,000**	1.0000	**500,000**	1.0000	**500,000**	1.0000	**500,000**	1.0000	**500,000**	1.0000	**500,000**
F18	352,798	6.9440	**1,047**	0.0206	2,212	0.0435	1,310	0.0258	77,824	1.5318	126,984	2.4994	189,960	3.7389	30,843	0.6071	50,806
F19	500,000	608.2725	2,527	3.0742	7,136	8.6813	1,657	2.0158	146,860	178.6618	**748**	0.9100	246,755	300.1886	2,411	2.9331	822
F20	499,881	2.1259	1,729	0.0074	4,234	0.0180	**1,528**	0.0065	126,397	0.5375	363,301	1.5450	227,254	0.9665	124,978	0.5315	235,142
F21	500,000	1742.1603	500,000	1742.1603	500,000	1742.1603	500,000	1742.1603	500,000	1742.1603	9,026	31.4495	500,000	1742.2160	20,592	71.7491	**287**
F22	500,000	1633.9869	446,212	1458.2092	500,000	1633.9869	500,000	1633.9869	500,000	1633.9869	223,68	73.0980	500,000	1633.9869	16,697	54.5654	**306**
F23	161,432	0.5431	500,000	1.6822	500,000	1.6822	500,000	1.6822	500,000	1.6822	500,000	1.6822	500,000	1.6822	**25,151**	0.0846	297,229
F24	500,000	1.0000	500,000	1.0000	500,000	1.0000	500,000	1.0000	500,000	1.0000	500,000	1.0000	500,000	1.0000	**491,413**	0.9828	500,000
F25	90,302	9.3191	719	0.0742	2,099	0.2166	**603**	0.0622	83,365	8.6032	91,528	9.4456	177,777	18.3464	24,470	2.5253	9,690
F26	243,795	0.4880	340,879	0.6823	23,484	0.0470	**12,248**	0.0245	479,758	0.9603	499,552	0.9999	490,182	0.9812	466,088	0.9329	499,597
F27	374,221	0.7484	500,000	1.0002	**336,795**	0.6736	350,824	0.7016	500,000	1.0000	500,000	1.0000	500,000	1.0000	500,000	1.0000	500,000
F28	36,047	3.5403	261,237	25.6567	**3,072**	0.3017	4,614	0.4532	383,710	37.6851	244,852	24.0475	69,273	6.8035	6,935	0.6811	10,182
F29	500,000	1101.3216	118,680	261.4097	45,119	99.3811	493,823	1087.7159	500,000	1101.3216	5,493	12.0991	279,962	616.6564	590	1.2996	**454**
F30	195,102	55.9192	878	0.2516	1,789	0.5128	**806**	0.2310	75,514	21.6435	1,647	0.4721	172,786	49.5231	22,566	6.4678	3,489
F31	500,000	667.5567	2,603	3.4753	7,596	10.1415	1,758	2.3471	144,655	193.1308	**731**	0.9760	245,864	328.2563	1,375	1.8358	749
F32	500,000	764.5260	2,673	4.0872	6,997	10.6988	2,168	3.3155	138,044	211.0765	785	1.2003	236,454	361.5505	64,950	99.3119	**654**
F33	193,254	3.5851	**1,431**	0.0265	2,458	0.0456	14,314	0.2655	448,991	8.3293	92,867	1.7228	196,137	3.6386	68,369	1.2683	53,905
F34	342,158	58.4985	1,073	0.1835	3,596	0.6148	**1,039**	0.178	111,033	18.9832	3,585	0.6129	217,533	37.1915	15,815	2.7039	5,849
F35	500,000	2.7756	153,737	0.8534	**35,818**	0.1988	500,000	2.7776	500,000	2.7756	231,347	1.2843	422,118	2.3433	207,610	1.1525	180,139
F36	458,212	19.3583	310,972	13.1378	12,383	0.5232	**11,534**	0.4873	291,597	12.3193	495,356	20.9276	240,804	10.1734	99,420	4.2003	23,670
F37	473,767	14.5972	250,945	7.7319	12,908	0.3977	**4,997**	0.1540	142,340	4.3856	494,784	15.2448	239,957	7.3933	102,288	3.1516	32,456
F38	478,351	15.4952	263,093	8.5223	13,851	0.4487	**13,166**	0.4265	450,021	14.5775	494,519	16.0189	240,250	7.7824	111,107	3.5991	30,871
F39	500,000	1.0000	**385,457**	0.7709	495,989	0.9920	500,000	1.0000	500,000	1.0000	500,000	1.0000	496,330	0.9927	500,000	1.0000	500,000
F40	500,000	1.0000	**461,837**	0.9237	488,682	0.9774	500,000	1.0000	497,386	0.9948	500,000	1.0000	484,190	0.9684	492,046	0.9841	500,000
F41	36,380	0.1861	**673**	0.0034	2,615	0.0134	768	0.0039	122,042	0.6243	100,352	0.5134	208,221	1.0652	35,233	0.1802	195,483
F42	310,484	0.6422	210,810	0.4360	11,179	0.0231	**2,962**	0.0061	151,318	0.3130	473,524	0.9794	245,591	0.5080	489,076	1.0116	483,491
F43	500,000	941.6196	280,569	528.3785	115,740	217.9661	109,496	206.2072	177,191	333.6930	5,846	11.0094	289,263	544.7514	29,566	55.6798	**531**
F44	500,000	1492.5373	100,677	300.5284	211,271	630.6597	129,579	386.8030	373,796	1115.8090	7,071	21.1075	400,549	1195.6687	36,340	108.4776	**335**
F45	500,000	1.0063	**129,501**	0.2606	175,408	0.3530	165,430	0.3329	177,824	0.3579	500,000	1.0063	266,756	0.5369	459,109	0.9240	496,869
F46	500,000	1.0394	174,491	0.3627	**97,607**	0.2029	107,882	0.2243	190,049	0.3951	499,938	1.0393	276,223	0.5742	491,694	1.0221	481,054
F47	500,000	553.0973	500,000	553.0973	500,000	553.0973	136,692	151.2080	500,000	553.0973	8,833	9.7710	487,253	538.9967	107,193	118.5763	**904**
F48	327,364	14.5211	70,914	3.1456	3,382	0.1500	**1,346**	0.0597	461,430	20.4680	20,296	0.9003	164,543	7.2987	164,543	7.2987	22,544
F49	498,302	1.0280	241,220	0.4976	41,307	0.0852	**11,610**	0.0240	500,000	1.0315	495,077	1.0213	490,082	1.0110	494,601	1.0203	484,752
F50	500,000	1.0000	**480,196**	0.9604	490,909	0.9818	500,000	1.0000	500,000	1.0000	500,000	1.0000	481,279	0.9626	500,000	1.0000	500,000
F51	400,170	3.8679	271,110	2.6204	7,655	0.0740	**5,304**	0.0513	500,000	4.8328	385,366	3.7248	242,811	2.3469	75,872	0.7333	103,460
F52	500,000	1.0000	**500,000**	1.0000	**500,000**	1.0000	**500,000**	1.0000	**500,000**	1.0000	**500,000**	1.0000	**500,000**	1.0000	**500,000**	1.0000	**500,000**
F53	500,000	1.0000	500,000	1.0000	**290,754**	0.5815	500,000	1.0000	500,000	1.0000	499,999	1.0000	500,000	1.0000	500,000	1.0000	500,000
F54	500,000	1.0198	**2,453**	0.0050	9,168	0.0187	3,524	0.0072	212,358	0.4331	499,714	1.0192	293,446	0.5985	204,202	0.4165	490,312
F55	500,000	1.0000	353,403	0.7068	**81,633**	0.1633	419,006	0.8380	305,304	0.6106	500,000	1.0000	345,613	0.6912	496,491	0.9930	500,000
F56	500,000	1.0000	470,151	0.9403	484,096	0.9682	500,000	1.0000	**417,508**	0.8350	500,000	1.0000	374,087	0.7482	500,000	1.0000	500,000
Average	430,026 (9)	347.4107	214,654 (5)	118.9409	166,687 (1)	127.0125	194,342 (2)	133.1074	311,705 (7)	210.5751	270,677 (6)	21.5300	330,742 (8)	239.9045	211,466(4)	20.3052	209,371 (3)
Total	1 (9)		13 (3)		10 (4)		15 (2)		3 (7)		5 (5)		2 (8)		4 (6)		18 (1)

an effective algorithm with an acceptable convergence speed and stronger robustness.

E. COMPREHENSIVE SIGNIFICANCE ANALYSIS

The comparison of algorithms does not guarantee the efficacy and superiority of the proposed algorithm. The possibility of getting good results, by chance, cannot be ignored. To provide more accurate conclusions, a non-parametric statistical test named Wilcoxon Signed-Rank Test (WSRT) [77], [78] is used to better compare the overall performance of the algorithms. WSRT is a pair-wise test that aims at detecting the statistically significant difference between the behaviors of two algorithms. The null hypothesis H_0 for a two-sided test is: ''there is no difference between the median of the solutions provided by algorithm A and the median of the solutions provided by algorithm B for the same test function'' [77], [78]. To determine whether algorithm A achieves a statistically better solution than algorithm B, or if not, whether the alternative hypothesis is valid, the sum of the ranks by produced by WSRT is examined. When using WSRT, the R^+ and R^- regarding the comparisons between the two algorithms are computed firstly. Once the R^+ and R^- have been achieved, the p-values can be computed. Usually, there are two types of WSRT: single-problem statistical analysis and multi-problem statistical analysis. In this part, WSRT at a significant level of $\alpha = 0.05$ is used for multi-problem statistical analysis.

For single-problem statistical analysis, the Mean values of 30 runs for each test problem presented in Tables 2-5 are used for the sample data. The single-problem-based statistical comparisons between the CBLSO algorithm and one of the referred algorithms by WSRT as well as the corresponding statistical results for each test problem in 30 runs are shown in Tables 2-5, respectively. In these tables, '+' means the case in which the null hypothesis is rejected and CBLSO shows a better performance in the single-problem-based statistical comparisons tests at 95% significance level ($\alpha = 0.05$); '-' means the case in which the null hypothesis is rejected and CBLSO shows a worse performance; and '=' means a case in which no statistically significant difference between LSO and the other algorithms exists. From the results in Tables 2-5, we can see that the CBLSO significantly performs better than GA, PSO, CS, ABC, GSA, GWO, ASO, and SMA on the US problems. For the UN problems, the CBLSO significantly outperforms other peer algorithms except for SMA. For the MS problems, CBLSO significantly improves other peer algorithms and achieves comparable results compared

TABLE 7. Experimental results in terms of SR and T obtained by GA, PSO, CS, ABC, GSA, GWO, ASO, SMA, and CBLSO.

Function	GA		PSO		CS		ABC		GSA		GWO		ASO		SMA		CBLSO	
	SR	Time(s)	SR	Time(s)	SR	Time(s)	SR	Time(s)	SR	Time(s)	SR	Time(s)	SR	Time(s)	SR	Time(s)	SR	Time(s)
F01	**1.00**	3.723400	**1.00**	0.773120	**1.00**	1.217800	**1.00**	3.200000	**1.00**	5.000300	**1.00**	**0.047500**	**1.00**	19.5597	**1.00**	0.068125	**1.00**	0.100630
F02	0.00	14.46780	**1.00**	1.294700	**1.00**	2.783400	**1.00**	7.372500	**1.00**	25.98910	**1.00**	0.135940	**1.00**	40.6163	**1.00**	1.933100	**1.00**	**0.088438**
F03	0.00	12.63590	**1.00**	1.193100	**1.00**	2.488100	**1.00**	6.990300	**1.00**	19.47190	**1.00**	0.102190	**1.00**	43.0497	**1.00**	1.105900	**1.00**	**0.049688**
F04	0.00	19.62780	0.00	20.49690	0.00	13.56220	0.00	53.82870	0.00	45.18810	0.00	**5.921900**	0.00	53.7997	0.12	39.29310	**0.58**	17.75630
F05	0.00	3.101200	0.96	0.923440	**1.00**	**0.214060**	**1.00**	0.365620	**1.00**	8.065300	**1.00**	1.373800	**1.00**	26.4606	**1.00**	1.916900	**1.00**	1.930300
F06	0.02	3.256900	**1.00**	**0.100310**	**1.00**	0.240630	**1.00**	0.267500	0.92	5.689100	**1.00**	0.231560	0.00	56.0416	0.30	11.33410	**1.00**	7.786200
F07	0.00	3.106300	**1.00**	0.080937	**1.00**	0.100310	**1.00**	0.200310	**1.00**	5.144400	**1.00**	**0.010000**	**1.00**	23.0494	**1.00**	0.039063	**1.00**	0.037500
F08	0.00	3.672500	**1.00**	2.621200	**1.00**	**0.922500**	0.06	53.31130	**1.00**	12.01910	0.38	1.856200	**1.00**	33.8806	0.56	11.53160	0.16	26.40130
F09	0.02	3.581900	**1.00**	0.146880	**1.00**	**0.128440**	**1.00**	1.267200	**1.00**	5.960600	**1.00**	3.020600	**1.00**	25.2131	**1.00**	6.470300	**1.00**	35.42970
F10	0.00	3.968800	**1.00**	0.575310	**1.00**	**0.573440**	**1.00**	6.006900	**1.00**	7.259700	0.82	4.469100	**1.00**	25.9262	**1.00**	12.18940	0.50	37.58530
F11	0.00	5.048400	**1.00**	0.432500	**1.00**	1.110900	**1.00**	9.450000	**1.00**	19.04060	**1.00**	**0.047813**	**1.00**	35.4712	**1.00**	0.605620	**1.00**	0.062187
F12	0.00	12.18840	0.16	16.77440	0.98	10.22560	0.00	48.52940	0.38	29.37780	**1.00**	1.705000	0.66	45.1622	**1.00**	0.546880	**1.00**	**0.064375**
F13	0.00	14.94090	0.00	24.16060	0.00	12.39000	0.00	64.97750	0.00	45.25310	0.00	4.636600	0.00	47.7994	**1.00**	18.36000	**1.00**	**0.078125**
F14	0.00	11.18340	**1.00**	0.140940	**1.00**	0.265940	**1.00**	0.142810	**1.00**	2.074400	**1.00**	**0.026250**	**1.00**	14.8875	**1.00**	0.075938	**1.00**	0.086563
F15	0.00	10.95560	**1.00**	1.586200	**1.00**	6.193400	**1.00**	7.127500	**1.00**	29.97530	**1.00**	0.123440	**1.00**	46.1244	**1.00**	3.126600	**1.00**	**0.050937**
F16	0.00	**10.83250**	0.00	22.02120	**0.86**	11.08590	0.00	46.42810	0.00	33.35940	0.00	4.788800	0.00	44.9931	0.02	30.72130	0.00	21.79440
F17	**0.00**	12.51530	**0.00**	12.84060	**0.00**	7.892800	**0.00**	55.14560	**0.00**	43.64500	**0.00**	**3.562800**	**0.00**	38.5913	**0.00**	45.1809	**0.00**	31.98870
F18	0.60	2.604700	**1.00**	**0.053750**	**1.00**	0.077188	**1.00**	0.161870	**1.00**	4.524100	**1.00**	0.480630	**1.00**	19.1347	**1.00**	1.157200	**1.00**	3.081900
F19	0.00	2.614100	**1.00**	0.157810	**1.00**	0.169690	**1.00**	0.186880	**1.00**	9.614100	**1.00**	**0.011875**	**1.00**	23.1928	**1.00**	0.061875	**1.00**	0.080312
F20	0.02	2.494100	**1.00**	**0.105940**	**1.00**	0.132190	**1.00**	0.240630	**1.00**	7.189100	**1.00**	1.859700	**1.00**	18.9272	**1.00**	2.736900	**1.00**	13.52030
F21	0.00	13.30590	0.00	16.91430	0.00	17.35060	0.00	63.67310	0.00	47.64630	**1.00**	0.196560	0.00	40.4500	**1.00**	1.602500	**1.00**	**0.032188**
F22	0.00	13.45470	0.14	16.86190	0.00	15.66560	0.00	49.51810	0.00	33.76690	**1.00**	0.199060	0.00	39.1325	**1.00**	1.102200	**1.00**	**0.044687**
F23	**1.00**	4.702500	0.00	17.94380	0.00	13.93090	0.00	54.92030	0.00	40.07220	0.00	5.153100	0.00	41.4269	**1.00**	**1.797500**	0.92	18.42250
F24	0.00	8.881200	0.00	20.74410	0.00	15.31250	0.00	46.23910	0.00	34.47750	0.00	**5.081200**	0.00	35.3147	**0.02**	34.68840	0.00	32.25090
F25	**1.00**	0.730620	**1.00**	0.056250	**1.00**	**0.038750**	**1.00**	0.094375	**1.00**	5.282200	**1.00**	0.295940	**1.00**	15.0803	**1.00**	0.596880	**1.00**	0.711560
F26	0.92	3.111600	0.32	17.10190	**1.00**	**0.725310**	**1.00**	1.084700	0.06	17.95370	0.08	4.018700	0.04	36.53160	0.62	13.13410	0.28	30.69660
F27	0.76	5.443100	0.00	22.28910	**1.00**	6.701200	0.84	33.33060	0.00	27.13880	0.00	**3.318100**	0.00	32.18870	0.00	28.39590	0.00	34.20500
F28	**1.00**	0.709690	0.48	13.58590	**1.00**	**0.121870**	**1.00**	0.572500	0.24	22.22440	0.52	3.088800	**1.00**	7.960600	**1.00**	0.264060	**1.00**	0.877190
F29	0.00	3.506900	0.84	5.790900	**1.00**	1.099700	0.02	57.87130	0.00	25.17910	**1.00**	0.049375	0.94	19.69130	**1.00**	**0.043750**	**1.00**	0.060312
F30	0.92	1.380000	**1.00**	0.057813	**1.00**	0.066562	**1.00**	0.119370	**1.00**	6.389100	**1.00**	**0.013750**	**1.00**	13.12780	**1.00**	0.540000	**1.00**	0.248750
F31	0.00	2.800000	**1.00**	0.130310	**1.00**	0.190620	**1.00**	0.193750	**1.00**	8.385300	**1.00**	**0.011875**	**1.00**	17.75440	**1.00**	0.055000	**1.00**	0.077500
F32	0.00	2.753800	**1.00**	0.143750	**1.00**	0.186250	**1.00**	0.232810	**1.00**	7.288800	**1.00**	**0.012500**	**1.00**	15.99810	**1.00**	0.043125	**1.00**	0.071875
F33	0.96	1.242800	**1.00**	0.098125	**1.00**	**0.082500**	**1.00**	1.298400	0.12	18.80840	**1.00**	0.534380	**1.00**	19.88630	**1.00**	2.058100	**1.00**	3.274100
F34	0.64	2.407500	**1.00**	0.074375	**1.00**	0.102500	**1.00**	0.102190	**1.00**	5.656600	**1.00**	**0.027500**	**1.00**	20.58090	**1.00**	0.509690	**1.00**	0.430000
F35	0.00	4.283100	0.72	7.671200	**1.00**	**0.940000**	0.00	57.52190	0.00	24.77250	0.84	1.595900	0.40	22.88130	**1.00**	7.000000	**1.00**	10.98660
F36	0.16	2.743100	0.38	13.79530	**1.00**	**0.213750**	**1.00**	1.038400	0.58	20.64120	0.90	2.711900	**1.00**	27.97000	**1.00**	3.624100	**1.00**	1.104700
F37	0.14	4.453400	0.50	12.37780	**1.00**	**0.304380**	**1.00**	0.550940	**1.00**	10.54280	0.98	3.460600	**1.00**	22.12750	**1.00**	3.283100	**1.00**	2.405600
F38	0.08	4.501200	0.48	12.93590	**1.00**	**0.340940**	**1.00**	1.415600	0.14	17.47060	**1.00**	2.762500	**1.00**	19.69090	**1.00**	4.263700	**1.00**	2.294100
F39	0.00	6.169400	**0.24**	15.32160	0.02	9.976600	0.00	43.73030	0.00	16.02970	0.00	**3.364400**	0.02	28.15250	0.00	17.95970	0.00	22.27370
F40	0.00	2.666200	**0.08**	15.26310	0.06	7.841200	0.00	55.28160	0.02	28.61030	0.00	**2.243100**	**0.08**	38.67000	**0.08**	15.70560	0.00	37.12620
F41	**1.00**	0.242810	**1.00**	**0.059062**	**1.00**	0.101250	**1.00**	0.113750	**1.00**	9.022500	0.92	0.503750	**1.00**	23.67750	**1.00**	1.170000	**1.00**	12.00030
F42	0.42	2.458400	0.58	7.811600	**1.00**	**0.251880**	**1.00**	0.308440	**1.00**	8.300600	0.54	2.063400	**1.00**	18.59750	0.08	13.43880	0.96	26.29740
F43	0.00	14.85720	0.46	13.05410	**1.00**	4.315000	**1.00**	12.46530	**1.00**	20.08250	**1.00**	0.107190	0.90	32.62810	**1.00**	2.253100	**1.00**	**0.071875**
F44	0.00	11.79660	0.86	5.279400	**1.00**	5.850300	**1.00**	11.98190	**1.00**	34.78060	**1.00**	0.081562	**1.00**	35.76340	**1.00**	2.556900	**1.00**	**0.046250**
F45	0.00	16.21190	0.78	5.818400	**1.00**	**5.997500**	**1.00**	20.02750	**1.00**	23.84970	0.00	10.41560	**1.00**	36.12970	0.18	39.66750	0.06	45.22090
F46	0.00	23.20470	0.68	10.15840	**1.00**	**6.074700**	**1.00**	15.75560	**1.00**	21.10250	0.10	14.14410	**1.00**	27.71160	0.08	47.89120	0.06	33.96160
F47	0.00	168.9800	0.00	179.6519	0.00	295.5872	**1.00**	95.68750	0.00	186.4931	**1.00**	2.732200	**1.00**	176.4409	**1.00**	32.06310	**1.00**	**0.390000**
F48	0.68	3.267200	0.86	3.733800	**1.00**	**0.143440**	**1.00**	0.209690	0.08	21.33810	**1.00**	0.207500	**1.00**	13.25220	**1.00**	0.447810	**1.00**	1.599400
F49	0.02	5.701600	0.52	13.11840	**1.00**	1.260000	**1.00**	**1.089100**	0.00	20.44560	0.52	4.880000	0.04	31.02690	0.08	13.31440	**1.00**	36.15440
F50	0.00	9.197200	0.04	14.91190	**0.10**	10.62750	0.00	43.59910	0.00	17.81280	0.00	**7.143400**	0.08	35.37220	0.00	18.91280	0.00	25.20090
F51	0.36	2.201600	0.46	8.161600	**1.00**	**0.136870**	**1.00**	0.512190	0.00	18.31500	0.52	2.019700	**1.00**	23.02250	**1.00**	1.813700	**1.00**	3.950600
F52	**0.00**	3.547200	**0.00**	20.05720	**0.00**	9.682200	**0.00**	39.77370	**0.00**	17.29030	0.00	**3.142200**	0.00	35.99280	0.00	17.03250	**0.00**	31.07410
F53	0.00	8.767200	0.00	17.19620	**0.48**	9.594100	0.00	48.63880	0.00	16.36720	0.02	**4.850000**	0.00	29.61280	0.00	18.50530	0.00	22.12970
F54	0.00	3.334100	**1.00**	**0.075313**	**1.00**	0.192810	**1.00**	0.318120	**1.00**	9.947800	0.58	1.652500	**1.00**	23.94880	**1.00**	5.494700	0.18	26.43870
F55	0.00	3.865900	0.32	10.13530	**1.00**	1.430900	0.44	41.95780	0.94	10.54090	0.00	**2.824100**	**1.00**	32.11720	0.26	19.26940	0.00	26.13590
F56	0.00	**7.431300**	0.10	20.33060	0.14	15.49500	0.00	61.97090	**0.46**	19.86280	0.00	6.443400	0.94	27.71840	0.00	21.19530	0.00	44.32030
Average	0.21 (9)	9.479134 (3)	0.59 (7)	11.52126 (5)	0.76 (1)	9.459051 (2)	0.67 (5)	21.93570 (7)	0.55 (8)	22.10230 (8)	0.64 (6)	2.424313 (1)	0.68 (4)	32.13410 (9)	0.70 (3)	10.35980 (4)	0.71 (2)	13.04571 (6)
Total	7 (9)	2 (5)	26 (8)	5 (4)	45 (1)	16 (2)	38 (4)	1 (7)	30 (6)	0 (8)	29 (7)	19 (1)	36 (5)	0 (8)	40 (2)	2 (5)	39 (3)	10 (3)

with SMA. For the MN problems, the performance of LSO is not inferior to that of GA, PSO, GSA, and GWO, and it seems that CBLSO is as competitive as other peer algorithms.

For multi-problem-based statistical analysis, the Mean values of the best solutions of 30 runs for each test problem are used for the pair-wise comparisons. In this part, fifty-six Mean values of all benchmark functions in Tables 2-5 are used for the sample data of WSRT. The multi-problem-based statistical comparisons between the CBLSO algorithm and one of the referred algorithms by WSRT as well as the corresponding statistical results for each test problem in 30 runs are shown in Table 8. According to Table 8, we can find that for US and MS problems, there is no statistically significant difference between CBLSO and its competitors. For UN problems, the CBLSO is statistically significantly better than GA and GWO, and performs as competitive as PSO, CS, ABC, GSA, ASO, and SMA. For MN problems, CBLSO performs significantly better than GA, PSO, and GSA, and shows no statistically significant difference between CBLSO and ABC, GWO, ASO, and SMA, although the performance of CBLSO is inferior to that of the CS. As for all problems, the CBLSO shows an improvement over GA, PSO, GSA, and GWO, and achieves comparable results compared with CS, ABC, ASO, and SMA. Thus, we can conclude that CBLSO is a superior and competitive algorithm for numerous functions.

F. FURTHER COMPARISON ON CEC2014 TEST SUITE

In this section, the performance of CBLSO is further compared with some renowned algorithms on 30 benchmark functions from the CEC2014 competition with $D = 30$ [79]. These functions consist of Uni-modal problems (FC01-FC03), Multi-modal problems (FC04-FC16), Hybrid and Composite type with varying difficulty levels (FC17-FC22 and FC23-FC30). Specifically, the range of the search space is in $[-100, 100]^d$. The algorithms used here are listed in Table 9. In the case of the problems with $D = 30$, the population size is set to 50 and the maxNFEs is set to D*5000. The CBLSO algorithm is run independently 30 times for each benchmark function, and the values of Mean, Std, and best solution (Best) of different algorithms are shown in Table 10.

TABLE 8. Experimental results of the multi-problem-based WSRT between CBLSO and GA, PSO, CS, ABC, GSA, GWO, ASO, and SMA on classic test functions.

CBLSO vs.	Uni-modal problems							
	US problems				UN problems			
	R^+	R^-	p-value	$\alpha = 0.05$	R^+	R^-	p-value	$\alpha = 0.05$
GA	0	6	0.250	=	0	91	0.000	+
PSO	0	10	0.125	=	16	39	0.275	=
CS	0	6	0.250	=	29	49	0.470	=
ABC	0	6	0.250	=	26	52	0.339	=
GSA	0	6	0.250	=	18	60	0.110	=
GWO	0	1	1.000	=	4	41	0.027	+
ASO	0	1	1.000	=	17	38	0.322	=
SMA	0	1	1.000	=	19	9	0.469	=
CBLSO vs.	Multi-modal problems							
	MS problems				MN problems			
	R^+	R^-	p-value	$\alpha = 0.05$	R^+	R^-	p-value	$\alpha = 0.05$
GA	21	24	0.910	=	75	276	0.009	+
PSO	4	24	0.109	=	55	221	0.010	+
CS	6	22	0.219	=	110	10	0.003	−
ABC	6	22	0.219	=	98	73	0.609	=
GSA	4	24	0.109	=	83	193	0.098	+
GWO	1	14	0.125	=	75	135	0.277	=
ASO	4	24	0.109	=	95	58	0.404	=
SMA	15	0	0.063	=	52	53	1.000	=
CBLSO vs.	All problems							
	US problems + UN problems + MS problems + MN problems							
	R^+	R^-	p-value	$\alpha = 0.05$				
GA	241	1085	0.000	+				
PSO	185	805	0.000	+				
CS	405	298	0.428	=				
ABC	338	482	0.340	=				
GSA	248	787	0.002	+				
GWO	139	491	0.003	+				
ASO	255	375	0.334	=				
SMA	233	145	0.301	=				

TABLE 9. List of algorithms used for comparison.

Method	Reference
Fitness-Distance-Ration based PSO (FDR-PSO)	Peram et al. [82]
Fully informed particle swarm (FIPS)	Liang et al. [83]; Mendes et al. [84]
Comprehensive learning particle swarm optimizer (CLPSO)	Liang et al. [85]
Self-Adapting Control Parameters in DE (jDE)	Brest et al. [86]
Self-Adaptive DE (SaDE)	Qin et al. [87]
Teaching-learning-based optimization (TLBO)	Rao et al. [47]
Elitist teaching-learning-based optimization (ETLBO)	Rao et al. [88]
Teaching-learning-based optimization with dynamic group strategy (DGSTLBO)	Zou et al. [89]
Backtracking search optimization algorithm (BSA)	Civicioglu [90]
Learning backtracking search optimization algorithm (LBSA)	Chen et al. [80]
Sine-cosine algorithm and PSO (SCA-PSO)	Nenavath et al. [81]

Note that the fitness function is the error between the real optimal solution and the solution obtained by the algorithm. Assume that the real global optimal solution is x^*, and the best solution provided by the optimization algorithm is y^*. Then, $|f(y^*)-f(x^*)|$ is selected to be the fitness function [80]. In Table 10, the data for these compared algorithms are provided by Chen et al. [80] and Nenavath et al. [81]. The best results are shown in bold.

Table 10 shows that CBLSO performs best for functions FC23, FC24, FC25, FC27, FC28, FC29, and FC30 according to the value of Mean. jDE performs best for functions FC03, FC04, FC08, FC10, FC20, FC21, FC25, and FC26 according to the value of Mean. SaDE performs best in terms of the value of Mean for functions FC02, FC06, FC07, FC18, FC19, and FC26. The performance in terms of Mean with LBSA is better than the others for functions FC09, FC12, FC14, FC16, and FC22, while SCA-PSO is better than the others for functions FC01, FC11, and FC17. For function FC13, the best value of Mean is provided by BSA. For function FECE15, FDR-PSO has the smallest mean value among all algorithms. The three TLBOs, SCA-PSO and CBLSO provide the same mean value for function FC24, which is the smallest of the twelve algorithms. Considering the best solutions, CBLSO performs best for functions FC23, FC24, FC25, FC26, FC27, FC28, FC29, and FC30. jDE performs best for 9 functions (FC02, FC03, FC05, FC08, FC10, FC18, FC20, FC25, and FC26), while SaDE performs best for 6 functions (FC02, FC06, FC07, FC21 and FC26). For functions FC05, the jDE, BSA, LBSA, and SCA-PSO have the same best solutions, which is the smallest of the twelve algorithms. For functions FC24, the best solutions obtained by TLBO, ETLBO, DGSTLBO, and CBLSO can produce are the smallest. For function FC25, jDE, TLBO, ETLBO, DGSTLBO and CBLSO provide the smallest best solutions among all algorithms, while for function FC26, all the algorithms can obtain the best solutions. FDAR-PSO is better than other algorithms in terms of the best solutions for functions FC15, FC16, and FC19. ETLBO outperforms other algorithms in terms of the best solutions for function FC14, BSA outperforms other algorithms in terms of the best solutions for function FC12 and FC13, LBSA outperforms other algorithms in terms of the best solutions for function FC04 and FC09, and SCA-PSO outperforms other algorithms in terms of the best solutions for function FC01, FC11, and FC22. When we pay attention to the value of Mean of the twelve algorithms for 30 functions, we can find that CBLSO, jDE, and SaDE, rank in first place for seven functions, BSA and SCA-PSO for six functions, ETLBO for three functions, BSA for two functions, and FDR-PSO, TLBO, and DGSTLBO for one function. The statistical analysis results conducted by the multi-problem-based WSRT at a significant level of $\alpha = 0.05$ in Table 11 show that CBLSO significantly outperforms FDR-PSO, FIPS, CLPSO, jDE, SaDE, ETLBO, DGSTLBO, BSA, and LBSA in terms of p-value, although the statistical results between CBLSO and TLBO, CBLSO, and SCA-PSO are not significant, suggesting that CBLSO is very effective. Thus, it can be concluded that CBLSO can show strong and competitive performance compared with other algorithms.

V. REAL APPLICATION ON OPTIMAL DISPATCH OF CASCADE HYDROPOWER STATIONS

The problem considered in this study is called the optimal dispatch of cascade hydropower stations. Traditional optimization methods or algorithms may face trouble with problems that having many equality or inequality constraints as well as interdependent relationships between decision variables and may require more *NFEs* [91]. Many water resources and hydrological optimal dispatch problems often involve many equality or inequality constraints as well as interdependent relationships among decision variables. Moreover, the size of the decision variables is often large. This paper utilizes these aspects to investigate the performance of CBLSO in tackling these problems.

In this section, the optimal dispatch problem of Wujiang cascade hydropower stations in Guizhou province, southwest of China [91], is employed to evaluate the great potential of CBLSO for real application. In this problem, there is often interdependence among one or more decision variables and

TABLE 10. Experimental results of 12 algorithms on 30-dimensional CEC2014 problems.

Function		FDR-PSO	FIPS	CLPSO	jDE	SaDE	TLBO	ETLBO	DGSTLBO	BSA	LBSA	SCA-PSO	CBLSO
FC01	Mean	2.78E+06	1.02E+07	2.92E+07	3.37E+05	4.41E+05	8.28E+05	3.33E+05	1.04E+07	5.12E+06	2.79E+05	**2.32E+05**	6.52E+07
	SD	1.98E+06	2.88E+06	4.33E+06	2.98E+05	3.04E+05	7.89E+05	2.02E+05	8.61E+06	4.10E+06	**1.65E+05**	2.33E+05	3.07E+07
	Best	1.32E+05	5.60E+06	2.14E+07	1.06E+05	1.23E+05	5.77E+04	1.37E+05	3.15E+06	4.46E+05	7.85E+04	**5.68E+04**	1.75E+07
FC02	Mean	1.53E+08	1.13E+04	1.26E+03	1.99E−14	**2.84E−15**	1.44E+02	1.19E+02	4.59E+06	8.34E−01	4.83E−14	6.77E+08	5.79E+09
	SD	3.61E+08	5.27E+03	7.40E+02	1.37E−14	**8.99E−15**	1.72E+02	1.56E+02	1.11E+07	1.48E+00	1.92E−14	2.04E+08	4.41E+09
	Best	4.60E+01	3.63E+03	2.76E+02	**0.00E+00**	**0.00E+00**	2.82E+01	3.29E−01	1.48E+03	5.38E−02	2.84E−14	3.35E+01	1.07E+09
FC03	Mean	5.40E+02	6.94E+03	7.67E+02	**4.55E−14**	4.30E−12	2.39E+03	2.14E+03	1.44E+01	5.74E−04	2.27E−13	9.57E+01	3.91E+04
	SD	6.05E+02	5.03E+03	1.20E+03	**2.40E−14**	1.25E−11	1.77E+03	1.02E+03	1.68E+01	9.44E−04	1.00E−13	1.85E+02	8.24E+03
	Best	1.36E+01	1.32E+03	5.76E+01	**0.00E+00**	0.00E+00	2.90E+02	5.59E+02	6.71E−01	3.13E−05	1.14E−13	5.63E+01	2.46E+04
FC04	Mean	9.50E+01	2.67E+01	1.16E+02	**2.49E+01**	8.29E+01	9.70E+01	7.53E+01	1.46E+02	9.83E+01	4.23E+01	4.79E+02	3.97E+02
	SD	3.24E+01	**6.37E−01**	1.33E+01	1.74E+01	2.70E+01	3.36E+01	5.88E+00	3.78E+01	2.96E+01	3.57E+01	1.62E+01	1.87E+02
	Best	6.66E+01	2.53E+01	9.56E+01	1.81E+01	6.73E+01	6.74E+01	6.79E+01	8.79E+01	6.81E+01	**1.73E−03**	4.50E+01	1.07E+01
FC05	Mean	2.09E+01	2.10E+01	2.05E+01	2.04E+01	2.06E+01	2.10E+01	2.09E+01	2.10E+01	2.04E+01	**2.03E+01**	**2.03E+01**	2.09E+01
	SD	1.22E−01	5.71E−02	4.58E−02	3.56E−02	5.89E−02	5.23E−02	7.24E−02	4.34E−02	**1.56E−02**	3.42E−02	3.92E−02	6.64E−02
	Best	2.06E+01	2.09E+01	2.04E+01	**2.03E+01**	2.05E+01	2.09E+01	2.08E+01	2.09E+01	**2.03E+01**	**2.03E+01**	**2.03E+01**	2.08E+01
FC06	Mean	7.67E+00	6.19E+00	1.70E+01	1.48E+01	**2.61E−01**	1.54E+01	1.59E+01	1.67E+01	1.62E+01	8.37E+00	6.08E+01	2.75E+01
	SD	1.98E+00	2.22E+00	6.10E−01	3.97E+00	**4.92E−01**	2.14E+00	2.75E+00	3.45E+00	9.69E−01	3.22E+00	5.96E−01	4.41E+00
	Best	4.63E+00	2.75E+00	1.60E+01	4.01E+00	**0.00E+00**	1.17E+01	1.01E+01	1.23E+01	1.47E+01	5.25E+00	6.06E+01	1.49E+01
FC07	Mean	1.33E+01	2.56E−03	3.92E−03	1.71E−13	**0.00E+00**	6.74E−02	2.75E−02	1.01E+00	4.19E−03	5.41E−03	4.08E−03	2.13E+01
	SD	1.31E+01	6.75E−03	1.61E−03	8.04E−14	**0.00E+00**	8.69E−02	2.23E−02	1.50E+00	1.32E−02	8.26E−03	1.83E−03	2.06E+01
	Best	1.48E−02	7.08E−05	1.37E−03	1.14E−13	**0.00E+00**	7.84E−12	1.02E−12	1.20E−01	6.62E−07	1.14E−13	2.03E−03	7.20E+00
FC08	Mean	3.48E+01	6.60E+01	5.86E−01	**1.14E−14**	1.99E−01	7.37E+01	8.00E+01	7.67E+01	2.93E+00	1.14E−03	8.39E+01	1.32E+02
	SD	5.70E+00	1.22E+01	6.20E−01	3.60E−14	4.20E−01	2.11E+01	1.88E+01	2.45E+01	1.46E+00	**0.00E+00**	4.23E+00	3.06E+01
	Best	2.19E+01	4.32E+01	2.65E−04	**0.00E+00**	0.00E+00	5.07E+01	4.48E+01	3.48E+01	1.14E+00	1.14E−13	6.26E+01	8.49E+01
FC09	Mean	5.53E+01	1.53E+02	7.79E+01	5.82E+01	7.70E+01	7.68E+01	8.00E+01	9.84E+01	5.95E+01	**4.32E+01**	9.41E+02	1.90E+02
	SD	1.54E+01	1.61E+01	1.49E+01	7.79E+00	1.68E+01	1.08E+01	1.42E+01	3.08E+01	7.94E+00	7.61E+00	**4.51E+00**	3.24E+01
	Best	3.51E+01	1.21E+02	5.52E+01	4.78E+01	4.47E+01	5.87E+01	5.87E+01	6.41E+01	4.56E+01	**2.70E+01**	9.33E+02	1.34E+02
FC10	Mean	9.70E+02	1.87E+03	2.50E+01	**1.69E−01**	2.56E+00	1.72E+03	1.57E+03	2.39E+03	3.22E+01	1.08E+01	1.81E+03	4.77E+03
	SD	3.47E+02	4.44E+02	5.21E+00	**5.12E−01**	4.09E+00	7.41E+02	4.26E+02	4.71E+02	6.57E+00	5.07E+00	1.08E+02	9.97E+02
	Best	3.70E+02	1.35E+03	1.83E+01	**0.00E+00**	3.39E−02	4.64E+02	9.52E+02	1.59E+03	2.15E+01	5.12E+00	1.55E+03	2.87E+03
FC11	Mean	3.08E+03	5.76E+03	3.24E+03	2.88E+03	4.11E+03	6.71E+03	6.46E+03	3.39E+03	2.56E+03	2.31E+03	**2.10E+03**	5.91E+03
	SD	6.19E+02	3.43E+02	2.18E+02	3.06E+02	4.80E+02	3.92E+02	4.46E+02	5.45E+02	2.56E+02	3.17E+02	**1.28E+02**	8.90E+02
	Best	1.77E+03	5.23E+03	2.98E+03	2.51E+03	3.18E+03	6.01E+03	5.52E+03	2.42E+03	2.12E+03	1.86E+03	**1.75E+03**	3.69E+03
FC12	Mean	8.12E−01	2.62E+00	5.42E−01	4.96E−01	1.05E+00	2.64E+00	2.60E+00	2.75E+00	4.37E−01	**4.18E−01**	2.63E+00	2.13E+00
	SD	6.51E−01	2.82E−01	8.53E−02	5.71E−02	1.24E−01	2.47E−01	3.32E−01	2.62E−01	7.85E−02	**5.15E−02**	1.28E−01	3.31E−01
	Best	3.12E−01	1.97E+00	4.12E−01	3.72E−01	7.98E−01	2.13E+00	1.93E+00	2.48E+00	**3.05E−01**	3.62E−01	2.20E+00	1.38E+00
FC13	Mean	4.35E−01	3.46E−01	4.18E−01	3.10E−01	3.09E−01	4.88E−01	4.11E−01	4.71E−01	**2.84E−01**	2.90E−01	4.90E−01	7.26E−01
	SD	1.07E−01	**3.30E−02**	7.14E−02	4.77E−02	3.58E−02	1.14E−01	6.86E−02	1.13E−01	4.68E−02	4.34E−02	1.17E−01	5.03E−01
	Best	2.55E−01	2.89E−01	2.77E−01	2.58E−01	2.62E−01	3.50E−01	2.97E−01	3.09E−01	**2.06E−01**	2.10E−01	3.60E−01	4.22E−01
FC14	Mean	8.91E−01	3.07E−01	3.44E−01	2.95E−01	2.81E−01	2.88E−01	2.78E−01	2.88E−01	2.45E−01	**2.30E−01**	3.17E−01	1.19E+01
	SD	3.59E−01	3.63E−02	3.38E−02	2.67E−02	**2.64E−02**	4.70E−02	4.42E−02	4.92E−02	4.02E−02	2.69E−02	3.73E−02	1.03E+01
	Best	1.88E−01	2.54E−01	2.97E−01	2.56E−01	2.25E−01	2.11E−01	**1.84E−01**	2.26E−01	2.03E−01	1.87E−01	2.64E−01	1.53E−01
FC15	Mean	**4.49E+00**	1.58E+01	1.09E+01	5.81E+00	9.71E+00	1.80E+01	1.89E+01	3.75E+01	7.06E+00	5.93E+00	1.63E+01	3.55E+02
	SD	**6.87E−01**	9.26E−01	9.63E−01	8.14E−01	1.28E+00	5.93E+00	8.66E+00	2.19E+01	1.07E+00	1.22E+00	8.29E−01	3.39E+02
	Best	**3.18E+00**	1.41E+01	9.09E+00	4.24E+00	7.32E+00	1.03E+01	9.75E+00	1.54E+01	5.16E+00	3.87E+00	9.53E+00	3.32E+01
FC16	Mean	1.05E+01	1.18E+01	1.11E+01	1.06E+01	1.15E+01	1.20E+01	1.19E+01	1.11E+01	1.07E+01	**1.02E+01**	1.60E+01	1.21E+01
	SD	8.64E−01	2.50E−01	2.31E−01	1.90E−01	2.07E−01	4.35E−01	3.94E−01	6.62E−01	2.71E−01	4.55E−01	**1.62E−01**	5.04E−01
	Best	**8.55E+00**	1.13E+01	1.07E+01	1.03E+01	1.13E+01	1.14E+01	1.14E+01	1.01E+01	1.01E+01	9.37E+00	1.56E+01	1.09E+01
FC17	Mean	1.36E+05	3.76E+05	2.07E+06	3.16E+04	1.33E+04	1.91E+05	2.11E+05	1.67E+05	1.54E+05	4.42E+04	**1.16E+04**	3.01E+06
	SD	1.04E+05	1.12E+05	6.62E+05	3.34E+04	6.50E+03	1.73E+05	9.21E+04	2.13E+05	8.75E+04	3.70E+04	**4.16E+03**	1.80E+06
	Best	2.69E+04	2.44E+05	1.07E+06	2.49E+03	**1.73E+03**	3.29E+04	8.02E+04	3.54E+04	4.92E+04	8.62E+03	4.60E+03	7.10E+05
FC18	Mean	3.38E+03	1.59E+03	4.92E+02	2.63E+02	**6.89E+01**	2.93E+03	3.31E+03	8.71E+02	9.10E+02	1.14E+03	9.69E+02	9.52E+06
	SD	4.21E+03	8.50E+02	9.30E+01	5.11E+02	**3.43E+01**	2.35E+03	4.11E+03	1.02E+03	1.05E+03	1.29E+03	1.26E+01	2.20E+07
	Best	3.13E+02	4.78E+02	3.44E+02	**1.35E+01**	3.09E+01	1.35E+02	1.64E+02	6.17E+01	4.35E+01	6.23E+01	2.37E+02	2.44E+04
FC19	Mean	1.57E+01	1.19E+01	1.01E+01	1.41E+01	**5.55E+00**	2.12E+01	2.12E+01	2.71E+01	6.91E+00	6.31E+00	1.90E+01	8.02E+01
	SD	2.33E+01	8.29E−01	1.02E+00	1.13E+00	6.59E−01	2.55E+01	2.54E+01	2.86E+01	6.25E−01	1.01E+00	**5.58E−01**	2.59E+01
	Best	**4.44E+00**	1.01E+01	8.20E+00	1.19E+01	4.74E+00	5.77E+00	6.72E+00	9.65E+00	6.06E+00	4.60E+00	1.90E+01	2.32E+01
FC20	Mean	7.05E+03	5.81E+03	5.95E+03	**3.17E+01**	4.31E+01	1.54E+03	1.44E+03	4.28E+02	1.68E+02	9.02E+01	2.31E+03	1.76E+04
	SD	1.27E+04	2.71E+03	3.35E+03	3.78E+01	**2.96E+01**	9.69E+02	6.69E+02	1.77E+02	1.91E+02	8.20E+01	1.25E+02	1.06E+03
	Best	1.37E+03	2.36E+03	2.48E+03	**1.08E+01**	1.91E+01	6.68E+02	3.41E+02	2.03E+02	2.75E+01	3.06E+01	2.10E+03	1.01E+04
FC21	Mean	6.77E+04	1.49E+05	3.39E+05	3.66E+03	**2.49E+03**	9.72E+04	1.19E+05	2.20E+04	6.20E+03	9.86E+03	3.43E+03	5.43E+05
	SD	4.21E+04	6.40E+04	1.22E+05	3.15E+03	2.80E+03	9.01E+04	1.00E+05	2.22E+04	3.02E+03	1.32E+04	**3.09E+02**	8.09E+04
	Best	8.85E+03	8.47E+04	1.25E+05	6.53E+02	**2.21E+02**	1.89E+04	3.16E+04	4.09E+03	2.22E+03	1.22E+03	2.96E+03	4.03E+05
FC22	Mean	2.27E+02	2.25E+02	2.70E+02	2.96E+02	1.36E+02	2.81E+02	3.27E+02	3.14E+02	1.79E+02	**1.14E+02**	2.24E+02	4.21E+02
	SD	9.97E+01	7.27E+01	5.98E+01	8.12E+01	7.92E+01	1.08E+02	1.33E+02	1.41E+02	8.11E+01	7.56E+01	**7.26E+00**	1.38E+02
	Best	1.44E+02	1.67E+02	1.59E+02	9.35E+01	2.51E+01	1.64E+02	5.82E+01	1.49E+02	5.06E+01	2.41E+01	**2.23E+01**	2.01E+02
FC23	Mean	3.18E+02	3.14E+02	3.15E+02	3.14E+02	3.15E+02	3.15E+02	3.15E+02	3.15E+02	3.15E+02	3.15E+02	3.15E+02	**2.00E+02**
	SD	3.50E+00	1.57E−04	2.29E−01	2.89E−13	**0.00E+00**	1.23E−11	1.78E−10	4.43E−01	2.92E−07	1.44E−13	3.45E−10	**0.00E+00**
	Best	3.15E+02	3.14E+02	3.15E+02	3.14E+02	3.15E+02	3.15E+02	3.15E+02	3.15E+02	3.15E+02	3.15E+02	3.15E+02	**2.00E+00**
FC24	Mean	2.27E+02	2.24E+02	2.27E+02	2.27E+02	2.25E+02	**2.00E+02**	**2.00E+02**	**2.00E+02**	2.27E+02	2.26E+02	**2.00E+02**	**2.00E+02**
	SD	2.58E+00	5.46E−01	1.01E+00	4.59E+00	5.21E−01	2.20E−03	2.19E−03	9.68E−04	2.42E+00	1.28E+00	2.48E−03	**0.00E+00**
	Best	2.23E+02	2.23E+02	2.25E+02	2.24E+00	2.24E+02	**2.00E+02**	**2.00E+02**	**2.00E+02**	2.25E+02	2.24E+02	**2.00E+02**	**2.00E+00**
FC25	Mean	2.06E+02	2.07E+02	2.10E+02	**2.00E+02**	2.06E+02	2.01E+02	**2.00E+02**	2.02E+02	2.07E+02	2.09E+02	2.65E+03	**2.00E+02**
	SD	2.09E+02	2.45E+00	1.62E+02	1.46E−01	2.49E+00	2.39E+00	5.33E−06	3.62E+00	6.48E−01	3.10E+00	5.25E+00	**0.00E+00**
	Best	2.04E+02	2.04E+02	2.08E+02	**2.00E+02**	2.03E+02	**2.00E+02**	**2.00E+02**	**2.00E+02**	2.06E+02	2.04E+02	2.63E+03	**2.00E+00**
FC26	Mean	1.51E+02	1.70E+02	**1.00E+02**	**1.00E+02**	**1.00E+02**	1.10E+02	**1.00E+02**	1.10E+02	**1.00E+02**	**1.00E+02**	**1.00E+02**	1.01E+02
	SD	5.18E+01	4.83E+01	**3.00E−02**	6.12E−02	7.09E−02	3.15E+01	1.57E−01	3.15E+01	5.83E−02	7.52E−02	7.66E−02	9.31E−01
	Best	**1.00E+02**	**1.00E+02**	**1.00E+02**	**1.00E+02**	**1.00E+02**	**1.00E+02**	**1.00E+02**	**1.00E+02**	**1.00E+02**	**1.00E+02**	**1.00E+02**	**1.00E+02**
FC27	Mean	7.12E+02	4.34E+02	4.38E+02	5.65E+02	3.62E+02	5.47E+02	6.56E+02	7.94E+02	4.09E+02	4.34E+02	2.75E+02	**2.00E+02**
	SD	1.84E+02	3.50E+02	1.33E+01	1.73E+02	5.04E+01	1.66E+02	2.13E+02	2.15E+02	3.71E+00	6.71E+01	2.33E+01	**0.00E+00**
	Best	4.02E+02	3.76E+02	4.17E+02	4.01E+02	3.00E+02	4.02E+02	4.02E+02	4.06E+02	4.04E+02	3.71E+02	2.71E+02	**2.00E+00**
FC28	Mean	1.48E+03	3.98E+02	9.21E+02	3.83E+02	8.51E+02	1.24E+03	1.18E+03	1.43E+03	8.77E+02	8.34E+02	9.23E+02	**2.00E+02**
	SD	3.70E+02	1.16E+01	3.47E+01	5.59E+00	2.76E+01	3.51E+02	2.30E+02	4.37E+02	1.66E+01	3.87E+01	2.29E+01	**0.00E+00**

TABLE 10. *(Continued.)* **Experimental results of 12 algorithms on 30-dimensional CEC2014 problems.**

	Best	1.02E+03	3.89E+02	8.64E+02	3.76E+02	8.06E+02	9.20E+02	8.94E+02	9.96E+02	8.54E+02	7.87E+02	8.20E+02	**2.00E+00**
FC29	Mean	6.65E+06	2.14E+02	3.11E+04	2.15E+02	9.67E+02	3.31E+06	3.93E+06	3.08E+06	1.41E+03	1.29E+03	3.58E+03	**2.00E+02**
	SD	1.10E+07	1.02E+00	1.52E+04	1.39E+00	1.47E+02	5.39E+06	5.14E+06	4.99E+06	1.89E+02	5.91E+02	1.86E+02	**0.00E+00**
	Best	2.17E+03	2.12E+02	7.11E+03	2.13E+02	6.69E+02	1.25E+03	1.44E+03	9.91E+02	1.21E+03	7.75E+02	3.30E+03	**2.00E+00**
FC30	Mean	1.82E+04	7.00E+02	7.94E+03	3.87E+02	1.15E+03	3.35E+03	3.37E+03	6.47E+03	2.55E+03	1.93E+03	3.89E+03	**2.00E+02**
	SD	3.03E+04	8.54E+01	2.72E+03	6.82E+01	5.64E+02	1.35E+03	1.75E+03	3.43E+03	7.49E+02	5.26E+02	7.56E+01	**0.00E+00**
	Best	1.78E+03	5.98E+02	5.62E+03	2.92E+02	5.90E+02	1.62E+03	1.58E+03	3.37E+03	1.42E+03	1.15E+03	3.76E+03	**2.00E+00**

TABLE 11. **Statistical results of the multi-problem-based WRST between CBLSO vs. its competitors on 30-dimensional CEC2014 test functions.**

CBLSO vs.	R^+	R^-	p-value	$\alpha = 0.05$
FDR-PSO	319	116	0.027	−
FIPS	350	115	0.015	−
CLPSO	355	110	0.010	−
jDE	353	82	0.003	−
SaDE	362	103	0.007	−
TLBO	304.5	130.5	0.060	=
ETLBO	273	105	0.044	−
DGSTLBO	324	111	0.020	−
BSA	362	103	0.007	−
LBSA	362	103	0.007	−
SCA-PSO	292	143	0.110	=

TABLE 12. **The basic characteristics of the Wujiang cascade hydropower station.**

Reservoir items	HJD	DF	SFY	WJD	Units
Average inflow	150	345	385	502	m^3/s
Normal water level	1140	970	837	760	m
Dead water level	1076	936	822	720	m
Total storage	4.5	0.9	0.2	2.1	billion m^3
Regulation storage	3.4	0.5	0.07	1.4	billion m^3
Regulation ability	multi-year	seasonal	daily	seasonal	−
Installed capacity	600	695	600	1250	MW
Annual generation	1.6	2.4	2.0	4.1	billion kW·h
Power coefficient	8.5	8.35	8.5	8.17	−

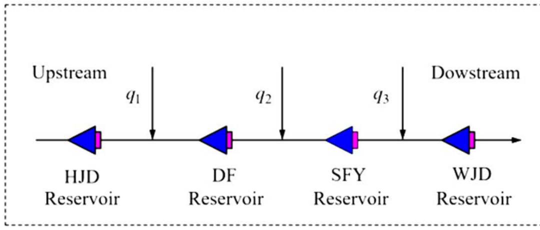


FIGURE 13. **The schematic diagram of the cascade reservoirs in the Wujiang River basin.**

has a series of equality and inequality constraints, which is very representative to validate the feasibility and effectiveness of CBLSO for solving real problems. Finally, the results obtained by CBLSO are validated and compared statistically with other well-known optimization algorithms, i.e., GA, variant of PSO, improved CS and standard LSO.

A. WUJIANG CASCADE HYDROPOWER STATIONS

The Wujiang River lies to the right bank of the upper Yangtze River and is the largest river in Guizhou province, southwest of China. The mainstream length of the Wujiang River is 1,037 km and the drainage area is 87,920 km^2 and 66,849 km^2 is in the district of Guizhou province, which accounts for 76% of the drainage area. The index map of the Wujiang River basin is presented in Fig.12. The rainfall of the Wujiang River is quite abundant and the annual average rainfall varies between 900 and 1,400 mm. Because of the representative continental monsoon climate, the rainfall presents a typical season characteristic that about 80% of the annual rainfall centralizes from April to August [92].

The Wujiang cascade hydropower station is one of thirteen large power generation bases in China. In this paper, 4 cascade reservoirs: Hong jiadu (HJD) reservoir, Dong feng (DF) reservoir, Suo fengying (SFY) reservoir, and Wu jiangdu (WJD) reservoir are selected as the object of the case study shown in Fig.12. Since the SFY is a daily regulation reservoir and the regulation storage is too small, the optimal dispatch model used in this paper is only applicable to the optimal dispatch of the HJD, DF, and WID reservoir are shown in Fig.13.

B. MODEL FORMULATION

The objective function of cascade hydropower stations adopted here is to maximize the annual energy production of the cascade hydropower stations, subject to some constraints (including technical and physical). Because the water level has an important impact on the storage capacity of the reservoir, therefore, the change of the outflow is indicated by the variation of the water level. Moreover, the output of hydropower station goes hand in hand with turbine release and hydraulic head determined by the water level of the period, dynamically. Based on the above discussion, once we can determine the upstream water level, we can calculate the outflow through Eq. (27). Furthermore, the downstream water level goes hand in hand with the outflow of the reservoir, if we enhance the outflow, the downstream water level will be higher, thus will lead the hydraulic head to be lower [91]. The objective function is expressed as follows:

$$\begin{aligned} \max f &= \max E = \max \sum_{i=1}^{S_{num}} \sum_{t=1}^{T} A_i H_{i,t} Q_{i,t} \Delta t \\ &= \max \sum_{i=1}^{S_{num}} \sum_{t=1}^{T} N_{i,t} \Delta t \end{aligned} \tag{22}$$

where E represents the annual energy production for the cascade hydropower stations (kW·h); S_{num} is the number for the hydropower stations; T is the size for the dispatch period; t denotes the current period; i denotes the current station; A_i is the power coefficient for the station i; $H_{i,t}$ is the average

TABLE 13. The optimal operation results for three scenarios obtained by GA, PSO, CS, ABC, GSA, GWO, LSO, ASO, SMA, and CBLSO.

Scheme index	Energy production (10^8 kW·h) Scenario 1 (Wet year)									
	GA	PSO	CS	ABC	GSA	GWO	LSO	ASO	SMA	CBLSO
1	122.1704	122.5011	122.5663	122.5663	120.6899	122.3403	122.5563	121.9713	122.5637	122.5620
2	122.4386	122.5624	122.5663	122.5663	120.2006	121.8731	122.4832	121.5270	122.5618	122.5200
3	122.1516	122.4931	122.5663	122.5663	120.6084	122.3557	122.5517	121.5017	122.5638	122.4842
4	121.0650	122.5613	122.5663	122.5663	120.4733	122.4783	122.4863	122.2169	122.5648	122.4847
5	121.7500	122.5658	122.5663	122.5663	121.1860	122.4520	122.5437	121.7846	122.5588	122.5319
6	121.8576	122.5630	122.5663	122.5663	119.4894	122.4267	122.4923	121.9961	122.5639	122.4843
7	121.7436	122.5657	122.5663	122.5663	120.3004	122.4195	122.4800	121.9212	122.5641	122.5568
8	122.3979	122.5294	122.5663	122.5663	118.6450	122.1488	122.4874	120.8139	122.5607	122.5072
9	122.4389	122.4892	122.5663	122.5663	120.4641	122.0404	122.4895	122.0306	122.4750	122.5367
10	121.9150	122.5631	122.5663	122.5663	119.7313	122.2148	122.5037	121.0823	122.5629	122.5001
Best	122.4389	122.5658	122.5663	122.5663	121.1860	122.4783	122.5563	122.2169	122.5648	122.5620
Average	121.9929	122.5394	122.5663	122.5663	120.1788	122.2750	122.5074	121.6845	122.5539	122.5168
Worst	121.0650	122.4892	122.5663	122.5663	118.6450	121.8731	122.4800	120.8139	122.4750	122.4842
SD	4.0309E−01	3.1222E−02	2.7335E−14	2.1460E−09	6.8351E−01	1.9065E−01	2.9007E−02	4.2749E−01	2.6382E−02	2.7933E−02
Scheme index	Energy production (10^8 kW·h) Scenario 2 (Normal year)									
	GA	PSO	CS	ABC	GSA	GWO	LSO	ASO	SMA	CBLSO
1	103.8291	103.9147	103.9147	103.9147	99.9764	103.1928	103.8472	103.7693	103.9132	103.8487
2	103.6733	103.8640	103.9147	103.9147	101.0557	103.5695	103.8529	103.5922	103.9143	103.8264
3	103.7887	103.7959	103.9147	103.9147	99.9685	100.5509	103.8480	101.7254	103.9357	103.8441
4	102.8035	103.9147	103.9147	103.9147	100.4436	102.2900	103.8450	103.5295	103.9022	103.8568
5	103.6833	103.8677	103.9147	103.9147	100.5527	102.8296	103.8320	103.2790	103.8913	103.8483
6	103.3217	103.9010	103.9147	103.9147	99.9562	103.0956	103.8417	103.6952	103.8880	103.8851
7	103.7783	103.9147	103.9164	103.9164	100.2714	101.8911	103.7243	103.1668	103.9285	103.8472
8	102.7886	103.9311	103.9147	103.9147	100.6526	103.2290	103.8435	103.5834	103.9147	103.8451
9	103.7032	103.9147	103.9353	103.9147	100.2779	102.2478	103.8468	103.7172	103.9116	103.8814
10	103.8430	103.8935	103.9147	103.9147	98.7535	103.2622	103.8523	102.6557	103.9113	103.8457
Best	103.8430	103.9311	103.9353	103.9147	101.0557	103.5695	103.8529	103.7693	103.9357	103.8851
Average	103.5213	103.9031	103.9169	103.9147	100.1909	102.6158	103.8334	103.2714	103.9083	103.8529
Worst	102.7886	103.8640	103.9147	103.9147	98.7535	100.5509	103.7243	101.7254	103.8872	103.8264
SD	3.8859E−01	2.0823E−02	6.1428E−03	1.1827E−10	5.8194E−01	8.5689E−01	3.6772E−02	6.0608E−01	1.5575E−02	1.6814E−02
Scheme index	Energy production (10^8 kW·h) Scenario 3 (Dry year)									
	GA	PSO	CS	ABC	GSA	GWO	LSO	ASO	SMA	CBLSO
1	99.6737	99.7686	99.8589	99.8589	97.6809	99.1933	99.7282	99.5924	99.8578	99.7393
2	99.4379	99.8519	99.8589	99.8589	97.4248	98.7576	99.5643	99.4269	99.8555	99.7030
3	99.7379	99.7959	99.8589	99.8589	97.7651	98.6948	99.6782	99.6220	99.8445	99.7282
4	99.5559	99.7977	99.8589	99.8223	97.8929	99.0246	99.6855	99.4882	99.7462	99.6907
5	99.5958	99.8347	99.8589	99.8589	97.2935	99.1431	99.7144	99.5505	99.7925	99.7174
6	99.2765	99.6819	99.8589	99.8300	98.0407	99.3091	99.7351	99.5295	99.7889	99.7279
7	99.6643	99.8420	99.8589	99.8589	98.0913	98.4824	99.7591	99.5669	99.7594	99.6987
8	99.7767	99.8094	99.8589	99.8589	96.6846	99.4747	99.6850	99.6700	99.7940	99.7106
9	99.6462	99.6686	99.8589	99.8589	97.6340	99.2425	99.7077	99.6193	99.7762	99.7168
10	99.3645	99.5600	99.8589	99.8231	96.7570	99.0841	99.6707	99.5835	99.7059	99.7071
Best	99.7767	99.8519	99.8589	99.8589	98.0913	99.4747	99.7591	99.6700	99.8578	99.7393
Average	99.5729	99.7611	99.8589	99.8488	97.5265	99.0406	99.6928	99.5649	99.7921	99.7140
Worst	99.2765	99.5600	99.8589	99.8223	96.6846	98.4824	99.5643	99.4268	99.7059	99.6907
SD	1.5587E−01	8.9679E−02	1.2711E−14	1.5596E−02	4.6642E−01	2.9129E−01	5.0454E−02	6.6906E−02	4.6858E−02	1.4180E−02

TABLE 14. Statistical results obtained by the multi-problem-based WSRT for CBLSO vs. GA, PSO, ABC, GSA, GWO, LSO, ASO, and SMA.

CBLSO vs.	Scenario 1 (Wet year)				Scenario 2 (Normal year)				Scenario 3 (Dry year)			
	R^+	R^-	p-value	$\alpha = 0.05$	R^+	R^-	p-value	$\alpha = 0.05$	R^+	R^-	p-value	$\alpha = 0.05$
GA	55	0	1.95300E−03	−	55	0	1.95300E−03	−	50	5	1.95310E−02	−
PSO	13	42	1.52344E−01	=	55	0	1.95300E−03	−	14	41	1.93359E−01	=
CS	55	0	1.95300E−03	−	55	0	1.95300E−03	−	0	55	1.95300E−03	+
ABC	55	0	1.95300E−03	−	55	0	1.95300E−03	−	0	55	1.95300E−03	+
GSA	55	0	1.95300E−03	−	55	0	1.95300E−03	−	55	0	1.95300E−03	−
GWO	55	0	1.95300E−03	−	55	0	1.95300E−03	−	55	0	1.95300E−03	−
LSO	34	21	5.56641E−01	=	41	14	1.93359E−01	=	43	12	1.30859E−01	=
ASO	55	0	1.95300E−03	−	55	0	1.95300E−03	−	55	0	1.95300E−03	−
SMA	6	49	2.73440E−02	+	0	55	1.95300E−03	+	1	54	3.90600E−03	+

hydraulic head for the station i in period t(m); $Q_{i,t}$ is the outflow for the station i in period t(m^3/s);Δt is the dispatch period (h); $N_{i,t}$ is the output of the station i in period t(kW).

C. CONSTRAINTS

In the cascade hydropower stations, the entire cascade hydropower stations should be subject to the constraints introduced by the interaction of hydropower stations or reservoirs, while each hydropower station should be subject to their constraints [93]. Specifically, the constraints can be expressed as follows:

(1) Water level constraint

$$\underline{Z}_{i,t} \le Z_{i,t} \le \bar{Z}_{i,t} \tag{23}$$

where $Z_{i,t}$ denotes the water level for the reservoir i in period t (m); $\underline{Z}_{i,t}$ and $\bar{Z}_{i,t}$ are lower and upper bounds of water level for the reservoir i in period t (m), respectively.

(2) Power output constraint

For each hydropower station

$$\underline{N}_{i,t} \le N_{i,t} \le \bar{N}_{i,t} \tag{24}$$

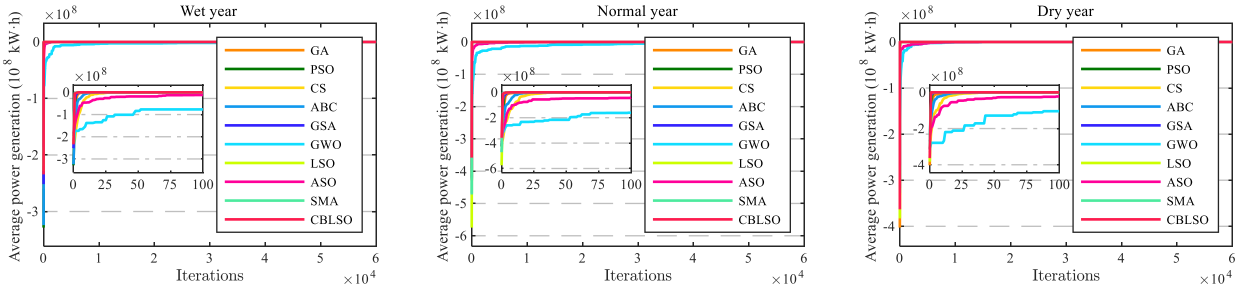


FIGURE 14. The convergence graphs of the average power generation obtained by all algorithms on Wet year, Normal year, and Dry year.

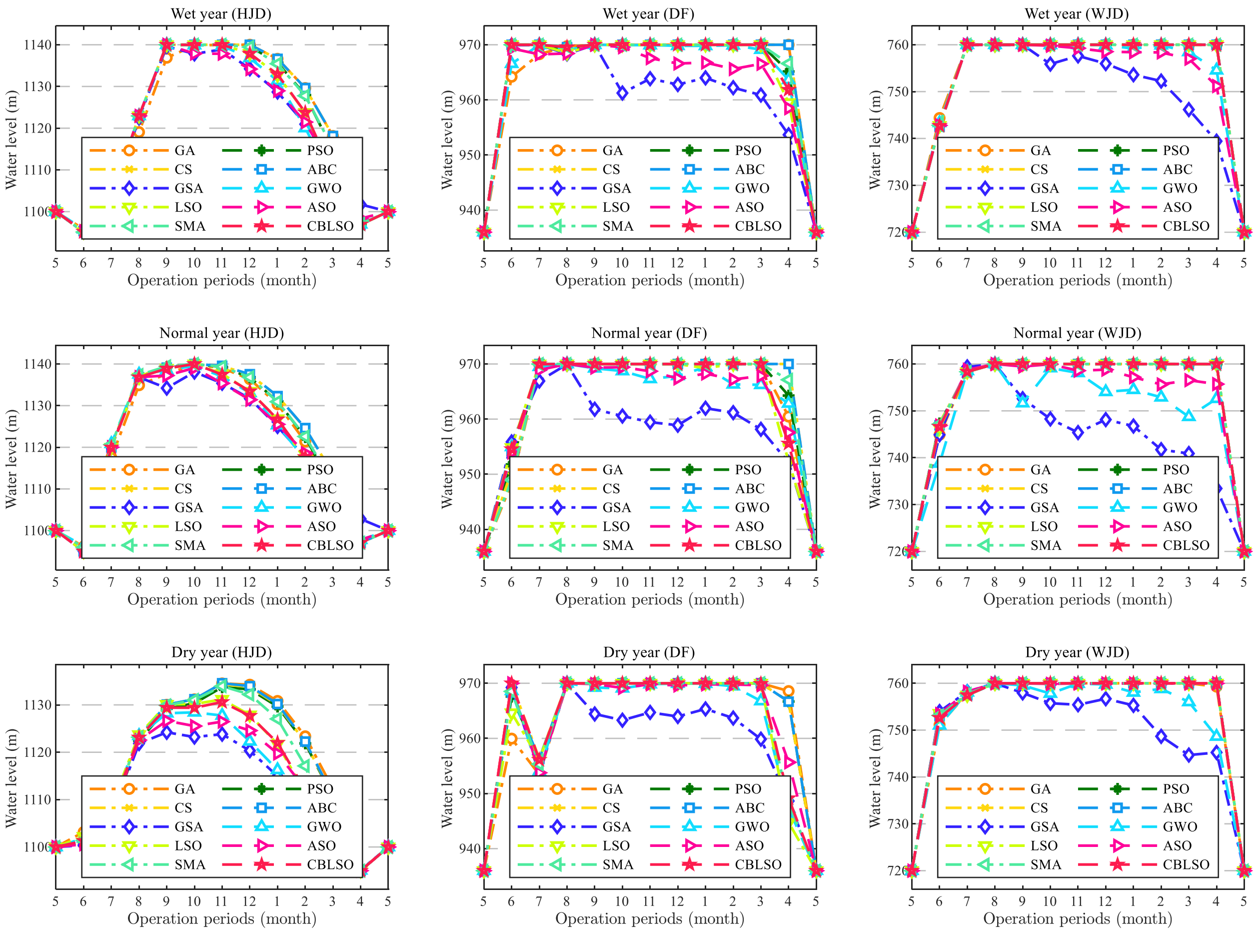


FIGURE 15. The monthly water level processes obtained by all algorithms on Wet year, Normal year, and Dry year.

For the whole system

$$\underline{N}_t \le \sum_{i=1}^{S_{num}} N_{i,t} \le \bar{N}_t \tag{25}$$

where $\underline{N}_{i,t}$ and $\bar{N}_{i,t}$ are lower and upper bounds of output for the station i in period t (kW), respectively; $\underline{N}_t$ and $\bar{N}_t$ are lower and upper bounds of output for the entire cascade hydropower station in period t (kW), respectively.

(3) Outflow constraint

$$\underline{Q}_{i,t} \le Q_{i,t} \le \bar{Q}_{i,t} \tag{26}$$

where $\underline{Q}_{i,t}$ and $\bar{Q}_{i,t}$ are lower and upper bounds of outflow for the reservoir i in period t (m^3/s), respectively.

(4) Water balance equation

$$V_{i,t+1} = V_{i,t} + \left(I_{i,t} - Q_{i,t}\right) \cdot \Delta t \tag{27}$$

where $V_{i,t}$ is the volume for the reservoir i storage at the beginning of period t (m^3);$V_{i,t+1}$ is the volume for the reservoir i storage at the end of period t (m^3); $I_{i,t}$ is the inflow for the reservoir i in period t (m^3/s); $Q_{i,t}$ is the outflow for the reservoir i in period t(m^3/s).

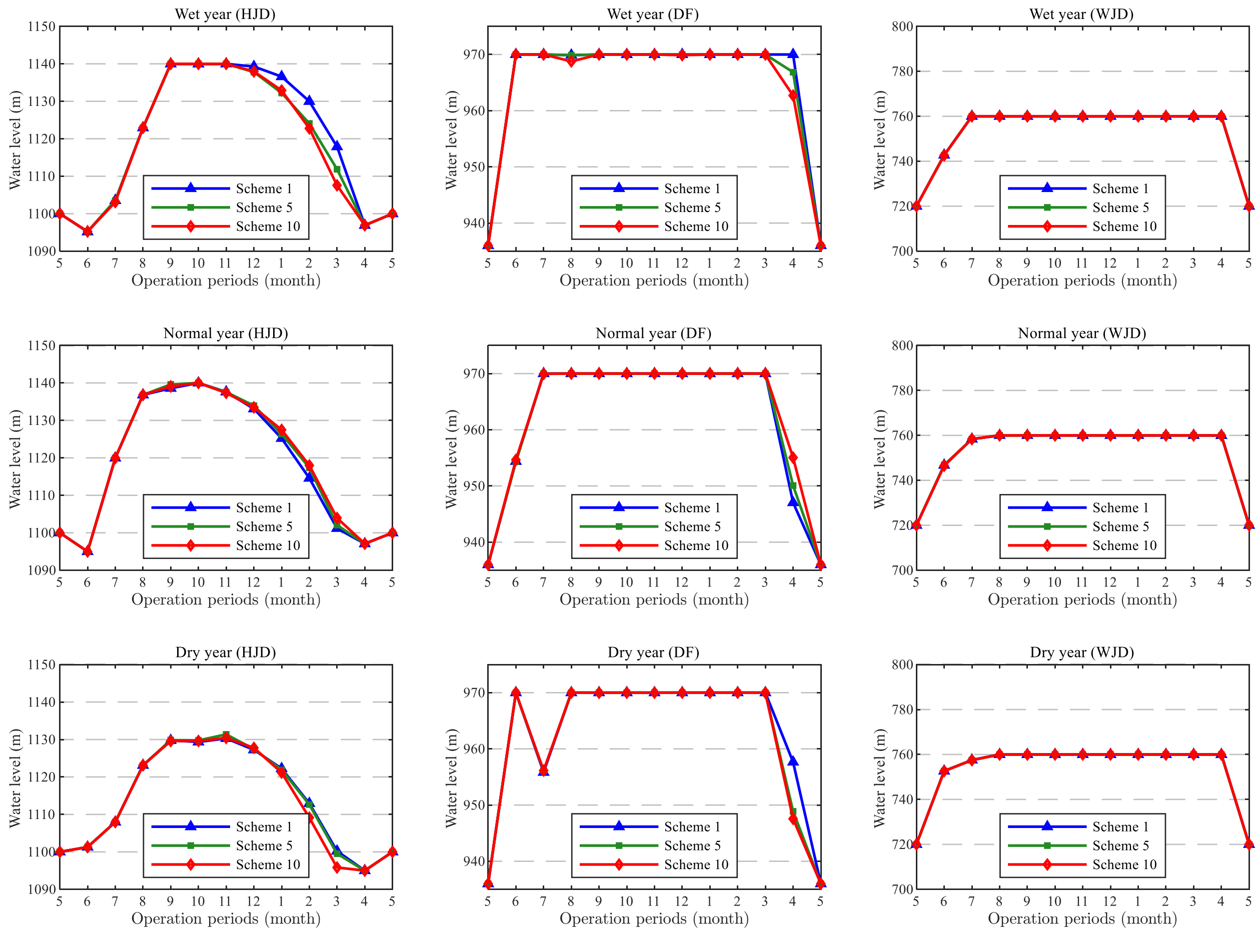


FIGURE 16. The monthly water level processes obtained by CBLSO on Wet year, Normal year, and Dry year.

(5) Hydraulic connection equation

$$I_{i+1,t} = Q_{i,t} + q_{i,t} \tag{28}$$

where α_f is the inflow for the reservoir $t+1$ in period t(m^3/s); $Q_{i,t}$ is the outflow for the reservoir $t+1$ in period t (m^3/s); $q_{i,t}$ is the interval inflow into the reservoir $t+1$ in period t (m^3/s).

(6) Boundary constraint

$$Z_{i,1} = Z_{i,b}, Z_{i,T+1} = Z_{i,e} \tag{29}$$

where $Z_{i,b}$ is the initial water level for the reservoir i(m); $Z_{i,e}$ is the final water level for the reservoir i(m).

Water levels, power outputs, reservoir inflows, and reservoir outflows are all variables that will be calculated, Historical monthly inflow sequences of reservoirs, curve of reservoir water level- reservoir storage capacity, curve of reservoir outflow- reservoir downstream water level and values of control parameters that used for reservoirs or hydropower stations are all known. Furthermore, we also assume that the generality loss is ignored and the evaporation loss of water can be canceled out by rainfall.

D. CONSTRAINTS HANDLING METHOD

The critical step in developing a successful application for efficiently solving the optimal dispatch of cascade hydropower stations is appropriately handling equality constraints and inequality constraints. In this section, to effectively handle the complex constraints of the optimal dispatch of cascade hydropower stations without degrading the optimization method's computation efficiency, a novel constraint handling method according to the characteristics of the different constraints and the different power generation mechanisms of hydropower stations or reservoirs is designed.

Water level constraints shown in Eq. (23) are inequality constraints. For these constraints, the handling method considering the feasible boundaries is carried out as follows:

$$Z_{i,t} = \begin{cases} Z_{i,t}^{min} & if\ Z_{i,t} \leq Z_{i,t}^{min} \\ Z_{i,t}^{max} & if\ Z_{i,t} \geq Z_{i,t}^{max}, \end{cases} \quad i = 1, 2, \ldots, S_{sum}, \quad t = 1, 2, \ldots, T \tag{30}$$

Power output constraints shown in Eqs. (24) and (25) and outflow constraints shown in Eq. (26) are inequality constraints. For power output constraints Eq. (24) and outflow

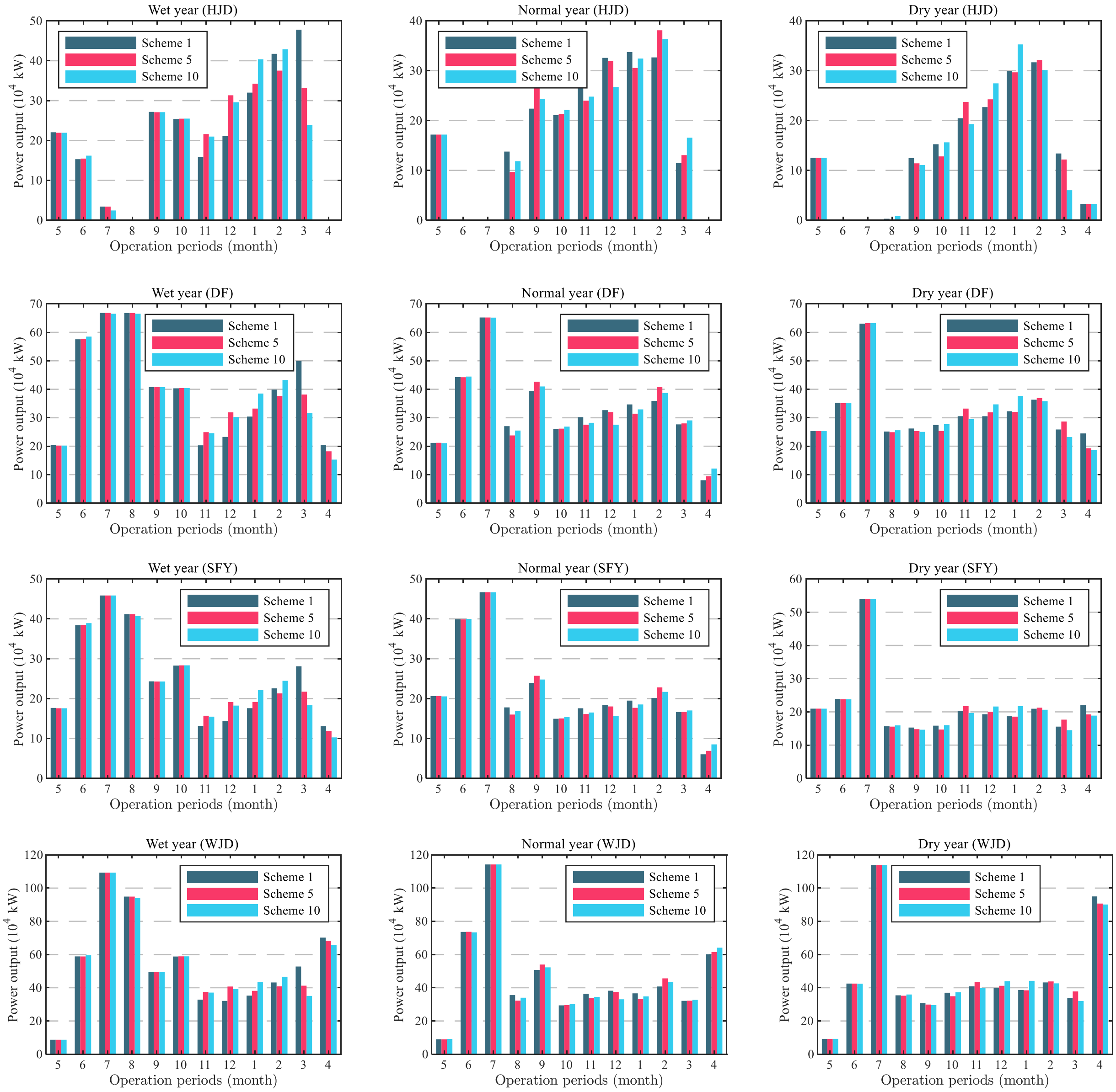


FIGURE 17. The monthly power output processes obtained by CBLSO on Wet year, Normal year, and Dry year.

constraints Eq. (26), the handling method (see Eq. (31)) combined with water balance equation constraints shown in Eq. (27), hydraulic connection equation constraints shown in Eq. (28), and initial and terminal upstream water level constraints shown in Eq. (29) will be implemented to adjust the values of the infeasible solutions to meet the feasible boundaries. For power output constraints Eq. (25), the handling method shown in Eq. (32) is conducted to adjust the values of the infeasible solutions to meet the feasible boundaries.

$$N_{i,t} = \begin{cases} M_1\left(Q_{i,t} - Q_{i,t}^{min}\right) & if\ -\infty < Q_{i,t} \leq Q_{i,t}^{min} \\ K_i Q_{i,t} H_{i,t} & if\ Q_{i,t}^{min} \leq Q_{i,t} \leq Q_{i,t}^{max} \\ M_2\left(Q_{i,t}^{max} - Q_{i,t}\right) & if\ Q_{i,t}^{max} \leq Q_{i,t} < +\infty \end{cases}$$

$$Q_{i,t} = I_{i,t} - \frac{V_{i,t} - V_{i,t-1}}{\Delta t} - S_{i,t} + \sum_{k=1}^{M_i}\left(Q_{k,t-\tau_{ki}} + S_{k,t-\tau_{ki}}\right)$$

$$i = 1, 2, \ldots, S_{sum}, t = 1, 2, \ldots, T \tag{31}$$

$$\sum_{i=1}^{S_{sum}} N_{i,t} = \begin{cases} \sum_{i=1}^{S_{sum}} N_{i,t} - M_2\left(N_t^{min} - \sum_{i=1}^{S_{sum}} N_{i,t}\right)^2 & if\ \sum_{i=1}^{S_{sum}} N_{i,t} \leq N_t^{min} \\ \sum_{i=1}^{S_{sum}} N_{i,t} & if\ \sum_{i=1}^{S_{sum}} N_{i,t} > N_t^{min} \end{cases}$$

$$i = 1, 2, \ldots, S_{sum}, t = 1, 2, \ldots, T \tag{32}$$

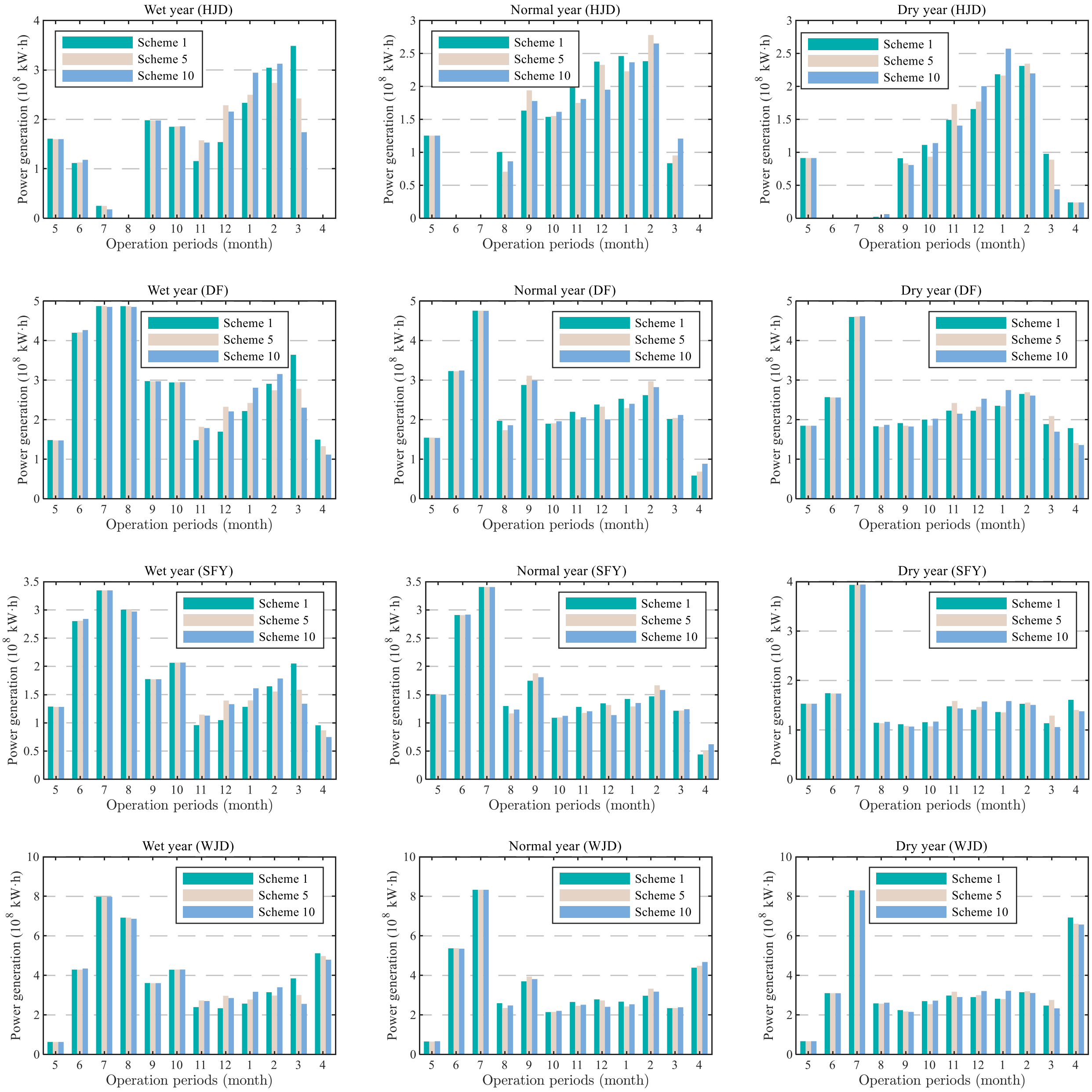


FIGURE 18. The monthly generation processes obtained by CBLSO on Wet year, Normal year, and Dry year.

Through the designed constraint handling method, it is possible to guarantee that the feasible solutions always have priorities than the infeasible solutions. What's more, when the solutions are out of the feasible regions, Eq. (32) can effectively evaluate the distance between the infeasible solutions and the bounds of the feasible regions, allowing the search to move quickly to the feasible regions.

E. SIMULATION SCENARIOS

Suppose the four reservoirs studied here are considered as a whole. The guaranteed power output for Wujiang cascade hydropower station should be higher than the minimum of the power output of the entire cascade hydropower station (680 MW). Because the interval inflow is quite abundant, the ecological runoff will be negligible. The basic characteristics of the Wujiang cascade hydropower station are listed in Table 12.

The simulation about the optimal dispatch of Wujiang cascade hydropower station for three scenarios is done on a monthly basis. The dispatch periods are 12 months, which range from May of the current year to April of the next year. Based on the hydrological frequency analysis of historical monthly inflow sequence data, we choose three typical years: 75% guaranteed rates of water supply (from 1951.5 to 1952.4), name as the wet year (Scenario 1), 50% guaranteed rates of water supply (from 1985.5 to 1986.4), name

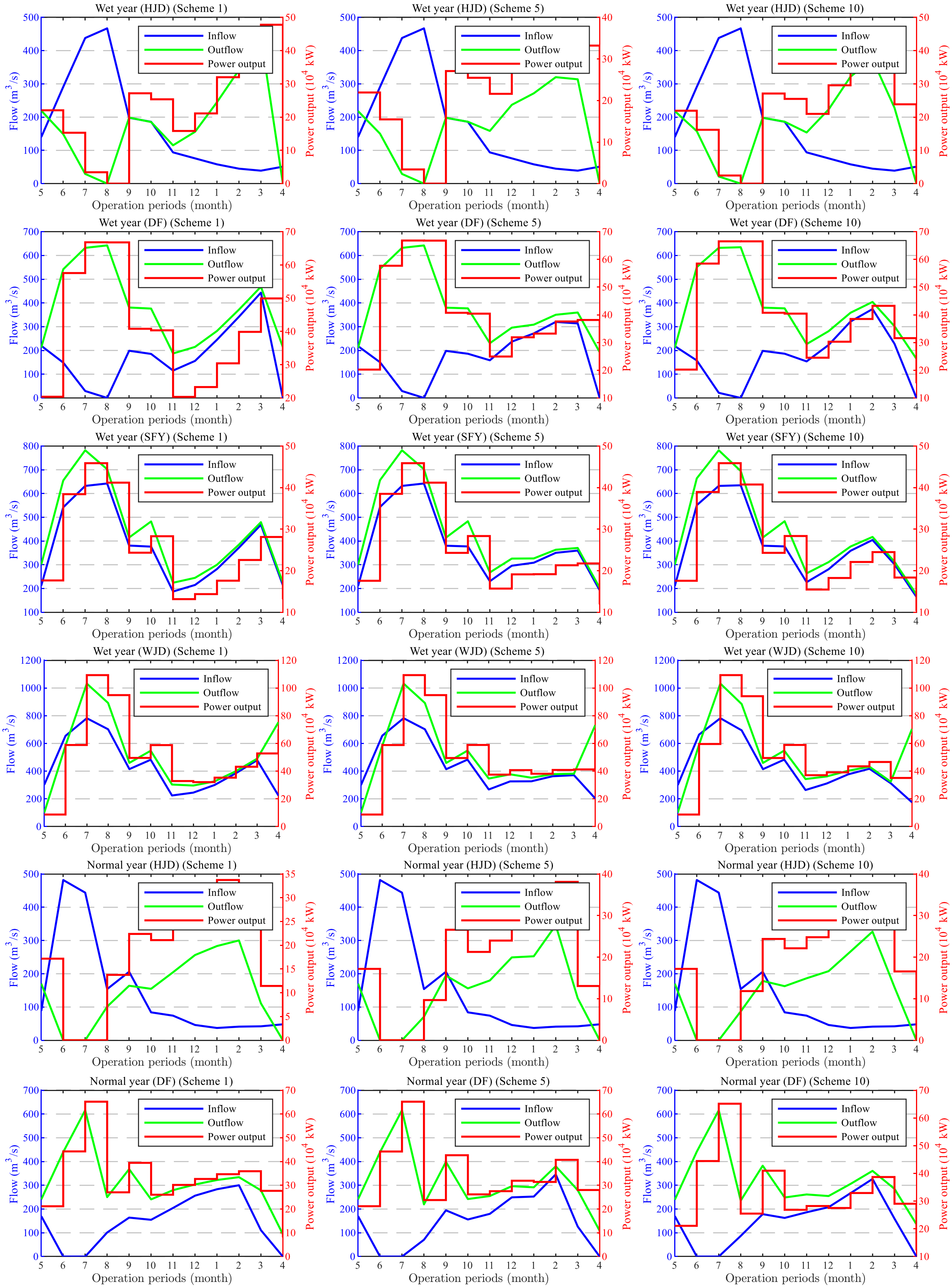


FIGURE 19. The discharge and power output processes obtained by CBLSO on Wet year, Normal year, and Dry year.

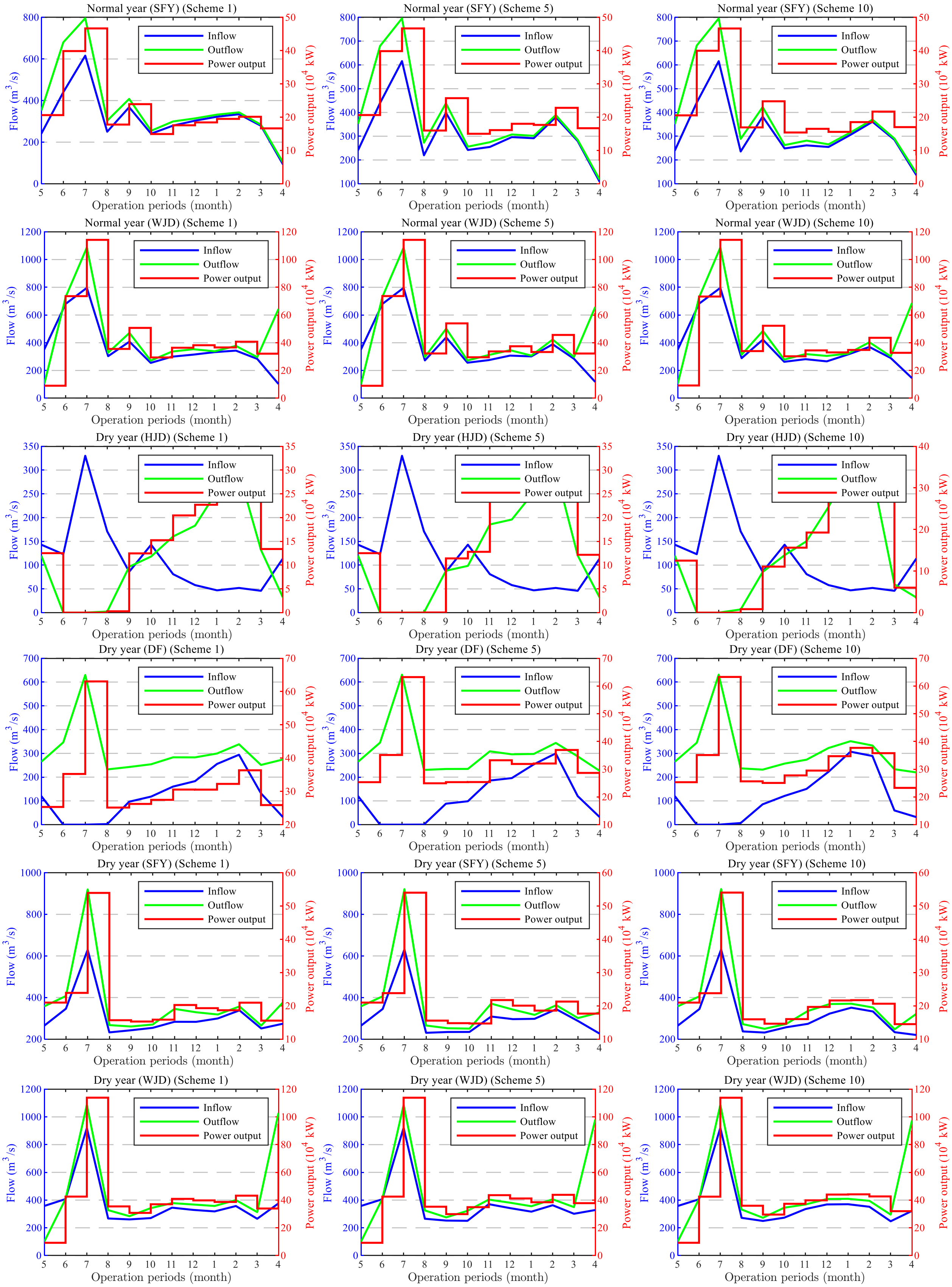


FIGURE 19. *(Continued.)* **The discharge and power output processes obtained by CBLSO on Wet year, Normal year, and Dry year.**

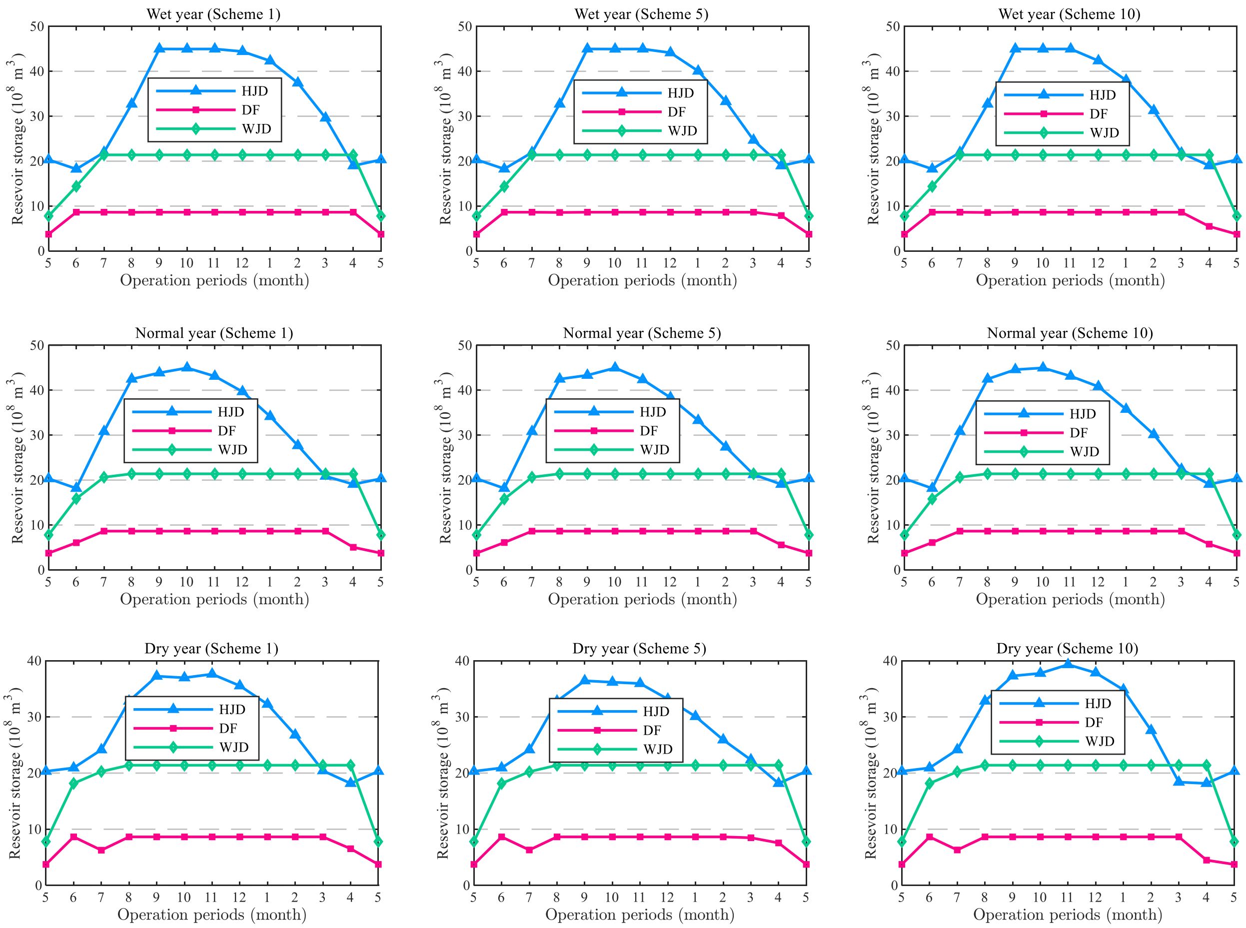


FIGURE 20. The monthly reservoir storages for each reservoir obtained by CBLSO on Wet year, Normal year, and Dry year.

as the normal year (Scenario 2), 25% guaranteed rates of water supply (from 1963.5 to 1963.4), name as the dry year (Scenario 3).

F. PARAMETER SETTINGS

The CBLSO along with the comparison algorithms include GA [23], PSO [29], [30], ABC [31], CS [32], GSA [40], GWO [34], LSO [35], ASO [41], [42], and SMA [54] are applied to solve the optimal dispatch problem of cascade hydropower stations. For all algorithms, the population size n is set to be 100, and the maximum number of iterations t_{max} is set to be 60,000. The results composed of best (Best), average (Average), worst (Worst), and standard deviation (Std) values for 10 independent runs on three scenarios (wet year, normal year, and dry year) are presented in Table 13, respectively. In addition, the details of default parameter settings for GA, PSO, ABC, CS, GSA, and GWO, ASO, and SMA are kept the same as in Subsection IV. The details of default parameter settings for LSO, and CBLSO are recommended by [64].

G. RESULTS ANALYSIS AND DISCUSSION

From Table 13, we can find that, compared with other peer algorithms, the Average and Std values provided by the CBLSO algorithm are quite competitive, indicating its effective and steady performance. In detail, CBLSO presents the Average values of generation on three scenarios are 122.5168, 103.8529, and 99.7140 (10^8 kW·h), respectively. By observing Table 13, we can find that the results on three scenarios obtained by CBLSO are also obviously better than the results provided by LSO. In addition, the worst solutions obtained by CBLSO on three scenarios are also relatively the least, indicating that the premature convergence's influence on CBLSO is limited owing to its effective structure and updating strategies. What's more, it is worth mentioning that the inflow of Scenario 3 is limited, making it more challenging to allocate the water resources rationally than Scenarios 1 and 2. However, the results yielded by CBLSO are comparable to the results yielded by other comparison algorithms, suggesting the superior search ability of CBLSO. To arrive at more precise conclusions, the WSRT at a significant level of $\alpha = 0.05$ is used for statistical analysis is conducted, and the statistical results are shown in Table 14. It can be seen from this table that although CBLSO cannot outperform GA, CS, ABC, GSA, GWO, and ASO on Scenario 1, GA, PSO, CS, ABC, GSA, GWO, and ASO on Scenario 2, and GA, GSA, GWO, and ASO on Scenarios 3, CBLSO shows a statistical

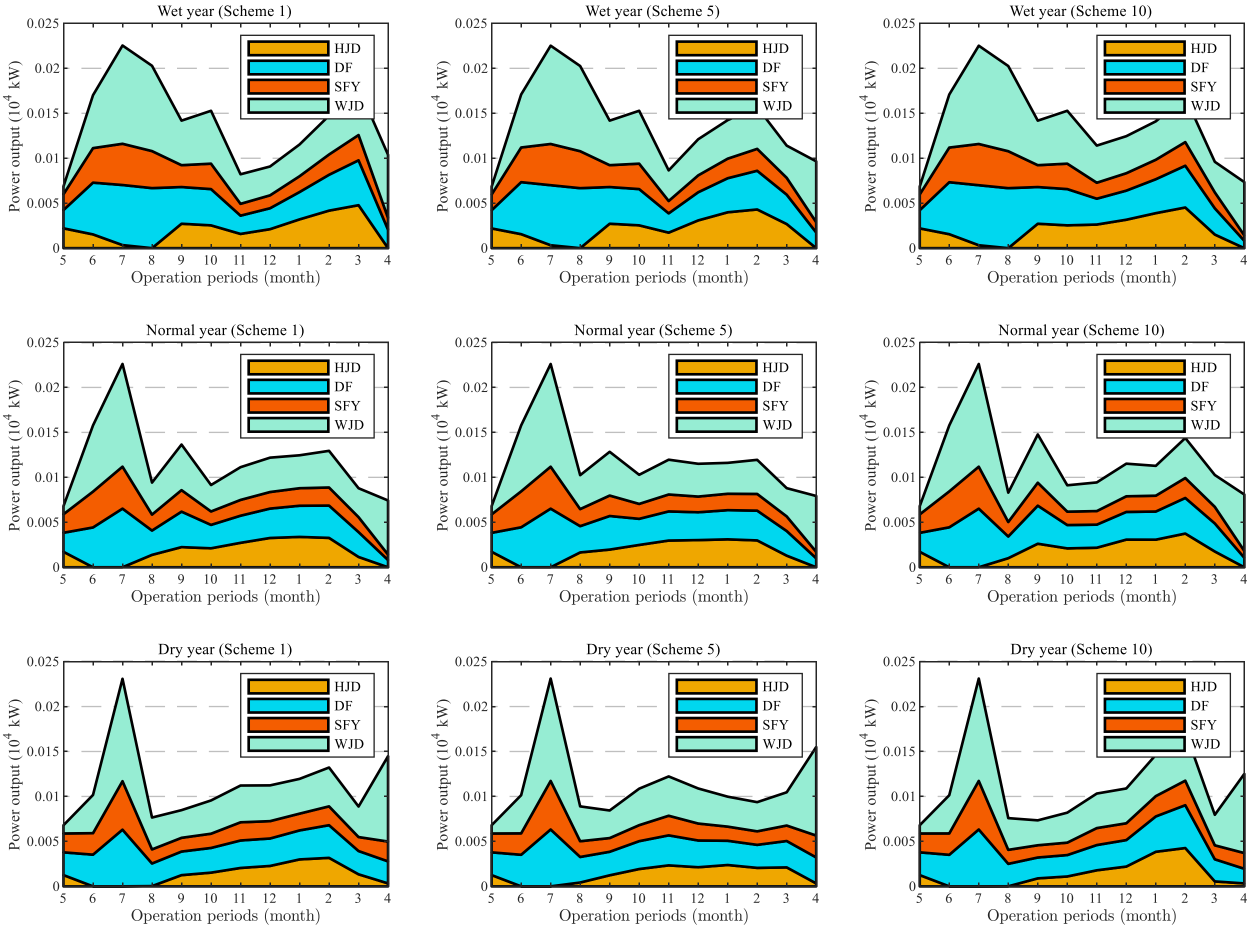


FIGURE 21. The stack of colors of monthly power output for each reservoir obtained by CBLSO on Wet year, Normal year, and Dry year.

significance over SMA on the three scenarios, CS and ABC on Scenario 3, and, achieves comparable results with PSO on Scenario 3 and Scenario 3, with LSO on all three scenarios. implying that CBLSO is capable of performing comparably well to other comparison algorithms.

To intuitively show each optimization algorithm's convergence characteristic and check whether the constraints of the problem are met or not, the average best-so-far power generations for three scenarios (wet year, normal year, and dry year) of all algorithms, the monthly mean water level processes of 10 schemes for the HJD reservoir, the DF reservoir, and the WJD reservoir obtained from three scenarios (wet year, normal year, and dry year) of CBLSO are plotted in Fig. 14 and Fig. 15, respectively. Meanwhile, to present the rationality and the correctness of the operation results, the monthly water level processes, the monthly power output processes, the monthly power generation processes, the discharge and power output processes, the monthly reservoir storages of each reservoir, and the stack of colors of monthly power out for each reservoir on 3 representative schedule results (i.e., scheme 1, scheme 5, and scheme 10) obtained by DMSDL-HHO are shown in Figs. 16-21, respectively.

Firstly, we describe the results obtained in the convergence graphs of the average power generation (Fig. 14). From Fig. 14, we can find that CS, ABC, GWO, and ASO converge slowly in the early stage of the search process for all three scenarios, while CBLSO presents a different convergence characteristic, i.e., it consistently converges in the early stage of the search process and finally achieves the best solution in the early half stage of the search process for all three scenarios, indicating that it is more effective. In particular, the convergence speed of CBLSO is faster than almost all other peer algorithms in the early stage of the search process. Thus, we can conclude that CBLSO can effectively alleviate the premature convergence and significantly enhance its optimization performance during its execution compared with other peer algorithms.

Next, we compare the algorithms in terms of the monthly water level processes obtained by CBLSO and comparison algorithms (Figs. 15 and 16). From Figs. 15 and 16, we can find that the water levels of each reservoir all locate in the rational ranges, indicating that the constraint handling method designed in this study can meet the practical requirement of the multi-reservoir system. It also can be seen from Figs. 15 and 16 that the water level of the HJD reservoir decreases suddenly and then gradually rises in the initial stage of the dispatch periods. This is because the HJD reservoir's inflow in the initial stage of the dispatch periods is quite limited (only 88 m^3/s). To meet the guarantee power output requirement of the whole cascade system (680 MW), the HJD reservoir must increase the outflow to compensate the downstream reservoirs, which results in a lower water level of itself. However, in the following dispatch periods, the inflows of the three reservoirs are abundant enough. In the precondition of meeting the guarantee power output requirement of the whole system, the HJD reservoir's outflow can be decreased to increase the storage volume of its own. Thus, the reservoir can use its storage volume to generate more energy in the following dispatch periods. In addition, it also can use the water stored during the wet seasons to compensate for the downstream reservoirs during the dry seasons. Meanwhile, from Fig. 16, we can see that the water levels of the three reservoirs are nearly identical for the wet year and are slightly different for the normal year and the dry year. The reason may be that the water supply of the wet year is abundant, which is helpful for optimal dispatch, but the water supply of the normal year and the dry year is relatively short, which make it more difficult to allocate the water resources than the wet year.

Thirdly, we pay attention to the results obtained in the power output processes and the power generation processes (Figs. 17 and 18). The optimal results indicate that the power output and the power generation on April, July, and August for the wet year, on April, June, and July for the normal year, and on June, July, and August for the dry year are near zero. The reason may be that the HJD reservoir's inflows in these dispatch periods are abundant, the reservoir needs to decrease the outflows of itself to rapidly increase its storage volume, which not only can utilize the higher hydraulic head to generate more output and more power generation during the following dispatch period but also can utilize the stored water resources to compensate for the downstream reservoirs in the dry periods. This phenomenon can also be explained by the variations in the water levels shown in Figs. 15 and 16.

We now analyze the results obtained in the discharge and power output processes (Fig. 19). From Fig. 19, we can find that the water discharge rate and the power output are all within the boundaries of constraints, suggesting that the proposed constraint handling method is reasonable and effective. In addition, the changes of the water discharge rate processes and the power output processes are all following the variations of the water levels, the power out, and the power generation, which further indicating the rationality and effectiveness of the proposed constraint handling method.

Finally, we compare the results obtained in the monthly reservoir storages and the Stack of colors of monthly power output for each reservoir (Figs. 20 and 21). It is shown by Fig. 20 that the monthly reservoir storages of the 3 schemes for Wet year, Normal year, and Dry year are all within the boundaries of the constraints. What's more, the results displayed in Fig. 21 show that the monthly power outputs all meet the whole system's guarantee power output requirement. That is to say, the dispatch results provided by CBLSO are reasonable, and the constraint handling method proposed in this study is effective.

VI. CONCLUSION

In this paper, a novel LSO with a chaotic mutation strategy and boundary mutation strategy (CBLSO) is presented to enhance the performance of LSO from the viewpoint of mutation strategy. Since some shortcomings of the standard LSO, such as premature convergence and limited global search ability, CBLSO is designed to address these issues. In this proposed algorithm, a chaotic mutation strategy based on chaotic cubic mapping is used to effectively improve the exploration ability for avoiding premature convergence and a boundary mutation strategy based on the concept of multilevel parallel is designed to manage boundary constraint violations for keeping and maintaining the population diversity. A comparative experiment is implemented to explore the effectiveness of the chaotic mutation strategy and the boundary mutation strategy. When compared with three variants of LSO algorithms (standard LSO, CLSO, and BLSO), the experimental results demonstrate that the effect of the chaotic mutation strategy is more evident than the effect of the boundary mutation strategy but introduces the two strategies in our CBLSO can achieve better positive effects.

The CBLSO is evaluated on 56 test functions and compared with 8 state-of-the-art algorithms: GA, PSO, CS, ABC, GSA, GWO, ASO, and SMA. The experimental results in terms of some often-used performance indicators and statistical analysis illustrate that our CBLSO is the most salient algorithm since it performs best on most of the test problems. The experiment results of the robustness analysis, the comprehensive significance analysis, and the study of high-dimension functions demonstrate that CBLSO can provide comparable results with remarkable convergence behavior compared to the other reported algorithms. Sequentially, the time complexity analysis of CBLSO shows that it is the same computational efficiency in contrast to standard LSO. Moreover, the results obtained by CBLSO on CEC 2014 are further compared with those provided by some well-known algorithms. It is shown by the experiment results that the proposed CBLSO performs better than or at least comparable to other algorithms and can be taken as a promising alternative optimization tool to solve GOPs. Finally, CBLSO is applied to the optimal dispatch problem of cascade hydropower stations to investigate its great potential for real-world applications. The obtained results show that the algorithm can produce competitive results when compared with standard LSO and other outstanding algorithms.

Summarizing, according to the test problems, the robustness analysis, the comprehensive significance analysis, the comparative experiments and the real-world application, the CBLSO is the most promising algorithm in our study. In our future work, the following future research can be focused on: (1) LSO can be further improved according to employ other strategies or operators used by other algorithms. (2) Other real-world engineering and practical problems can be used to determine whether the CBLSO can present good results with the parameter settings provided in this study. (3) Other heuristic algorithms should be reinforced by the chaotic mutation strategy and boundary mutation strategy used in this paper. (4) The CBLSO can be developed for solving optimization problems with nonlinear constraints. (5) Extending the proposed CBLSO for handling multi-objective optimization problems and many-objective optimization problems is also meaningful work.

APPENDIX

The test functions are described by the following format:

No. Name (Symbol, Characteristic (C): Uni-modal (U), Multi-modal (M), Separable (S), Non-separable (N)).

Description of the benchmark functions:

A. US TEST FUNCTIONS (F01-F04)

1. Step function (F01, US)

$$f(x) = \sum_{i-1}^{D} (\lfloor x_i + 0.5 \rfloor)^2, -100 \le x_i \le 100, \quad i = 1, 2, \cdots, 30$$
$$f_{min} = f\left(x_i^*\right) = 0, -0.5 \le x_i^* < 0.5, \quad i = 1, 2, \cdots, 30$$

2. Sphere function (F02, US)

$$f(x) = \sum_{i=1}^{D} x_i^2, -100 \le x_i \le 100, \quad i = 1, 2, \cdots, 30$$
$$f_{min} = f(0, 0, \cdots, 0) = 0$$

3. Sum Squares function (F03, US)

$$f(x) = \sum_{i=1}^{D} i \cdot x_i^2, -10 \le x_i \le 10, \quad i = 1, 2, \cdots, 30$$
$$f_{min} = f(0, 0, \cdots, 0) = 0$$

4. Quartic function (F04, US)

$$f(x) = \sum_{i=1}^{D} i \cdot x_i^4 + random\,[0, 1), \quad -1.28 \le x_i \le 1.282, i = 1, 2, \cdots, 30$$
$$f_{min} = f(0, 0, \cdots, 0) = 0$$

B. UN TEST FUNCTIONS (F05-F17)

1. Beale function (F05, UN)

$$f(x) = (1.5 - x_1 + x_1 x_2)^2 + \left(2.25 - x_1 + x_1 x_2^2\right)^2 + \left(2.625 - x_1 + x_1 x_2^3\right)^2$$
$$-4.5 \le x_i \le 4.5, i = 1, 2$$
$$f_{min} = f(3, 0.5) = 0$$

2. Easom function (F06, UN)

$$f(x) = -\cos(x_1)\cos(x_2)\exp\left(-(x_1 - \pi)^2 - (x_2 - \pi)^2\right), \quad -100 \le x_i \le 100, i = 1, 2$$
$$f_{min} = f(\pi, \pi) = -1$$

3. Matyas function (F07, UN)

$$f(x) = 0.26\left(x_1^2 + x_2^2\right) - 0.48 x_1 x_2, \quad -10 \le x_i \le 10, i = 1, 2$$
$$f_{min} = f(0, 0) = 0$$

4. Colville function (F08, UN)

$$f(x) = 100\left(x_1^2 - x_2\right)^2 + (x_1 - 1)^2 + (x_3 - 1)^2 + 90\left(x_3^2 - x_4\right)^2 + 10.1\left((x_2 - 1)^2 + (x_4 - 1)^2\right) + 19.8(x_2 - 1)(x_4 - 1),$$
$$-10 \le x_i \le 10, \quad i = 1, 2, 3, 4$$
$$f_{min} = f(1, 1, 1, 1) = 0$$

5 and 6. Trid (Trid 6, Trid 10) function (F09 and F10, UN)

$$f(x) = \sum_{i=1}^{D} (x_i - 1)^2 - \sum_{i=2}^{D} x_i x_{i-1}, -D^2 \le x_i \le D^2, \quad i = 1, 2, \cdots, \text{ 6 or 10}$$
$$f_{min} = f(i(n + 1 - i)) = -n(n + 4)(n - 1)/6, \quad i = 1, 2, \cdots, \text{ 6 or 10}$$

7. Zakharov function (F11, UN)

$$f(x) = \sum_{i=1}^{D} x_i^2 + \left(\sum_{i=2}^{D} 0.5 i x_i\right)^2 + \left(\sum_{i=2}^{D} 0.5 i x_i\right)^4, \quad -5 \le x_i \le 10, i = 1, 2, \cdots, 10$$
$$f_{min} = f(1, 1, \cdots, 1) = 0$$

8. Powell function (F12, UN)

$$f(x) = \sum_{i=1}^{D/4} \left[\begin{array}{l}(x_{4i-3} + 10x_{4i-2})^2 + 5(x_{4i-1} - x_{4i})^2 \\ + (x_{4i-2} + 2x_{4i-1})^4 + 10(x_{4i-3} - x_{4i})^4\end{array}\right], \quad -4 \le x_i \le 5, \quad i = 1, 2, \cdots, 24$$
$$f_{min} = f(0, 0, \cdots, 0) = 0$$

9. Schwefel 1.2 function (F13, UN)

$$f(x) = \sum_{i=1}^{D}\left(\sum_{j=1}^{i} x_j\right)^2, \quad -100 \le x_i \le 100, \quad i = 1, 2, \cdots, 30$$
$$f_{min} = f(0, 0, \cdots, 0) = 0$$

10. Schwefel 2.21 function (F14, UN)

$$f(x) = max\{|x_i|, 1 \le i \le D\}, \quad -100 \le x_i \le 100, i = 1, 2, \cdots, 30$$
$$f_{min} = f(0, 0, \cdots, 0) = 0$$

11. Schwefel 2.22 function (F15, UN)

$$f(x) = \sum_{i=1}^{D} |x_i| + \prod_{i=1}^{D} |x_i|, \quad -10 \le x_i \le 10, \quad i = 1, 2, \cdots, 30$$
$$f_{min} = f(0, 0, \cdots, 0) = 0$$

12. Rosenbrock function (F16, UN)

$$f(x) = \sum_{i=1}^{D-1} \left[100\left(x_{i+1} - x_i^2\right)^2 + (x_i - 1)^2\right], \quad -30 \le x_i \le 30, \quad i = 1, 2, \cdots, 30$$
$$f_{min} = f(0, 0, \cdots, 0) = 0$$

13. Dixon-Price function (F17, UN)

$$f(x) = (x_1 - 1)^2 \sum_{i=2}^{D} i\left(2x_i^2 - x_{i-1}\right)^2, \quad -10 \le x_i \le 10, \quad i = 1, 2, \cdots, 30$$
$$f_{min} = f\left(2^{-(2^i-2)/2^i}\right) = 0, \quad i = 1, 2, \cdots, 30$$

C. MS TEST FUNCTIONS (F18-F27)

1. Branin function (F18, MS)

$$f(x) = \left(x_2 - \frac{5.1x_1}{4\pi^2} + \frac{5x_1}{\pi} - 6\right)^2 + 10\left(1 - \frac{1}{8\pi}\right)\cos x_1 + 10, \quad -5 \le x_1 \le 10, 0 \le x_2 \le 15$$
$$f_{min} = f\left(x^*\right) = 0.397887, \quad x^* = (-\pi, 12.275), (\pi, 2.275), (3\pi, 2.475)$$

2. Bohachevsky1 function (F19, MS)

$$f(x) = x_1^2 + 2x_2^2 - 0.3\cos(3\pi x_1) - 0.4\cos(4\pi x_2) + 0.7, \quad -100 \le x_i \le 100, i = 1, 2$$
$$f_{min} = f(0, 0) = 0$$

3. Booth function (F20, MS)

$$f(x) = (x_1 + 2x_2 - 7)^2 + (2x_1 + x_2 - 5)^2, \quad -10 \le x_i \le 10, \quad i = 1, 2$$
$$f_{min} = f(1, 3) = 0$$

4. Rastrigin function (F21, MS)

$$f(x) = \sum_{i=1}^{D}\left[x_i^2 - 10\cos(2\pi x_i) + 10\right], \quad -5.12 \le x_i \le 5.12, \quad i = 1, 2, \cdots, 30$$
$$f_{min} = f(0, 0, \cdots, 0) = 0$$

5. Non-continuous Rastrigin function (F22, MS)

$$f(x) = \sum_{i=1}^{D}\left[y_i^2 - 10\cos(2\pi y_i) + 10\right]$$

where

$$y_i = \begin{cases} x_i, & |x_i| < 1/2 \\ round(2x_i)/2, & |x_i| \ge 1/2, \end{cases} \quad -5.12 \le x_i \le 5.12, \quad i = 1, 2, \cdots, 30$$
$$f_{min} = f(0, 0, \cdots, 0) = 0$$

6-7. Schwefel 2.26 function (F23 and F24, MS)

$$f(x) = -\sum_{i=1}^{D} x_i \sin\left(\sqrt{|x_i|}\right), \quad -500 \le x_i \le 500, \quad i = 1, 2, \cdots, 30$$
$$f_{min} = f(420.9687, \cdots, 420.9687) = -12569.5$$

or

$$f(x) = 418.98288727243369D - \sum_{i=1}^{D} x_i \sin\left(\sqrt{|x_i|}\right), \quad -500 \le x_i \le 500, \quad i = 1, 2, \cdots, 30$$
$$f_{min} = f(420.9687, \cdots, 420.9687) = 0$$

8-10. Michalewicz function family (F25 to F27, MS)

$$f(x) = -\sum_{i=1}^{D} \sin(x_i)\left(\sin\left(ix_i^2/\pi\right)\right)^{2m}, \quad 0 \le x_i \le \pi, \quad i = 1, 2, \cdots, D, m = 10$$
$$f_{min} = f(2.20, 1.57) = -1.8013, D = 2$$
$$f_{min} = f\begin{pmatrix} 2.693170, 0.258897, \\ 2.074365, 1.022922, 1.720470 \end{pmatrix} = -4.687658, \quad D = 5$$
$$f_{min} = f\begin{pmatrix} 2.693170, 0.258897, \\ 2.074365, 1.022922, \\ 2.275369, 0.500115, \\ 2.137603, 0.793609, \\ 2.818757, 1.570796 \end{pmatrix} = -9.660152, \quad D = 10$$

D. MN TEST FUNCTIONS (F28-F56)

1. Shekel's Foxholes function (F28, MN)

$$f(x) = \left[\frac{1}{500} + \sum_{j=1}^{25} \frac{1}{\sum_{i=1}^{2}\left(x_i - a_{ij}\right)^6}\right], \quad -65.536 \le x_i \le 65.536, \quad i = 1, 2$$
$$f_{min} = f(-32, \cdots, -32) = 0.998004$$

The parameters used in Shekel's Foxholes function can be seen in [64].

2. Schaffer function (F29, MN)

$$f(x) = 0.5 + \frac{\sin^2\left(\sqrt{x_1^2 + x_2^2}\right) - 0.5}{\left[1 + 0.001\left(x_1^2 + x_2^2\right)\right]^2}, \quad -100 \le x_i \le 100, \quad i = 1, 2$$
$$f_{min} = f(0, 0) = 0$$

3. Six Hump Camel Back function (F30, MN)

$$f(x) = 4x_1^2 - 2.1x_1^4 + \frac{1}{3}x_1^6 + x_1x_2 - 4x_2^2 + 4x_2^4,$$
$$-5 \le x_i \le 5, i = 1, 2$$
$$f_{min} = f(x^*) = -1.0316285,$$
$$x^* = (0.08983, -0.7126), (-0.08983, 0.7126)$$

4. Bohachevsky2 function (F31, MN)

$$f(x) = \sum_{i=1}^{D-1} x_i^2 + 2x_{i+1}^2 - 0.3\cos(3\pi x_i)\cos(4\pi x_{i+1}) + 0.3,$$
$$-100 \le x_i \le 100, \quad i = 1, 2$$
$$f_{min} = f(0, 0, \cdots, 0) = 0$$

5. Bohachevsky3 function (F32, MN)

$$f(x) = x_1^2 + 2x_2^2 - 0.3\cos(3\pi x_1 + 4\pi x_2) + 0.3,$$
$$-100 \le x_i \le 100, i = 1, 2$$
$$f_{min} = f(0, 0) = 0$$

6. Shubert function (F33, MN)

$$f(x) = \left(\sum_{i=1}^{5} i\cos((i+1)x_1 + i)\right) \times \left(\sum_{i=1}^{5} i\cos((i+1)x_2 + i)\right),$$
$$-10 \le x_i \le 10, i = 1, 2$$
$$f_{min} = f(x^*) = -186.7309$$

7. GoldStein-Price function (F34, MN)

$$f(x) = \left[1 + (x_1 + x_2 + 1)^2 \times \left(19 - 14x_1 + 3x_1^2 - 14x_2 + 6x_1x_2\right)\right] \times \left[30 + (2x_1 - 3x_2)^2 \times \left(18 - 32x_1 + 12x_1^2 + 48x_2 - 36x_1x_2 + 27x_2^2\right)\right],$$
$$-2 \le x_i \le 2, \quad i = 1, 2$$
$$f_{min} = f(0, -1) = 3$$

8. Kowalik function (F35, MN)

$$f(x) = \sum_{i=1}^{11}\left[a_i - \frac{x_1(b_i^2 + b_1x_2)}{b_i^2 + b_1x_3 + x_4}\right]^2,$$
$$-5 \le x_i \le 5, \quad i = 1, 2, 3, 4$$
$$f_{min} = f(0.192833, 0.190836, 0.123117, 0.135766) = 0.0003074861$$

The parameters used in Kowalik function can be seen in [64].

9. Shekel (Shekel 5,4, Shekel 7,4, Shekel 10,4) function (F36 to F38, MN)

$$f(x) = -\sum_{i=1}^{m}\left[\sum_{j=1}^{4}(x_j - a_{ji})^2 + c_i\right]^{-1},$$
$$0 \le x_j \le 10, \quad j = 1, 2, 3, 4, \quad m = 5, 7, 10$$
$$f_{min} = f(4.00004, 4.00013, 4.00004, 4.00013) = -10.1532, \quad m = 5$$
$$f_{min} = f(4.00057, 4.00069, 3.99949, 3.99961) = -10.40294, \quad m = 7$$
$$f_{min} = f(4.00075, 4.00059, 3.99966, 3.99951) = -10.53641, \quad m = 10$$

The parameters a and c used in Kowalik function can be seen in [64].

10. Perm function (F39, MN)

$$f(x) = \sum_{k=1}^{D}\left\{\sum_{i=1}^{D}\left(i^k + 0.5\right)\left[\left(\frac{x_i}{i}\right)^i - 1\right]\right\}^2,$$
$$-D \le x_i \le D, \quad i = 1, 2, 3, 4$$
$$f_{min} = f(1, 2, 3, 4) = 0$$

11. Power Sum function (F40, MN)

$$f(x) = \sum_{k=1}^{D}\left[\sum_{i=1}^{D} x_i^k - b_k\right],$$
$$0 \le x_i \le D, \quad D = 1, 2, 3, 4,$$
$$b = [8, 18, 44, 114]$$
$$f_{min} = f(1, 2, 3, 4) = 0$$

12-13. Hartman function family (F41 and F42, MN)

$$f(x) = -\sum_{i=1}^{4} c_i \exp\left[-\sum_{j=1}^{D} a_{ij}(x_j - p_{ij})^2\right], \quad 0 \le x_j \le 1$$

For Hartman 3,4 function,

$$f_{min} = f(0.114614, 0.555649, 0.852547) = -3.86278$$

For Hartman 6,4 function,

$$f_{min} = f\begin{pmatrix}0.20169, 0.150011, 0.476874, \\ 0.275332, 0.311652, 0.6573\end{pmatrix} = -3.32237$$

The coefficients used in Hartman 3,4 function and Hartman 6,4 function can be seen in [64].

14. Griewank function (F43, MN)

$$f(x) = \frac{1}{4000}\sum_{i=1}^{D} x_i^2 - \prod_{i=1}^{D}\cos\left(\frac{x_i}{\sqrt{i}}\right) + 1,$$
$$-600 \le x_i \le 600, \quad i = 1, 2, \cdots, 30$$
$$f_{min} = f(0, 0, \cdots, 0) = 0$$

15. Ackley function (F44, MN)

$$f(x) = -20\exp\left(-0.2\sqrt{\frac{1}{D}\sum_{i=1}^{D} x_i^2}\right)$$

$$- \exp\left(\frac{1}{D}\sum_{i=1}^{D}\cos(2\pi x_i)\right) + 20 + e,$$
$$-32 \le x_i \le 32, \quad i = 1, 2, \cdots, 30$$
$$f_{min} = f(0, 0, \cdots, 0) = 0$$

16. Penalised Levy and Montalvo 1 function (F45, MN)

$$f(x) = \frac{\pi}{D}\left\{10\sin^2(\pi y_1) + \sum_{i=1}^{D-1}(y_i - 1)^2 \times \left[1 + 10\sin^2(\pi y_{i+1})\right] + (y_D - 1)^2\right\} + \sum_{i=1}^{D} u(x_i, 10, 100, 4),$$
$$-50 \le x_i \le 50, \quad i = 1, 2, \cdots, 30$$

where

$$y_i = 1 + 0.25(x_i + 1) \text{ and}$$
$$u(x_i, a, k, m) = \begin{cases} k(x_i - a)^m, & x_i > a \\ 0, & -a \le x_i \le a \\ k(-x_i - a)^m, & x_i < -a \end{cases}$$
$$f_{min} = f(0, 0, \cdots, 0) = 0$$

17. Penalised Levy and Montalvo 2 function (F46, MN)

$$f(x) = 0.1\left\{\begin{array}{l} \sin^2(3\pi x_1) + \sum_{i=1}^{D-1}(x_i - 1)^2 \\ \left[1 + \sin^2(3\pi x_{i+1})\right] \\ + (x_D - 1)^2\left[1 + \sin^2(2\pi x_D)\right] \end{array}\right\} + \sum_{i=1}^{D} u(x_i, 5, 100, 4),$$
$$-50 \le x_i \le 50, \quad i = 1, 2, \cdots, 30$$

where

$$u(x_i, a, k, m) = \begin{cases} k(x_i - a)^m, & x_i > a \\ 0, & -a \le x_i \le a \\ k(-x_i - a)^m, & x_i < -a \end{cases}$$
$$f_{min} = f(1, 1, \cdots, 1) = 0$$

18. Weierstrass function (F47, MN)

$$f(x) = \sum_{i=1}^{D}\left\{\sum_{k=0}^{k_{max}}\left[a^k \cos\left(2\pi b^k(x_i + 0.5)\right)\right]\right\} - D \cdot \sum_{k=0}^{k_{max}}\left[a^k \cos\left(\pi b^k\right)\right],$$
$$a = 0.5, \quad b = 3, \quad k_{max} = 20,$$
$$-0.5 \le x_i \le 0.5, \quad i = 1, 2, \cdots, 30$$
$$f_{min} = f(0, 0, \cdots, 0) = 0$$

19. Modified Langerman function (F48 to F50, MN)

$$f(x) = -\sum_{i=1}^{m} c_i \exp\left[-\frac{1}{\pi}\sum_{j-1}^{D}(x_j - a_{ij})^2\right] \times \cos\left[\pi\sum_{j-1}^{D}(x_j - a_{ij})^2\right],$$
$$0 \le x_i \le 10, \quad i = 1, 2, \cdots, D,$$
$$m = n, \quad n = 2, 5, 10$$
$$f_{min} = f(9.6810707, 0.6666515) = -1.0809, \quad D = 2$$
$$f_{min} = f(8.074, 8.777, 3.467, 1.867, 6.708) = -1.5, \quad D = 5$$
$$f_{min} = f\begin{pmatrix} 8.074, 8.777, 3.467, 1.867, 6.708, \\ 6.349, 4.534, 0.276, 7.633, 1.567 \end{pmatrix} = -0.965, \quad D = 10$$

20. Modified Shekel's Foxholes function (F51 to F53, MN)

$$f(x) = -\sum_{j=1}^{30}\left[c_j + \sum_{i=1}^{D}(x_i - a_{ji})^2\right]^{-1},$$
$$0 \le x_i \le 10, \quad i = 1, 2, \cdots, D$$
$$f_{min} = f(8.024, 9.147) = -12.1191, \quad D = 2$$
$$f_{min} = f(8.025, 9.152, 5.114, 7.621, 4.564) = -10.4056, \quad D = 5$$
$$f_{min} = f\begin{pmatrix} 8.025, 9.152, 5.114, 7.621, 4.564, \\ 4.711, 2.996, 6.126, 0.734, 4.982 \end{pmatrix} = -10.2088, \quad D = 10$$

The constants used in Modified Shekel's Foxholes function can be seen in [64].

21. Fletcher-Powell function (F54 to F56, MN)

$$f(x) = \sum_{i=1}^{D}(A_i - B_i)^2, -\pi \le x_i \le \pi,$$
$$i = 1, 2, \cdots, D, \quad D = 2, 5, 10$$
$$A_i = \sum_{j=1}^{D}\left(a_{ij}\sin\alpha_j + b_{ij}\cos\alpha_j\right),$$
$$B_i = \sum_{j=1}^{D}\left(a_{ij}\sin x_j + b_{ij}\cos x_j\right)$$
$$f\left(\boldsymbol{x}^*\right) = 0, \quad \boldsymbol{x}^* = \boldsymbol{\alpha}$$

The constants used in Fletcher-Powell function can be seen in [64].

ACKNOWLEDGMENT

Sincere gratitude is extended to the editor and the anonymous reviewers for their valuable comments, which greatly improved the presentation of the paper.

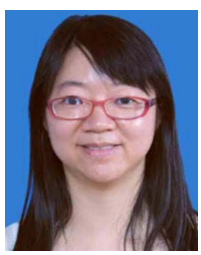
REFERENCES

[1] X. Li, K. Tang, P. N. Suganthan, and Z. Yang, ''Editorial for the special issue of Information Sciences Journal (ISJ) on 'Nature-inspired algorithms for large scale global optimization,''' *Inf. Sci.*, vol. 316, pp. 437–439, Sep. 2015.

[2] M. Lozano, D. Molina, and F. Herrera, "Editorial scalability of evolutionary algorithms and other metaheuristics for large-scale continuous optimization problems," *Soft Comput.*, vol. 15, no. 11, pp. 2085–2087, Nov. 2011.
[3] A. R. Yildiz, "A novel hybrid whale–Nelder–Mead algorithm for optimization of design and manufacturing problems," *Int. J. Adv. Manuf. Technol.*, vol. 105, no. 12, pp. 5091–5104, Nov. 2019.
[4] A. R. Yildiz, H. Abderazek, and S. Mirjalili, "A comparative study of recent non-traditional methods for mechanical design optimization," *Arch. Comput. Methods Eng.*, vol. 27, no. 4, pp. 1031–1048, Sep. 2020.
[5] P. M. Vasant, "Meta-heuristics optimization algorithms in engineering, business, economics, and finance," IGI Global., Philadelphia, PA, USA, Tech. Rep., Sep. 2012.
[6] M. Jain, V. Singh, and A. Rani, "A novel nature-inspired algorithm for optimization: Squirrel search algorithm," *Swarm Evol. Comput.*, vol. 44, pp. 148–175, Feb. 2019.
[7] S. Mirjalili and A. Lewis, "The whale optimization algorithm," *Adv. Eng. Softw.*, vol. 95, pp. 51–67, May 2016.
[8] S. Akyol and B. Alatas, "Plant intelligence based metaheuristic optimization algorithms," *Artif. Intell. Rev.*, vol. 47, no. 4, pp. 417–462, 2017.
[9] R. Moghdani and K. Salimifard, "Volleyball premier league algorithm," *Appl. Soft. Comput.*, vol. 64, pp. 161–185, Mar. 2018.
[10] A. Sajwan and A. Yadav, "AEFA: Artificial electric field algorithm for global optimization," *Swarm Evol. Comput.*, vol. 48, pp. 93–108, Aug. 2019.
[11] B. Alatas and H. Bingol, "A physics-based novel approach for travelling tournament problem: Optics inspired optimization," *Inf. Technol. Control*, vol. 48, no. 3, pp. 373–388, Sep. 2019.
[12] B. Alatas and H. Bingol, "Comparative assessment of light-based intelligent search and optimization algorithms," *Light Eng.*, vol. 28, no. 6, pp. 51–59, 2020.
[13] H. Bingol and B. Alatas, "Chaos based optics inspired optimization algorithms as global solution search approach," *Chaos, Solitons Fractals*, vol. 141, Dec. 2020, Art. no. 110434.
[14] J. Liu, X. Liu, Y. Wu, Z. Yang, and J. Xu, "Dynamic multi-swarm differential learning Harris hawks optimizer and its application to optimal dispatch problem of cascade hydropower stations," *Knowl.-Based Syst.*, vol. 242, Apr. 2022, Art. no. 108281.
[15] N. Baba, T. Shoman, and Y. Sawaragi, "A modified convergence theorem for a random optimization method," *Inf. Sci.*, vol. 13, no. 2, pp. 159–166, Jan. 1977.
[16] F. J. Solis and R. J.-B. Wets, "Minimization by random search techniques," *Math. Oper. Res.*, vol. 6, no. 1, pp. 19–30, Feb. 1981.
[17] E. K. Burke, G. Kendall, and E. Soubeiga, "A tabu-search hyperheuristic for timetabling and rostering," *J. Heuristics*, vol. 9, no. 6, pp. 451–470, 2003.
[18] M. Lozano, F. Herrera, N. Krasnogor, and D. Molina, "Real-coded memetic algorithms with crossover hill-climbing," *Evol. Comput.*, vol. 12, no. 3, pp. 273–302, Sep. 2004.
[19] C. Voudouris and E. P. K. Tsang, "Guided local search," in *Handbook Metaheuristics*. Aug. 2010, pp. 185–218.
[20] I. Rechenberg, "Cybernetic solution path of an experiment problem," Library Transl., Roy. Aircr. Establishment, Farnborough, Hants, U.K., Tech. Rep., 1965, vol. 1122.
[21] H. P. Schwefel, "Kybernetische evolution als strategie der experimentellen forschung in der strömungstechnik," M.S. thesis, Tech. Univ. Berlin, Germany, Jan. 1965.
[22] L. J. Fogel, A. J. Owens, and M. J. Walsh, *Artificial Intelligence Through Simulated Evolution*. New York, NY, USA: Wiley, Oct. 1966.
[23] J. H. Holland, *Adaptation in Natural and Artificial Systems*. Ann Arbor, MI, USA: University of Michigan Press, Apr. 1975.
[24] J. R. Koza, "Genetic programming: A paradigm for genetically breeding populations of computer programs to solve problems," Stanford Univ. Comput. Sci. Dept., Stanford, CA, USA, Tech. Rep. STAN-CS-90-1314, Jun. 1990.
[25] P. J. Angeline, G. M. Saunders, and J. B. Pollack, "An evolutionary algorithm that constructs recurrent neural networks," *IEEE Trans. Neural Netw.*, vol. 5, no. 1, pp. 54–65, Jan. 1994.
[26] R. Storn and K. Price, "Differential evolution—A simple and efficient adaptive scheme for global optimization over continuous spaces," Int. Comput. Sci. Inst., Berkley, CA, USA, Tech. Rep., Jan. 1995.
[27] R. Storn and K. Price, "Differential evolution—A simple and efficient heuristic for global optimization over continuous spaces," *J. Global Optim.*, vol. 11, no. 4, pp. 341–359, 1997.
[28] M. Dorigo, "Optimization, learning and natural algorithm," Ph.D. thesis, Dipartimento di Elettronica, Politecnico di Milano, Milano, Italy, Jan. 1992.
[29] R. C. Eberhart and J. Kennedy, "A new optimizer using particle swarm theory," in *Proc. 6th Int. Symp. Micro Mach. Hum. Sci.*, vol. 1, 1995, pp. 39–43.
[30] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Proc. IEEE Int. Conf. Neural Netw.*, vol. 4, Aug. 2002, pp. 1942–1948.
[31] D. Karaboga, "An idea based on honey bee swarm for numerical optimization," Erciyes Univ., Eng. Fac., Comput. Eng. Dept., Tech. Rep., TR06, Jan. 2005.
[32] X.-S. Yang and S. Deb, "Cuckoo search via Lévy flights," in *Proc. World Congr. Nature Biologically Inspired Comput. (NaBIC)*, Dec. 2009, pp. 210–214.
[33] X.-S. Yang, "A new metaheuristic bat-inspired algorithm," in *Nature Inspired Cooperative Strategies for Optimization* (Studies in Computational Intelligence), vol. 284. Berlin, Germany: Springer, Apr. 2010, pp. 65–74.
[34] S. Mirjalili, S. M. Mirjalili, and A. Lewis, "Grey wolf optimizer," *Adv. Eng. Softw.*, vol. 69, pp. 46–61, Mar. 2014.
[35] S. Liu, Y. Yang, and Y. Zhou, "A swarm intelligence algorithm-lion swarm optimization," *Pattern Recognit. Artif. Intell.*, vol. 3, no. 5, pp. 431–441, Jun. 2018.
[36] A. A. Heidari, S. Mirjalili, H. Faris, I. Aljarah, M. Mafarja, and H. Chen, "Harris hawks optimization: Algorithm and applications," *Future Gener. Comput. Syst.*, vol. 97, pp. 849–872, Aug. 2019.
[37] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi, "Optimization by simulated annealing," *Science*, vol. 220, no. 4598, pp. 671–680, 1983.
[38] V. Černý, "Thermodynamical approach to the traveling salesman problem: An efficient simulation algorithm," *J. Optim. Theory Appl.*, vol. 45, no. 1, pp. 41–51, Jan. 1985.
[39] O. K. Erol and I. Eksin, "A new optimization method: Big-bang-big crunch," *Adv. Eng. Softw.*, vol. 37, no. 2, pp. 106–111, Feb. 2006.
[40] E. Rashedi, H. Nezamabadi-Pour, and S. Saryazdi, "GSA: A gravitational search algorithm," *J. Inf. Sci.*, vol. 179, no. 13, pp. 2232–2248, 2009.
[41] W. Zhao, L. Wang, and Z. Zhang, "Atom search optimization and its application to solve a hydrogeologic parameter estimation problem," *Knowl.-Based Syst.*, vol. 163, pp. 283–304, Jan. 2019.
[42] W. Zhao, L. Wang, and Z. Zhang, "A novel atom search optimization for dispersion coefficient estimation in groundwater," *Future Gener. Comput. Syst.*, vol. 91, pp. 601–610, Feb. 2019.
[43] Z. W. Geem, J. H. Kim, and G. V. Loganathan, "A new heuristic optimization algorithm: Harmony search," *J. Simul.*, vol. 76, no. 2, pp. 60–68, Feb. 2001.
[44] E. Atashpaz-Gargari and C. Lucas, "Imperialist competitive algorithm: An algorithm for optimization inspired by imperialistic competition," in *Proc. IEEE Congr. Evol. Comput.*, Sep. 2007, pp. 4661–4667.
[45] A. H. Kashan, "League championship algorithm: A new algorithm for numerical function optimization," in *Proc. Int. Conf. Soft Comput. Pattern Recognit.*, 2009, pp. 43–48.
[46] R. V. Rao, V. J. Savsani, and D. P. Vakharia, "Teaching-learning-based optimization: A novel method for constrained mechanical design optimization problems," *Comput.-Aided Des.*, vol. 43, no. 3, pp. 303–315, Mar. 2011.
[47] R. V. Rao, V. J. Savsani, and D. P. Vakharia, "Teaching–learning-based optimization: An optimization method for continuous non-linear large scale problems," *Inf. Sci.*, vol. 183, no. 1, pp. 1–15, Jan. 2012.
[48] A. R. Mehrabian and C. Lucas, "A novel numerical optimization algorithm inspired from weed colonization," *Ecol. Inform.*, vol. 1, no. 4, pp. 355–366, Dec. 2006.
[49] D. Simon, "Biogeography-based optimization," *IEEE Trans. Evol. Comput.*, vol. 12, no. 6, pp. 702–713, Dec. 2008.
[50] H. Chen and Y. Zhu, "Optimization based on symbiotic multi-species coevolution," *Appl. Math. Comput.*, vol. 205, no. 1, pp. 47–60, Nov. 2008.
[51] I. Fister, Jr., X.-S. Yang, I. Fister, J. Brest, and D. Fister, "A brief review of nature-inspired algorithms for optimization," *Electrotech. Rev.*, vol. 80, no. 3, pp. 1–7, 2013.
[52] K. Hussain, M. N. M. Salleh, S. Cheng, and Y. Shi, "Metaheuristic research: A comprehensive survey," *Artif. Intell. Rev.*, vol. 52, no. 4, pp. 2191–2233, Dec. 2019.
[53] A. Faramarzi, M. Heidarinejad, S. Mirjalili, and A. H. Gandomi, "Marine predators algorithm: A nature-inspired metaheuristic," *Exp. Syst. Appl.*, vol. 152, Aug. 2020, Art. no. 113377.

[54] S. Li, H. Chen, M. Wang, A. A. Heidari, and S. Mirjalili, "Slime mould algorithm: A new method for stochastic optimization," *Future Gener. Comput. Syst.*, vol. 111, pp. 300–323, Oct. 2020.
[55] A. Mohammadi-Balani, M. D. Nayeri, A. Azar, and M. Taghizadeh-Yazdi, "Golden eagle optimizer: A nature-inspired metaheuristic algorithm," *Comput. Ind. Eng.*, vol. 152, Feb. 2021, Art. no. 107050.
[56] W. Zhao, L. Wang, and S. Mirjalili, "Artificial hummingbird algorithm: A new bio-inspired optimizer with its engineering applications," *Comput. Methods Appl. Mech. Eng.*, vol. 388, Jan. 2022, Art. no. 114194.
[57] D. H. Wolper and W. G. Macready, "No free lunch theorems for optimization," *IEEE Trans. Evol. Comput.*, vol. 1, no. 1, pp. 67–82, Apr. 1997.
[58] Z. Yang and C. Wei, "Prediction of equipment performance index based on improved chaotic lion swarm optimization–LSTM," *Soft Comput.*, vol. 24, no. 13, pp. 9441–9465, Jul. 2020.
[59] Z. Daoqing and J. Mingyan, "Parallel discrete lion swarm optimization algorithm for solving traveling salesman problem," *J. Syst. Eng. Electron.*, vol. 31, no. 4, pp. 751–760, Aug. 2020.
[60] W. Qiao, H. Lu, G. Zhou, M. Azimi, Q. Yang, and W. Tian, "A hybrid algorithm for carbon dioxide emissions forecasting based on improved lion swarm optimizer," *J. Cleaner Prod.*, vol. 244, Jan. 2020, Art. no. 118612.
[61] Y. Guo and M. Y. Jiang, "Job-shop scheduling problem with improved lion swarm optimization," in *Proc. Int. Conf. Natural Comput., Fuzzy Syst. Knowl. Discovery, ICNC-FSKD, Adv. Natural Comput., Fuzzy Syst. Knowl. Discovery*, vol. 1348, Jun. 2021, pp. 661–669.
[62] W. J. Niu, Z. K. Feng, and C. T. Cheng, "Multi-objective quantum-behaved particle swarm optimization for operation of cascade hydropower stations," *J. Hydroelectric Eng.*, vol. 36, no. 5, pp. 47–57, Jul. 2017.
[63] J. L. Song, Y. L. Tan, and Z. T. Xiong, "Multilevel parallel quantum-behaved particle swarm optimization based on asymmetrical boundary variation," *Appl. Res. Comput.*, vol. 30, no. 6, pp. 1630–1632, Mar. 2013.
[64] J. Liu, D. Li, Y. Wu, and D. Liu, "Lion swarm optimization algorithm for comparative study with application to optimal dispatch of cascade hydropower stations," *Appl. Soft Comput.*, vol. 87, Feb. 2020, Art. no. 105974.
[65] H. R. Tizhoosh, "Opposition-based learning: A new scheme for machine intelligence," in *Proc. Int. Conf. Comput. Intell. Modelling, Control Automat., Int. Conf. Intell. Agents, Web Technol. Internet Commerce*, vol. 1, Nov. 2005, pp. 695–701.
[66] X. Y. Zhou, Z. J. Wu, H. Wang, K. S. Li, and H. Y. Zhang, "Elite opposition-based particle swarm optimization," *Acta Electronica Sinica*, vol. 41, no. 8, pp. 1647–1652, Sep. 2013.
[67] Y. Feng, J. Liu, and Y. He, "Chaos-based dynamic population firefly algorithm," *J. Comput. Appl.*, vol. 33, no. 3, pp. 796–799, Sep. 2013.
[68] X. Q. Zhang and L. X. Qin, "Glowworm swarm optimization based on hybrid mutation," *Comput. Appl. Software.*, vol. 33, no. 2, pp. 272–275, Feb. 2016.
[69] H. Bingol and B. Alatas, "Chaotic league championship algorithms," *Arabian J. Sci. Eng.*, vol. 41, no. 12, pp. 5123–5147, May 2016.
[70] E. Varol Altay and B. Alatas, "Bird swarm algorithms with chaotic mapping," *Artif. Intell. Rev.*, vol. 53, no. 2, pp. 1373–1414, Feb. 2020.
[71] X. Lin, B. Feng, and J. Sun, "Quantum-behaved particle swarm optimization algorithm based on bounded mutation," *Comput. Eng.*, vol. 34, no. 12, pp. 187–188, Jun. 2008.
[72] M. Helbig and A. P. Engelbrecht, "Dynamic multi-objective optimization using PSO," in *Metaheuristics for Dynamic Optimization*. Berlin, Germany: Springer, vol. 433, 2013, pp. 147–188.
[73] W. Gao, "Fast immunized evolutionary programming," in *Proc. 6th Annu. Conf. Evol. Program.*, Jul. 1996, pp. 460–461.
[74] D. Karaboga and B. Akay, "A comparative study of artificial bee colony algorithm," *Appl. Math. Comput.*, vol. 214, no. 1, pp. 108–132, 2009.
[75] K. Price, R. M. Storn, and J. A. Lampinen, *Differential Evolution: A Practical Approach to Global Optimization*. New York, NY, USA: Springer, Jan. 2005.
[76] S. Rahnamayan, H. R. Tizhoosh, and M. M. A. Salama, "Opposition-based differential evolution," *IEEE Trans. Evol. Comput.*, vol. 12, no. 1, pp. 64–79, Feb. 2008.
[77] M. Ali, P. Siarry, and M. Pant, "An efficient differential evolution based algorithm for solving multi-objective optimization problems," *Eur. J. Oper. Res.*, vol. 217, no. 2, pp. 404–416, Mar. 2012.
[78] N. Zhang, C. Li, R. Li, X. Lai, and Y. Zhang, "A mixed-strategy based gravitational search algorithm for parameter identification of hydraulic turbine governing system," *Knowl.-Based Syst.*, vol. 109, pp. 218–237, Oct. 2016.
[79] J. J. Liang, B. Y. Qu, and P. N. Suganthan, "Problem definitions and evaluation criteria for the CEC2014 special session and competition on single objective real-parameter numerical optimization," Comput. Intell. Lab., Zhengzhou Univ., Zhengzhou China Tech. Rep., Nanyang Technol. Univ., Singapore, Tech. Rep., 11, Dec. 2013.
[80] D. Chen, F. Zou, R. Lu, and P. Wang, "Learning backtracking search optimisation algorithm and its application," *Inf. Sci.*, vol. 376, pp. 71–94, Jan. 2017.
[81] H. Nenavath, D. R. K. Jatoth, and D. S. Das, "A synergy of the sine-cosine algorithm and particle swarm optimizer for improved global optimization and object tracking," *Swarm Evol. Comput.*, vol. 43, pp. 1–30, Dec. 2018.
[82] T. Peram, K. Veeramachaneni, and C. K. Mohan, "Fitness-distance-ratio based particle swarm optimization," in *Proc. IEEE Swarm Intell. Symp.*, 2003, pp. 174–181.
[83] J. J. Liang, A. K. Qin, P. N. Suganthan, and S. Baskar, "Comprehensive learning particle swarm optimizer for global optimization of multimodal functions," *IEEE Trans. Evol. Comput.*, vol. 10, no. 3, pp. 281–295, Jun. 2006.
[84] R. Mendes, J. Kennedy, and J. Neves, "The fully informed particle swarm: Simpler, maybe better," *IEEE Trans. Evol. Comput.*, vol. 8, no. 3, pp. 204–210, Jun. 2004.
[85] F. Van Den Bergh and A. P. Engelbrecht, "A cooperative approach to particle swarm optimization," *IEEE Trans. Evol. Comput.*, vol. 8, no. 3, pp. 225–239, Jun. 2004.
[86] 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 Trans. Evol. Comput.*, vol. 10, no. 6, pp. 646–657, Jul. 2006.
[87] A. K. Qin, V. L. Huang, and P. N. Suganthan, "Differential evolution algorithm with strategy adaptation for global numerical optimization," *IEEE Trans. Evol. Comput.*, vol. 13, no. 2, pp. 398–417, Sep. 2009.
[88] R. V. Rao and V. Patel, "An elitist teaching-learning-based optimization algorithm for solving complex constrained optimization problems," *Int. J. Ind. Eng. Comput.*, vol. 3, no. 4, pp. 535–560, Jul. 2012.
[89] Z. Feng, W. Lei, X. Hei, D. Chen, and D. Yang, "Teaching–learning-based optimization with dynamic group strategy for global optimization," *Inf. Sci.*, vol. 273, no. 273, pp. 112–131, 2014.
[90] P. Civicioglu, "Backtracking search optimization algorithm for numerical optimization problems," *Appl. Math. Comput.*, vol. 219, no. 15, pp. 8121–8144, Apr. 2013.
[91] M. J. Reddy and D. N. Kumar, "Multiobjective differential evolution with application to reservoir system optimization," *J. Comput. Civil Eng.*, vol. 21, no. 2, pp. 136–146, Mar. 2007.
[92] B. Ming, J.-X. Chang, Q. Huang, Y.-M. Wang, and S.-Z. Huang, "Optimal operation of multi-reservoir system based-on cuckoo search algorithm," *Water Resour. Manage.*, vol. 29, no. 15, pp. 5671–5687, 2015.
[93] X. Li, J. Wei, T. Li, G. Wang, and W. W.-G. Yeh, "A parallel dynamic programming algorithm for multi-reservoir system optimization," *Adv. Water Resour.*, vol. 67, pp. 1–15, May 2014.

JUNFENG LIU received the B.S. degree from Xinyang Normal University, Xinyang, China, in 2012, the M.S. degree from Wenzhou University, Wenzhou, China, in 2016, and the Ph.D. degree from Wuhan University, Wuhan, China. He is currently pursuing the Postdoctoral degree with Zhejiang University, Hangzhou, China. His research interests include computational intelligence and evolutionary algorithms.



YUN WU received the B.S. degree from Central China Normal University, Wuhan, China, in 2004, the M.S. degree from Capital Normal University, Beijing, China, in 2010, and the Ph.D. degree in mathematics from Wuhan University, Wuhan. Since 2004, she has been a Lecturer with the School of Science, Jiujiang University, China. Her research interests include computational intelligence and evolutionary algorithms.