Soft Computing (2022) 26:2075–2097
https://doi.org/10.1007/s00500-021-06623-2

FOUNDATIONS

A hybrid whale optimization algorithm with artificial bee colony

Chenjun Tang[1,2] · Wei Sun[1,2] · Min Xue[1,2] · Xing Zhang[1,2] · Hongwei Tang[1,2] · Wei Wu[1,2]

Accepted: 27 November 2021 / Published online: 28 January 2022

Abstract
In this paper, the defects and deficiencies of the recently proposed whale optimization algorithm (WOA) are improved. A whale optimization algorithm mixed with an artificial bee colony (ACWOA) is proposed to solve the WOA problems of slow convergence, low precision, and easy to fall into local optimum. The ACWOA algorithm integrates the artificial bee colony algorithm and chaotic mapping, effectively avoiding the local optimal situation and improving the quality of the initial solution. Also, nonlinear convergence factors and adaptive inertia weight coefficients are added to accelerate the convergence rate. To verify the performance of the improved algorithm, 20 benchmark functions and CEC2019 multi-modal multi-objective benchmark functions have been used to compare ACWOA with the classical intelligent population algorithms (PSO, MVO, and GWO) and the recent state-of-the-art algorithms (CWOA, HWPSO, and HIWOA) in recent years. The proposed algorithm is applied to two well-known engineering mathematical models and a real application (the quality process control). The experiments show that the ACWOA algorithm has strong competitiveness in convergence speed and solution accuracy and has certain practical value in complex mathematical model scenarios.

Keywords Whale optimization algorithm · Artificial bee colony algorithm · Nonlinear convergence factor · Adaptive inertia weight

1 Introduction

In the fields such as machine learning, image processing, and neural networks, there are large-scale complex global optimization problems. Such problems have characteristics, for example, the variables are closely related to each other and the dimensions are high, so it is very difficult to obtain good results using traditional optimization algorithms in this complicated environment (Luo and Shi 2018). To solve the large-scale complex high-dimensional optimization problems, new and various optimization algorithms are constantly appearing. Population intelligence algorithm is one of the most popular research fields. This is an algorithm that is generated by simulating the way life animals live in nature (Kennedy and Eberhart 2002; Rajabioun 2011; Rashedi et al. 2009; Failed 2021d). Due to the complex and ever-changing problems of predators in nature (like predation of prey, handling of food, searching for paths, etc.), the cooperation and interconnection between individuals have formed a powerful group force to solve these problems. For example, Marco Dorigo comes up with the ant colony optimization (ACO) algorithm after discovering the path behavior of ants searching for food

✉ Wei Sun
wei_sun@hnu.edu.cn

Chenjun Tang
tangchenjun00@163.com

Min Xue
xueminxhd@163.com

Xing Zhang
xing_zhangshare@163.com

Hongwei Tang
thwei2008@126.com

Wei Wu
antonie10@163.com

[1] College of Electrical and Information Engineering, Hunan University, Changsha 410082, China

[2] Hunan Provincial Key Laboratory of Intelligent Robot Technology in Electronic Manufacturing, Hunan University, Changsha 410082, China

(Dong et al. Oct 2018). Mirjalili represents gray wolf optimization (GWO) algorithm by observing gray wolfs kill their prey (Mirjalili et al. 2014). Karaboga adopts the artificial swarm optimization (ASO) algorithm based on the bees' behavior of going out to collect honey (Karaboga 2010). Kennedy comes up with the famous particle swarm optimization (PSO) algorithm (Niknam and Amiri 2010). The population intelligence algorithms have brought great practical value to society and have achieved success in many projects (Talbi et al. 2004; He et al. Nov 2015; Yang et al. 2019; Cai et al. Aug 2019; Failed 2021a, b; Mohammadzadeh and Gharehchopogh 2021). However, different algorithms have their advantages and disadvantages in application environments (Singh et al. 2015; Zhang et al. 2019).

The whale optimization algorithm (WOA) is one of the population intelligence algorithms, which originates from the feeding behavior of whales. Because of its advantages of simple implementation and few parameters, it has been successfully applied in various fields (Mirjalili and Lewis May 2016; Mohammadzadeh and Gharehchopogh 2021). In the WOA, each individual represents a solution to the problem. According to the historical optimal value in the population, each individual is assigned a fitness function value and a speed to update the location. After continuous position updates, the optimal solution of the problem is obtained. However, in the process of individual search, the speed of the individual is fast in the early stage, and it is easy to fall into the local optimum. As the number of iterations increases, the diversity of the population will decrease, causing individuals to lose the ability to search. The final result cannot meet our needs in some complex problems.

Artificial bee colony algorithm (ABC) is also a high-quality population intelligent algorithm. It has the extremely high exploratory ability, and the richness of the population is strong. However, it has a slower convergence speed and lower accuracy in the latter part of the iteration.

However, how to balance the exploitation and detection of algorithms is still the focus of research and attention (Gharehchopogh and Gholizadeh 2019; Mehne and Mirjalili Jul 2018; Ding et al. 2018d). To solve the above problem and make up for the shortcomings of the two algorithms (WOA and ABC), we propose a combination of whale optimization algorithm and artificial bee colony algorithm. The main contributions of this paper are the following:

To avoid the local optimum and improve the accuracy, we propose to combine the whale optimization algorithm with the artificial bee colony algorithm to make up for the shortcomings of each algorithm. The whales are subjected to the behavior of bees. The individuals who will carry out the whale movement, the individuals who take the honey movement, and the last moment individual whales make greedy decisions through the fitness function, retaining the individuals with the best fitness values, and the other two are directly eliminated.

2. To improve the sensitivity of population initialization, we propose to use a chaotic mapping model instead of a random average distribution model. The chaos theory can make the initial conditions more sensitive and can produce a more variable number range (Sayed et al. Jul 2018).

3. To further accelerate the convergence speed, we use nonlinear convergence and adaptive inertia weights.

4. We experimented with the proposed algorithm and other advanced optimization algorithms in multiple environments (like 20 benchmark functions, two well-known engineering mathematical models, and a real application).

The rest of this paper is organized as follows: In Sect. 2, we review the related works. In Sect. 3, a detailed introduction to the traditional WOA algorithm, artificial bee colony algorithm, and chaotic mapping is given. Section 4 details the architecture of the ACWOA algorithm. Sections 5 and 6, respectively, describe the comparison between ACWOA and other advanced algorithms on various experiments (20 test benchmark functions, two well-known mathematical models, and a real application). The algorithms are analyzed and discussed from the experimental results. Section 7 summarizes the results of this study and the outlook for the future.

2 Related works

The whale optimization algorithm is a new population intelligence algorithm proposed in 2016. It is constructed by Mirjalili, a scholar at Griffith University in Australia (Mirjalili and Lewis May 2016). This algorithm is inspired by the hunting behavior of the humpback whales (bubble net attack behavior). They prey on prey by fighting with prey, attacking prey, and searching for prey (Sun et al. 2018a). The WOA algorithm simulates the process of predation behaviors to find the optimal solution. Due to its simple principle, convenient implementation, and fewer parameters, the WOA is widely used in various fields (Gharehchopogh and Gholizadeh 2019). Karlekar uses a support vector machine based on whale optimization to medical data classification (Karlekar and Gomathi 2018). The algorithm is applied to the nonlinear Gaussian adaptive PID controller (Khadanga et al. 2018b, c). This algorithm is also used by Aziz in image segmentation (Abd El Aziz et al. 2017), etc.

Although the WOA algorithm has achieved results in many engineering applications, it has early convergence speeds and is easy to fall into local optimum in dealing with some complex or high-dimensional optimization

problems (Gharehchopogh and Gholizadeh 2019; Abdel-Basset et al. Aug 2018; Mohapatra et al. 2017; Khadanga et al. 2018c). In recent years, experts and scholars have proposed many improved measures to compensate for their shortcomings in the population intelligence algorithm. Mafarja improves the performance of the WOA algorithm by adding crossover and mutation (Mafarja and Mirjalili 2018). Mafarja optimizes the optimal solution by mixing the WOA algorithm with the classical annealing algorithm (Mafarja and Mirjalili 2017). Sun, Y. J. uses the Levy flight strategy to WOA algorithm to avoid local optimization and adds a quadratic interpolation to improve local mining capacity (Sun et al. 2018). Yan, Z. H. adopts weight coefficient to the WOA algorithm to improve the convergence rate and accuracy (Yan et al. 2018). Mohammadzadeh et al. propose to add a flower pollination algorithm (FPA) to WOA to improve efficiency (Mohammadzadeh and Gharehchopogh 2021). And these measures have been confirmed in practical engineering applications.

On the other hand, to overcome the shortcomings of the ABC algorithm, there are currently many schemes. For example, Nouria Rahnema et al. propose a combined algorithm of ABC and WOA (ABCWOA). Random memory (RM) and elite memory (EM) are proposed in the ABC algorithm, where RM is used in the search phase of the WOA algorithm and EM is used to improve the convergence of the algorithm (Rahnema and Gharehchopogh 2020). Peng Shao et al. represent the enhancing ABC algorithm using the refraction principle (EABC-RP). In the search phase, to increase population diversity, the unified opposition-based learning (UOBL) based on the refraction principle is applied to the ABC algorithm. For exploitation, the UOBL is used to effectively avoid the local optimal (Shao et al. 2020).

Some measures (such as chaotic mapping, nonlinear convergence, and adaptive inertia weights) can also have a better effect on the population intelligence algorithm. Various chaotic maps are used to the vortex search algorithm (VSA) to obtain better performance by Gharehchopogh (2021). Sayed, G. I. improves the quality of the initial solution of the WOA algorithm using chaotic maps (Sayed et al. 2018). Converting linear convergence to nonlinear convergence and assigning their respective weight values to the populations can make the algorithm get better results (Tang et al. 2019).

The algorithm proposed in this paper is more similar to the ABCWOA algorithm proposed by Nouria Rahnema et al. (Rahnema and Gharehchopogh 2020). The difference compared with ABCWOA is that we use a greedy decision-making method to update the position of individuals in the population. The individuals who will carry out the whale movement, the individuals who take the honey movement, and the last moment individual whales make greedy decisions through the fitness function, retaining the individuals with the best fitness values, and the other two are directly eliminated. At the same time, the threshold is set up. When the position update of the whale population does not change, the whales are converted into "detection whales" to further jump out of the local optimum. On the other hand, chaos mapping, nonlinear convergence, and adaptive weights are added to further optimize the performance of the algorithm.

3 Background

3.1 Whale optimization algorithm

The whale optimization algorithm is a novel optimization algorithm whose mechanism is derived from the social behavior of humpback whales. Adult humpback whales are typically 13–15 m long and feed on small fish and shrimp. Its special predation method is blister-net predation, which includes three activities (enveloping prey, spiraling off prey, and randomly searching for prey). The entire predation behavior process does not know the optimal solution location in advance. Through the interaction and cooperation between individuals, the individual positions are continuously updated, and finally an approximate optimal solution is obtained (Mirjalili and Lewis May 2016).

3.2 The activity of surrounding prey

Average random placement is used to initialize all individual whales and infer which whale is closest to the prey. The movement of other whales is changed by the optimal individual, so that the whales in the non-optimal position are close to the optimal position of the whales. The movement of whale individual position is as follows:

$$\overrightarrow{X}(t+1) = \overrightarrow{X_{\text{best}}}(t) - \overrightarrow{A} \cdot \overrightarrow{D} \quad \text{if} \quad q < 0.5 \tag{1}$$

$$\overrightarrow{A} = \overrightarrow{a} \cdot (2 \cdot \overrightarrow{\text{rand}} - 1) \tag{2}$$

$$\overrightarrow{C} = 2 \cdot \overrightarrow{\text{rand}} \tag{3}$$

$$\overrightarrow{D} = \left| \overrightarrow{C} \cdot \overrightarrow{X_{\text{best}}}(t) - \overrightarrow{X}(t) \right| \tag{4}$$

$$\overrightarrow{a} = 2 \cdot \left(1 - \frac{t}{t_{\max}}\right) \tag{5}$$

where $\vec{X}(t)$ and $\vec{X}(t+1)$ represent the current location of the whale individual and the location of the next moment whale, respectively. $\overrightarrow{X_{\text{best}}}(t)$ is the location of the whale closest to the prey. $\vec{a}$ is used as a linear convergence factor, and its value decreases linearly from 2 to 0 as the number of iterations increases. The number of iterations at this time t and the maximum number of iterations $t_{\max}$ are adopted. $\overrightarrow{\text{rand}}$ is a random number, ranging from 0 to 1.

3.3 The activity of spirally attacking prey

The humpback whale revolves around the prey with its unique blister-like spiraling behavior. The mathematical function model of the individual position is as follows:

$$\overrightarrow{X}(t+1) = \overrightarrow{X_{\text{best}}}(t) + \overrightarrow{D1} \cdot e^{\text{bl}} \cdot \cos(2\pi l) \quad \text{if} \quad q \geq 0.5 \tag{6}$$

$$\overrightarrow{D1} = \left|\overrightarrow{X_{\text{best}}}(t) - \overrightarrow{X}(t)\right| \tag{7}$$

where the logarithmic spiral shape constant b is 1 and then $l \in [-1, 1]$ represents a random number. It is worth noting that q is a random number of [0,1], and its value determines whether the whale population is surrounding prey or spirally ascending prey. Each behavior has a 50% probability.

3.4 The activity of searching for prey

The search for prey behavior is divided into two categories. One is the behavior of the whale population surrounding the prey, which is Formula (1), and the whale population moves closer to $\overrightarrow{X_{\text{best}}}(t)$. The other is to randomly acquire the location of a whale individual and force other whale individuals to make corresponding location updates. The detailed whale running model is as follows:

$$\overrightarrow{X}(t+1) = \overrightarrow{X}_{\text{rand}}(t) - \overrightarrow{A} \cdot \overrightarrow{D2} \tag{8}$$

$$\overrightarrow{D2} = \left|\overrightarrow{C} \cdot \overrightarrow{X}_{\text{rand}}(t) - \overrightarrow{X}(t)\right| \tag{9}$$

where $\overrightarrow{X}_{\text{rand}}(t)$ represents the location of a random whale individual. Other whales are affected by $\overrightarrow{X}_{\text{rand}}(t)$ rather than $\overrightarrow{X_{\text{best}}}(t)$. This behavior will increase the diversity of the population and improve global detection capabilities. The location update of the whale population depends on $|A|$. It is defined that when $|A| \leq 1$, the whale population performs Formula (1), and when $|A| \geq 1$, the whale population performs Formula (6). As the number of iterations increases, $\vec{a}$ will decrease linearly. When $\vec{a}$ is reduced to 1, the whale population will only move to $\overrightarrow{X_{\text{best}}}(t)$.

The pseudo-code of the WOA algorithm is as follows, and the flowchart of the WOA algorithm is shown in Fig. 1.

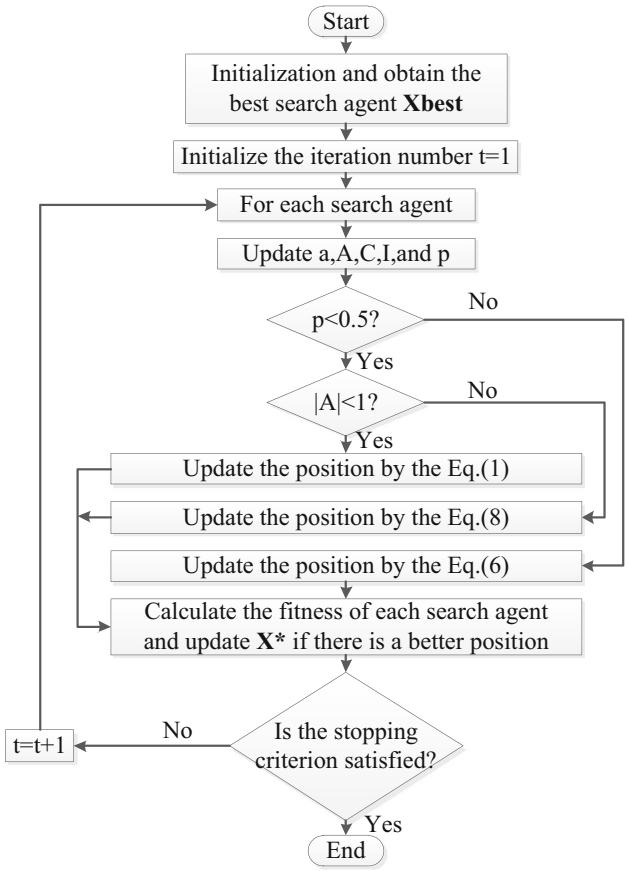

Fig. 1 Flowchart for WOA algorithm

Algorithm 1 The pseudo code of the whale optimization algorithm (WOA)

```
Initialization {
        initializes t_max, b=1, a, A, C, l , q, and Test functions,
        initialize individual whale X_i (i = 1, 2, 3,..., n) ,
        calculate the fitness of each search agent,
            X_best = the best search agent
           }
Main loop {
        while (t < t_max)
            for each search agent
                update a, A, C, l and q
                if1(q < 0.5)
                    if2(|A| < 1)
                        update the position by the Eq. (1)
                    else if2(|A|≥ 1)
                        select a random search agent (X_rand) update the position by the Eq. (8).
                    end if2
                else if1 (q≥ 0.5)
                    update the position by the Eq. (6)
                end if1
            end for
            make sure each search agent is valid
            update X_best if there is a better solution
            t = t + 1
        end while
       }
return X_best
```

3.5 Artificial bee colony algorithm

In 2005, scholar Karaboga proposes a new population intelligence algorithm, an artificial population bee algorithm, by observing the bee colony for honey collecting behavior (Xue et al. 2018). The principle of the algorithm comes from the cooperation of the bees. These bees have different identities and perform different behaviors. They conduct collective movements through the exchange of bee colonies and finally find the location of the largest honey source (the optimal solution to the problem).

The artificial bee colony algorithm divides the bee colony into three identities: employed bees, onlooker bees, and scout bees. Each bee learns the honey source of the current location, which represents a possible solution to the optimization problem. The employed bees occupy 50% of the number of populations, and the rest are onlooker bees. The employed bees look for food near the food source through memory and share food-related information with onlooker bees. The onlooker bees get the best food source through comparison and look for food around the best food source. At this point, some employed bees are converted into scout bees to find new food sources. After the continuous updates, we finally get the corresponding results. The specific implementation process is as follows.

We suppose that the possible solution of the high-dimensional complex problem is $X_i = \{x_{i1}, x_{i2}, ..., x_{i\,\dim}\}$. Firstly, all of the populations are initialized. The specific mathematical expression is as follows:

$$x_{ij} = x^j_{\min} + r_1 \cdot (x^j_{\max} - x^j_{\min}), \quad j = 1, 2, ..., \dim, \quad i = 1, 2, ..., N \tag{10}$$

where dim and i indicate the number of dimensions and the total number of populations, respectively. r_1 is random between [0, 1]. $x^j_{\max}$ and $x^j_{\min}$ are upper boundary and lower boundary, respectively.

Secondly, the employed bees establish a new food source near their respective locations. The specific location update is as follows:

$$x^*_{ij} = x_{ij} + r_2 \cdot (x_{ij} - x_{kj}), \quad j = 1, 2, ..., \dim, \quad i = 1, 2, ..., N, k \neq i \tag{11}$$

where x_{ij} is the j-dimensional position of the ith honey source at the current moment and x_{kj} randomly selects another (non-ith) honey source position. r_2 is a random number between [− 1, 1], and x^*_{ij} represents the position of a new honey source. The original possible solution $X_i = \{x_{i1}, x_{i2}, ..., x_{i\,\dim}\}$ is compared with the new possible solution $X^*_i = \{x_{i1}, x_{i2}, ..., x_{i\,\dim}\}$. If the fitness value of the new possible solution $F(X^*_i)$ is better than the original possible solution $F(X_i)$, the original honey source is replaced by the new honey source, otherwise the original honey source information is retained.

Thirdly, after employed bees searched, the employed bees pass the result to onlooker bees. The onlooker bees will select a honey source to follow by a probability calculation. The specific probability formula is as follows:

$$p_i = \frac{F_i}{\sum_{i=1}^{N} F_i} \tag{12}$$

p_i is the probability of following the ith honey source and F_i represents the fitness value of X_i. According to the probability value p_i, the onlooker bees select the corresponding food source, and these onlooker bees perform a position update according to Formula (11). X_i does not find a better honey source when it reaches the threshold *Limit*; all of the bees will become scout bees to find a new honey source. The mathematical model for its location update is Formula (10).

3.6 Chaos map

Most of the population algorithms use Gaussian distribution or uniform distribution to initialize individuals, but the initial population produced by this method is not sensitive. The chaos theory can make the initial conditions more sensitive and can produce a more variable number range (Sayed et al. 2018). Therefore, many experts and scholars are working to construct different chaotic maps instead of random initialization. It has been confirmed in documents (Sayed et al. 2018; Gupta and Deep 2019; Li et al. 2018e) that chaotic maps will be used in most population intelligence algorithms to show better results. Five common chaotic map information is detailed in Table 1. We can set up different population initialization values by choosing different mapping functions.

4 ACWOA

This section describes the proposed ACWOA algorithm in detail, which integrates the artificial bee colony algorithm and chaotic map to improve the global search ability and adds nonlinear convergence factor and adaptive weight coefficient to improve its convergence rate.

4.1 Adding chaotic mapping

The quality of the initial population will directly affect the performance of the entire algorithm. Therefore, a good initial decision has been discussed by many scholars and experts. It is common to use chaotic map initialization instead of random average initialization in the process of optimization and improvement (Cai et al. Aug 2019; Sayed et al. Jul 2018; Li et al. 2018e). The experimental results are effective, and in this paper, a chaotic map, Logistic, is selected to initialize the population (Gandomi et al. Feb 2013).

4.2 Inserting artificial bee colony algorithm

In the process of optimizing the population intelligence algorithm, many experts and scholars improve the performance of the algorithm by mixing other algorithms. The ultimate goal is to improve the diversity and convergence rate of the algorithm. For example, the PSO algorithm is added to the WOA algorithm (Ding et al. 2018d). The fusion algorithm of PSO and biogeography-based optimization is applied in a novel computer-aided diagnosis (Zhang et al. 2015). The hybrid algorithm of the GWO algorithm and cellular automata is applied in urban growth simulation (Cao et al. Aug 2019). Experimental results show that the hybrid algorithm is improved in terms of local and global search capabilities. Inspired by this, this paper proposes to add the artificial bee colony algorithm to the traditional WOA algorithm to improve the detection ability and avoid falling into the local optimum.

In this paper, the whales are subjected to the behavior of collecting bees. A random dimension in the individual whale is updated in the position of Formula (10), the fitness function $F(X_i^*)$, and then the greedy strategy is adopted. The individuals who will carry out the whale movement, the individuals who take the honey movement, and the last moment individual whales make greedy decisions through the fitness function, retaining the individuals with the best fitness values, and the other two are directly eliminated. Its function model is as follows:

Table 1 Five common chaotic maps

Type	Form	Range
Chebyshev	$o_{i+1} = \cos(i \cos^{-1}(o_i))$	$(-1,1)$
Logistic	$o_{i+1} = \mathrm{co}_i(1 - o_i)$	(0,1)
Singer	$o_{i+1} = \mu\left(7.86o_i - 23.31o_i^2 + 28.75o_i^3 - 13.3002875o_i^4\right)\mu = 1.07$	(0,1)
Sine	$o_{i+1} = \frac{c}{4}\sin(\pi o_i),\ c = 2.3$	(0,1)
Tent	$o_{i+1} = \begin{cases} \frac{o_i}{0.7}, & o_i < 0.7 \\ \frac{10}{3}(1 - o_i), & o_i \geq 0.7 \end{cases}$	(0,1)

$$\overrightarrow{X_i}(t+1)=\begin{cases}\overrightarrow{X}_i(t) & \text{if } F(\overrightarrow{X}_i(t))\geq F(\overrightarrow{X_i}(t+1)) \text{ and } F(\overrightarrow{X}_i(t))\geq F(\overrightarrow{X_i^*}(t+1)) \quad L_i=L_i+1\\ \overrightarrow{X_i}(t+1) & \text{if } F(\overrightarrow{X_i}(t+1))\geq F(\overrightarrow{X}_i(t)) \text{ and } F(\overrightarrow{X_i}(t+1))\geq F(\overrightarrow{X_i^*}(t+1)) \quad L_i=0\\ \overrightarrow{X_i^*}(t+1) & \text{if } F(\overrightarrow{X_i^*}(t+1))\geq F(\overrightarrow{X}_i(t)) \text{ and } F(\overrightarrow{X_i^*}(t+1))\geq F(\overrightarrow{X_i}(t+1)) \quad L_i=0\end{cases} \tag{13}$$

where $\overrightarrow{X_i^*}(t+1)$ indicates the position at which the next moment of the bee movement. $\overrightarrow{X_i}(t)$ and $\overrightarrow{X_i}(t+1)$, respectively, represent the current position of the ith whale and the position of the whale movement at the next moment. $F(\cdot)$ represents its corresponding fitness function. Next, we set a statistic L_i (i = 1, 2, …, N) and a threshold *Limit*. When the individual position of the population changes ($\overrightarrow{X_i}(t+1)=\overrightarrow{X_i}(t+1)$ or $\overrightarrow{X_i}(t+1)=\overrightarrow{X_i^*}(t+1)$), L_i = 0. Otherwise, $L_i = L_i + 1$. If L_i = Limit, it means that the individual of the population has not changed position for a long time. At this point, this individual is converted into a "detection whale." The "detection whale" performs location updates according to Formula (10).

4.3 Nonlinear convergence factor

The most important parameter in the original WOA algorithm is A, which will directly determine the convergence speed and accuracy of the algorithm. However, it can be seen from Formula (2) that A is mainly determined by a. As the number of iterations increases, a decreases gradually, which leads to the enhancement of the local development ability of the algorithm and the convergence rate is accelerated, but the possibility of falling into local optimum will increase. However, although the global detection ability is strong at the initial stage of the iteration, the convergence speed is slow. To improve the convergence rate in the early stage, it is necessary to speed up the decrement of a. For another thing, the environment is complex and variable, and a linear decrement cannot fully reflect the nonlinear search. Therefore, a new type of nonlinear convergence factor is proposed here. The specific model is as follows:

$$\overrightarrow{a1}=2\cdot\left(-1+\frac{t}{t_{\max}}\right)^2 \tag{14}$$

Therefore, the motion update models of Formulas (1) and (8) are changed to Formulas (15) and (16), respectively:

$$\overrightarrow{X}(t+1)=\overrightarrow{X_{\text{best}}}(t)-\overrightarrow{a1}\cdot(2\cdot\overrightarrow{\text{rand}}-1)\cdot\overrightarrow{D} \quad \text{if} \quad q<0.5 \tag{15}$$

$$\overrightarrow{X}(t+1)=\overrightarrow{X}_{\text{rand}}(t)-\overrightarrow{a1}\cdot(2\cdot\overrightarrow{\text{rand}}-1)\cdot\overrightarrow{D2} \tag{16}$$

4.4 Adaptive weight coefficient

Literature studies (Eberhart and Shi 2000; Fan and Chiu 2007) illustrate that the weight coefficient directly affects the searchability of the algorithm. Therefore, the optimization of the algorithm by inserting the inertia weight coefficient is also a hot topic. The common point of the intelligent population algorithm is that as the number of iterations increases, the ability of local detection will continue to increase. The final ideal state is that all the populations are in the vicinity of the optimal value. The optimal value is the optimal individual position in the ideal state. To further reflect that the proportion of the optimal individual position becomes more and more important as the number of iterations increases, the adaptive weight coefficient is introduced. The following is the specific expression of the adaptive coefficient proposed in this paper.

$$w=\frac{t}{t_{\max}} \tag{17}$$

$$\overrightarrow{X}(t+1)=w\cdot\overrightarrow{X_{\text{best}}}(t)+\overrightarrow{D1}\cdot e^{bl}\cdot\cos(2\pi l) \tag{18}$$

Therefore, Formula (6) of the traditional WOA algorithm is updated to Formula (18).

The pseudo-code of the ACWOA algorithm is as follows, and the flowchart of the ACWOA algorithm is shown in Fig. 2.

Algorithm 2 The pseudo-code of ACWOA

```
Initialization {
        initializes t_max, b=1, Limit, L_i=0(i = 1, 2, 3,..., n), a, A, C, l, q and Test functions,
        The logistic chaotic map is used to initialize individual whale X_i (i = 1, 2, 3,..., n) ,
        calculate the fitness value,
        X_best = the best search agent
       }
Main loop {
    while (t < t_max)
        for1 each search agent
            update a, A, C, l and q
            if1(q < 0.5)
                if2(|A| < 1)
                   update the position whale_X_i by the Eq. (15)
                else if2(|A|≥ 1)
                     select a random agent (X_rand) update the position whale_X_i by the Eq. (16)
                end if2
            else if1 (q≥ 0.5)
                     update the position whale_X_i by the Eq. (18)
            end if1
            update the position newbee_X_i by the Eq. (11)
            make sure each search agent is valid
            select the best fitness from whale_X_i, newbee_X_i and X_i
            if3(the best fitness is X_i)
               L_i= L_i+1
            else
                update the position X_i
               L_i= 0
            end if3
        end for1
        for2 each search agent
            if4(L_i> Limit)
              update the position X_i by logistic chaotic map
                 L_i=0
            end if4
        end for2
        update X_best if there is a better solution
        t = t + 1
    end while
   }
return X_best
```

4.5 Experimental results and analysis of benchmark functions

To test the performance of the proposed ACWOA algorithm, this paper uses 20 standard benchmark functions and CEC2019 multimodal multi-objective benchmark functions for experimental detection and evaluation. The simulation experiment platform is run under Windows 10 (64-bit) operating system, Intel(R) Core(TM) i7-8750cpu, 2.20 Ghz main frequency, and 16 GB memory environment. Meanwhile, we use MATLAB 2014b programming software. The traditional WOA algorithm, three advanced improved WOA algorithms (CWOA, HWPSO, and HIWOA), and three classical population intelligent algorithms (PSO, MVO, and GWO) are selected for experimental performance comparison.

4.6 Set up the experiment and benchmark function

To reflect the fairness of the experiment, the number N of the solution is set to 30. The dimension dim is 30. (The number of dimensions of the partial test basis function is given.) The maximum number of iterations $t_{\max}$ is 1000, and each algorithm is separate on each function. The number of times each one all the fitness functions $T_{\max}$ is 10000 and the number of times the fitness functions is counted each time. After a lot of simulation experiments,

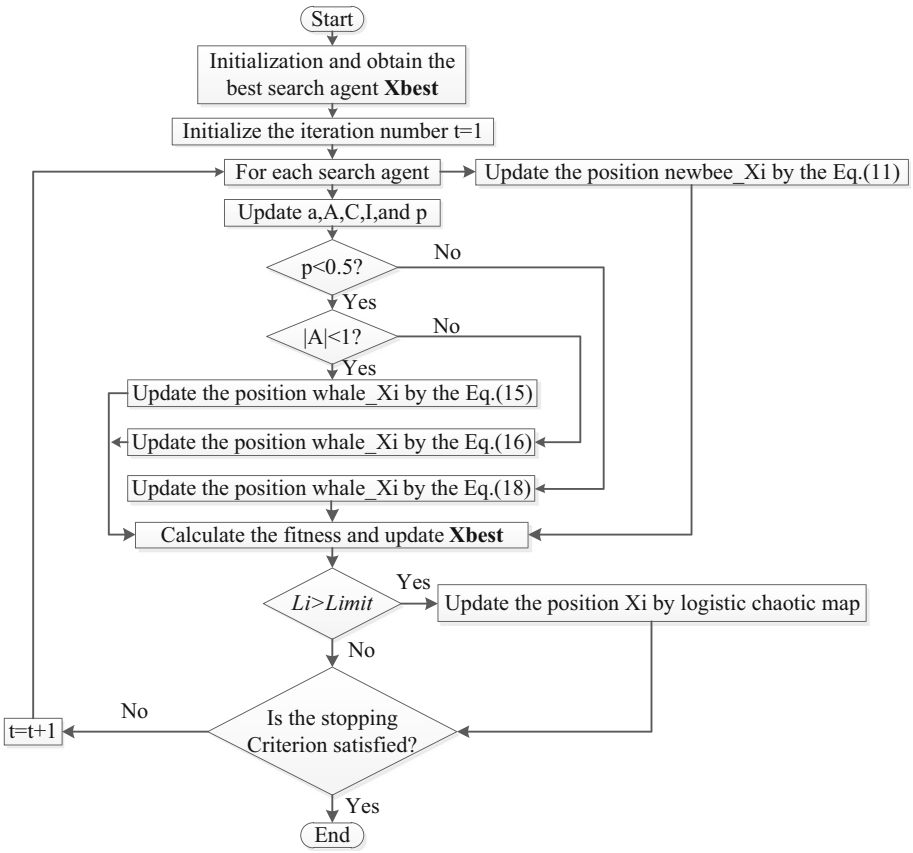

Fig. 2 Flowchart for the ACWOA algorithm

we set the threshold Limit = 20. The number of runs is N_r = 20 times, and the standard deviation, average value, and optimal value are averaged. Most of the parameters in this paper are based on the literature (Mirjalili and Lewis 2016). Other parameter values are obtained according to experimental experience. Document specifically describes the above (Kennedy and Eberhart 2002; Mirjalili et al. 2014, 2015; Mirjalili and Lewis 2016; Gandomi et al. 2013; Laskar et al. 2019; Tang et al. 2019) HIWOA, HWPSO, CWOA, WOA, PSO, MVO, and GWO.

Tables 2, 3, and 4 provide a detailed description of the 20 standard test basis functions used in this experiment. These functions are roughly divided into three categories: single-peak functions (the global optimal solution is set to have one and only one), the variable-dimensional multi-peak functions (when the default dimension dim is 30, the optimal solution has two or more), and the fixed dimension multimodal functions (in the case of a given dimension, there are two or more global optimal solutions). F1–F7 are the single-peak functions, F8–F12 are the variable-dimensional multi-peak functions, and F13-F20 are the fixed dimension multimodal functions.

To further verify the effectiveness of the ACWOA algorithm, we also add CEC2019 multimodal multi-objective benchmark functions. These functions are described in Table 5.

4.7 Evaluation measures

The metrics used to evaluate the performance of each algorithm are the average, standard, and optimal values of the fitness values, and the specific descriptions are given in Table 6.

4.8 Analysis of experimental results

This section compares and analyzes the proposed ACWOA algorithm with the other six optimization algorithms described above. Tables 7, 8, 9, 10, 11, and 12 provide the average values, the best values, and the standard deviations of the fitness values of the respective algorithms.

4.9 Average fitness value

As shown in Tables 7 and 8, the average values of the respective algorithms on the 30 test functions.

On the single-peak functions, the proposed ACWOA has the best performance on F1, F2, F3, F4, and F7. The proposed algorithm, HWPSO, performs well on F4 and F5.

Table 2 F1–F7 single-peak functions

Function	Dim	Range	$F_{\min}$
$F_1(x) = \sum_{i=1}^{n} x_i^2$	30	$[-100, 100]$	0
$F_2(x) = \sum_{i=1}^{n} \lvert x_i \rvert + \prod_{i=1}^{n} \lvert x_i \rvert$	30	$[-10, 10]$	0
$F_3(x) = \sum_{i=1}^{n} (\sum_{j-1}^{i} x_j)^2$	30	$[-100, 100]$	0
$F_4(x) = \max_i \{\lvert x_i \rvert, 1 \le i \le n\}$	30	$[-100, 100]$	0
$F_5(x) = \sum_{i=1}^{n-1} [100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2]$	30	$[-30, 30]$	0
$F_6(x) = \sum_{i=1}^{n} ([x_i + 0.5])^2$	30	$[-100, 100]$	0
$F_7(x) = \sum_{i=1}^{n} i x_i^4 + random[0, 1)$	30	$[-1.28, 1.28]$	0

On the variable-dimensional multi-peak functions, ACWOA has the best performance on F8, F9, F10, and F12, ranking second on F11. The WOA and ACWOA perform best on F8 and F10. The GWO has the best performance on F10. The PSO performs well on F11. HIWOA has the best performance on F8 and F10.

On the fixed dimension multimodal functions, all algorithms have the best performance on F15, F16, and F17. The ACWOA achieves the best performance on F13, F14 and F19, ranking third on F18 and F20. The WOA performs best on F18. The GWO performs best on F18 and F20. The MVO and HIWOA both achieve the best solution on F13.

On the CEC2019 test functions, the ACWOA has the best performance on F21, F22, F23, F25, F26, and F28, ranking second on F27 and F30 and ranking third on F24 and F29. GWO, PSO, MVO, and HIWOA perform well on functions F24, F30, F27, and F21, respectively.

4.10 Best fitness value

Tables 9 and 10 provide the best values of the respective algorithms on the 30 test functions.

On the single-peak functions, the proposed ACWOA has the best performance on F1, F2, F3, F4, F5, and F7, ranking third on F6. The proposed algorithms, HWPSO and HIWOA, perform well on F6 and F4, respectively.

On the variable-dimensional multi-peak functions, ACWOA has the best performance on F8, F9, F10, and F12, ranking third on F11. However, PSO achieves the best performance on F11. WOA, CWOA, HWPSO, GWO, and HIWOA perform well on functions F8 and F10.

On the fixed dimension multimodal functions, all algorithms have the best performance on F13 and F15–F20. The ACWOA has the best performance on F14.

On the CEC2019 test functions, the ACWOA has the best performance on F21, F22, F23, F24, F25, and F26, ranking second on F27 and F30 and ranking third on F29. GWO performs well on functions F21, F28, and F29. PSO has the best performance on F26 and F30. MVO and HIWOA achieve the best performance on F27 and F21, respectively.

4.11 Standard deviation

To test the stability of the algorithm, the standard deviation is used to measure the performance of the algorithm on the 30 test functions, as shown in Tables 11 and 12.

On the single-peak functions, the proposed ACWOA has the best performance on F1, F2, F3, F5, and F7, ranking second on F4 and F6. The proposed algorithm, HIWOA, performs well on F1 and F4. HWPSO has the best performance on F6.

On the variable-dimensional multi-peak functions, ACWOA has the best performance on F8, F9, F10, and F12, ranking second on F11. However, the PSO achieves the best performance on F11. WOA, CWOA, and HIWOA perform well on functions F8 and F10. GWO performs well on function F10.

On the fixed dimension multimodal functions, ACWOA ranks second on F20 and ranks third on F17. ACWOA has the best performance on other functions. The proposed algorithm, HWPSO, performs well on F15 and F16. PSO achieves the best performance on F15, F16, and F17. GWO performs well on function F20.

On the CEC2019 test functions, the ACWOA has the best performance on F21, F26, and F28, ranking second on functions F22, F24, F29, and F30. GWO achieves the best performance on F26, F27, and F29. MVO has the best performance on F24 and F30. CWOA and HIWOA achieve the best performance on F23 and F22, respectively.

4.12 Convergence comparison

To compare the convergence speed of the various algorithms, one single-peak function F7, one localization multimodal function F9, one nondeterministic multimodal function F14, and one CEC2019 test function F21 are given. As shown in Fig. 3, with the increase in the number of iterations, the fitness value of ACWOA algorithm

Table 3 F8–F12 variable-dimensional multi-peak functions

Function	Dim	Range	$F_{\min}$
$F_8(x)=\sum_{i=1}^{n}[x_i^2-10\cos(2\pi x_i)+10]$	30	$[-5.12,5.12]$	0
$F_9(x)=-20\exp\left(-0.2\sqrt{\frac{1}{n}\sum_{i=1}^{n}x_i^2}\right)-\exp\left(\frac{1}{n}\sum_{i=1}^{n}\cos(2\pi x_i)\right)+20+e$	30	$[-32,32]$	0
$F_{10}(x)=\frac{1}{4000}\sum_{i=1}^{n}x_i^2-\prod_{i=1}^{n}\cos(\frac{x_i}{\sqrt{i}})+1$	30	$[-600,600]$	0
$F_{11}(x)=\frac{\pi}{n}\{10\sin(\pi y_1)+\sum_{i=1}^{n-1}(y_i-1)^2[1+10\sin^2(\pi y_{i+1})]+(y_n-1)^2\}+\sum_{i=1}^{n}u(x_i,10,100,4)$ $y_i=1+\frac{x_i+1}{4}$ $u(x_i,a,k,m)=\begin{cases}k(x_i-a)^m & x_i>a\\ 0 & -a<x_i<a\\ k(-x_i-a)^m & x_i<-a\end{cases}$	30	$[-50,50]$	0
$F_{12}(x)=0.1\{\sin^2(3\pi x_1)+\sum_{i=1}^{n}(x_i-1)^2[1+\sin^2(3\pi x_i+1)]+(x_n-1)^2[1+\sin^2(2\pi x_n)]\}+\sum_{i=1}^{n}u(x_i,5,100,4)$	30	$[-50,50]$	0

Table 4 F13–F20 fixed dimension multimodal functions

Function	Dim	Range	$F_{\min}$
$F_{13}(x)=\left(\frac{1}{500}+\sum_{j=1}^{25}\frac{1}{j+\sum_{i=1}^{2}(x_i+a_{ij})^6}\right)^{-1}$	2	$[-65,65]$	1
$F_{14}(x)=\sum_{i=1}^{11}[a_i-\frac{x_1(b_i^2+b_ix_2)}{b_i^2+b_ix_3+x_4}]^2$	4	$[-5,5]$	0.00030
$F_{15}(x)=4x_1^2-2.1x_1^4+\frac{1}{3}x_1^6+x_1x_2-4x_2^2+4x_2^4$	2	$[-5,5]$	$-$ 1.0316
$F_{16}(x)=\left(x_2-\frac{5.1}{4\pi^2}x_1^2+\frac{5}{\pi}x_1-6\right)^2+10\left(1-\frac{1}{8\pi}\right)\cos x_1+10$	2	$[-5,5]$	0.398
$F_{17}(x)=[1+(x_1+x_2+1)^2(19-14x_1+3x_1^2-14x_2+6x_1x_2+3x_2^2)]\times[30+(2x_1-3x_2)^2\times(18-32x_1+12x_1^2+48x_2-36x_1x_2+27x_2^2)]$	2	$[-2,2]$	3
$F_{18}(x)=-\sum_{i=1}^{4}c_i\exp\left(-\sum_{j=1}^{3}a_{ij}(x_j-p_{ij})^2\right)$	3	$[1,3]$	$-$ 3.86
$F_{19}(x)=-\sum_{i=1}^{4}c_i\exp\left(-\sum_{j=1}^{6}a_{ij}(x_j-p_{ij})^2\right)$	6	$[0,1]$	$-$ 3.32
$F_{20}(x)=-\sum_{i=1}^{5}[(X-a_i)(X-a_i)^T+c_i]^{-1}$	4	$[0,10]$	$-$ 10.1532

decreases rapidly, indicating that the convergence speed is significantly accelerated, which is shown in all four functions. ACWOA algorithm has the potential to jump out of local optimization. Among functions F7, F9, F14, and F21, ACWOA algorithm has strong convergence, and its results are very good.

4.13 Statistical comparison

To further compare the performance of each algorithm, we use the Mann–Whitney U test and Friedman's test. The Mann–Whitney U test results of ACWOA and the other algorithms are illustrated in Table 13. The marks ($+$ / $\approx$/ $-$) are used to describe the performance of the algorithms. Compared with the WOA algorithm, the ACWOA has better performance on 26 functions and is "tied" on 4 functions. Compared with the HIWOA algorithm, the proposed algorithm "wins" on 23 functions, is "tied" on 5 functions, and has no "loss." Compared with the CWOA, HWPSO, GWO, PSO, and MVO, the ACWOA "wins" on 24 functions, 24 functions, 23 functions, 24 functions, and 25 functions, respectively.

To further reflect the performance of each algorithm, we use Friedman's test to compare all the algorithms. The Friedman's test results of eight algorithms in Table 14 demonstrate that the ACWOA has the best Friedman rank. Meanwhile, p value is 1.50e-05, which means the performance of ACWOA is different than other algorithms.

5 Application of ACWOA in a classical mathematical model

To further verify that the ACWOA algorithm has certain practical value in complex scenarios. The proposed algorithm, the classical intelligent population algorithms (PSO,

Table 5 The 100-digit challenge basic test functions

Function	Dim	Range	$F_{\min}$
$F_{21}(x)$ Storn's Chebyshev polynomial fitting problem	9	$[-8192, 8192]$	1
$F_{22}(x)$ Inverse Hilbert matrix problem	16	$[-16384, 16384]$	1
$F_{23}(x)$ Lennard-Jones minimum energy cluster	18	$[-4, 4]$	1
$F_{24}(x)$ Rastrigin's function	10	$[-100, 100]$	1
$F_{25}(x)$ Griewank's function	10	$[-100, 100]$	1
$F_{26}(x)$ Weierstrass function	10	$[-100, 100]$	1
$F_{27}(x)$ modified Schwefel's function	10	$[-100, 100]$	1
$F_{28}(x)$ expanded Schaffer's F6 function	10	$[-100, 100]$	1
$F_{29}(x)$ Happy cat function	10	$[-100, 100]$	1
$F_{30}(x)$ Ackley function	10	$[-100, 100]$	1

Table 6 Evaluation of measurement

Type	Quick description	Expression
Average value	Calculate the average value for N_r = 30 times	$\text{Ave} = \frac{1}{N_r}\sum_{i=1}^{N_r} F(\overrightarrow{X_{\text{best}}^i}(t))$
Standard value	The best fitness value for running N_r times	$\text{Best} = \min_{1 \le i \le N_r} F(\overrightarrow{X_{\text{best}}^i}(t))$
Optimal value	The deviation of the fitness value of the running N_r times from the average value	$\text{STD} = \sqrt{\frac{1}{N_r - 1}\sum_{i=1}^{N_r} (F(\overrightarrow{X_{\text{best}}^i}(t)) - \text{Ave})^2}$

MVO, and GWO), and the recent state-of-the-art algorithms (CWOA, HWPSO, and HIWOA) are applied to two well-known mathematical models (the spring design and the photovoltaic design) and a real application (the control process of a welding production line) (Chen et al. 2019). In this experiment, each algorithm is iterated 1000 times, run 30 times separately, and take the optimal solution. Since these mathematical models have multiple constraints, each algorithm needs to find the most suitable and optimal combination according to these conditions to obtain the optimal fitness value. As known in the literature (Coello 2002), the penalty functions are divided into static penalty, dynamic penalty, adaptive penalty, death penalty, etc. In this paper, the static penalty proposed by Hoffmeister and Sprave is used as the fitness value in the spring model, and the death penalty is used as the penalty function in the photovoltaic model. A detailed description of the penalty function is found in Coello (2002).

5.1 Spring design problem

The spring design problem is a classic problem in mathematical models (Dhar and Introduction to Optimum Design[M]. xxxx). The problem is the three main variable factors: wire diameter d, average coil diameter D, and the number of active coils Num. The minimum value of the spring–mass is obtained by a reasonable combination with each other. The relevant constraints that exist here are as follows:

$$\text{Variable} \quad \overrightarrow{x} = [x_1, x_2 x_3] = [d \cdot D \cdot \text{Num}]$$

$$\text{Objective} \quad f(\overrightarrow{x})_{\min} = x_1^2 x_2 x_3 + 2x_1^2 x_2$$

$$\text{Restrictions to} \; g_1(\overrightarrow{x}) = 1 - \frac{x_2^3 x_3}{71875 x_1^4} \le 0,$$

$$g_2(\overrightarrow{x}) = \frac{4x_2^2 - x_1 x_2}{12566(x_2 x_1^3 - x_1^4)} + \frac{2.46}{12566 x_1^2} - 1 \le 0,$$

$$g_3(\overrightarrow{x}) = 1 - \frac{140.54 x_1}{x_2^2 x_3} \le 0,$$

$$g_4(\overrightarrow{x}) = \frac{x_1 + x_2}{1.5} - 1 \le 0$$

$$\text{Ranges}: 0.05 \le x_1 \le 2.00$$
$$0.25 \le x_2 \le 1.30$$
$$2.00 \le x_3 \le 15.0$$

We conduct 30 experiments on each algorithm. It can be seen from Table 9 that the proposed ACWOA algorithm has the best penalty function value compared with other algorithms. We set d, D, and Num to 0.05231571, 0.37198223, and 10.4602112, respectively, and obtain the minimum value of this optimization task among the above algorithms. The HIWOA, CWOA, and HWPSO rank 4th,

Table 7 Average values of the respective algorithms

Function		WOA	CWOA	HWPSO	GWO	ACWOA	PSO	MVO	HIWOA
F1	Mean	1.9e−157	1.4e−154	3.6e−74	2.7e−59	**0**	6.2e−109	0.34	1.6e−170
	Rank	3	4	6	7	**1**	5	8	2
F2	Mean	1.3e−101	1.2e−104	1.2e−52	7.0e−35	**1.2e−319**	4.8e−4	0.41	1.0e−125
	Rank	4	3	5	6	**1**	7	8	2
F3	Mean	1.70e + 4	3.34e + 4	7.32	4.9e−16	**0**	14.39	47.37	1.50e−75
	Rank	7	8	4	3	**1**	5	6	2
F4	Mean	56.94	52.78	1.37	2.0e−14	**6.19e−45**	0.70	0.92	1.86e−27
	Rank	8	7	6	3	**1**	4	5	2
F5	Mean	27.14	27.51	**21.06**	26.76	22.92	65.21	548.3	27.40
	Rank	4	6	**1**	3	2	7	8	5
F6	Mean	0.09	0.07	**1.31e−10**	0.53	2.25e−8	1.56e−7	0.31	0.03
	Rank	6	5	**1**	8	2	3	7	4
F7	Mean	2.02e−3	7.02e−4	0.01	6.28e−4	**5.56e−5**	0.08	0.02	1.87e−4
	Rank	5	4	6	3	**1**	8	7	2
F8	mean	**0**	**0**	60.10	0.66	**0**	47.76	128.8	**0**
	Rank	**1**	**1**	7	5	**1**	6	8	**1**
F9	Mean	5.15e−15	3.73e−15	0.13	1.6e−14	**8.88e−16**	0.12	1.12	4.09e−15
	Rank	4	2	7	5	**1**	6	8	3
F10	Mean	**0**	**0**	2.96e−3	**0**	**0**	9.10e−3	0.66	**0**
	Rank	**1**	**1**	6	**1**	**1**	7	8	**1**
F11	Mean	0.03	4.75e−3	0.34	0.04	1.61e−09	**4.44e−11**	1.78	1.56e−3
	Rank	5	4	6	7	2	**1**	8	3
F12	Mean	0.20	0.20	3.30e−3	0.43	**1.10e−3**	5.49e−3	0.07	0.04
	Rank	6	6	2	8	**1**	3	5	4
F13	Mean	3.15	2.37	2.37	4.52	**1.00**	3.86	**1.00**	**1.00**
	Rank	5	4	4	7	**1**	6	**1**	**1**
F14	Mean	8.94e−4	6.18e−4	6.03e−3	6.45e−3	**3.08e−4**	3.90e−4	6.3e−3	6.01e−4
	Rank	5	4	6	8	**1**	2	7	3
F15	Mean	**−1.0316**	**−1.0316**	**−1.0316**	**−1.0316**	**−1.0316**	**−1.0316**	**−1.0316**	**−1.0316**
	Rank	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**
F16	Mean	**0.398**	**0.398**	**0.398**	**0.398**	**0.398**	**0.398**	**0.398**	**0.398**
	Rank	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**
F17	Mean	**3.00**	**3.00**	**3.00**	**3.00**	**3.00**	**3.00**	**3.00**	**3.00**
	Rank	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**
F18	Mean	**−3.8604**	−3.8607	−3.8612	**−3.8596**	−3.8605	−3.8620	−3.8628	−3.8575
	Rank	**1**	4	5	**1**	3	6	8	7
F19	Mean	−3.2642	−3.2713	−3.2628	−3.2616	**−3.2982**	−3.2744	−3.25	−3.2348
	Rank	4	3	5	6	**1**	2	7	8
F20	Mean	−7.3604	−8.6224	−7.3617	**−9.6473**	−8.565	−8.1231	-7.376	−7.2907
	Rank	7	2	6	**1**	3	4	5	8

5th, and 6th, respectively, defeating the traditional WOA algorithm. The GWO algorithm has good performance in this model, ranking second. However, the MVO algorithm is not suitable for this model, and the result is the worst. Therefore, the ACWOA algorithm can be a very effective auxiliary tool when solving the spring problem. Figure 4 shows the box diagram of the experimental results of each algorithm tested on this model. The mean and median values of the ACWOA algorithm are 0.01304 and 0.01313, respectively. ACWOA has the best performance. Although the mean and median values of the WOA algorithm are 0.01352 and 0.01361, respectively, the traditional WOA

Table 8 Average values of the respective algorithms in CEC2019

Function		WOA	CWOA	HWPSO	GWO	ACWOA	PSO	MVO	HIWOA
F21	Mean	3.6e+6	4.7e+7	4.6e+6	1.896e+3	**1**	2.01e+8	4.589e+9	**1**
	Rank	4	6	5	3	**1**	7	8	**1**
F22	Mean	8.94e+3	9.61e+3	4.90e+3	3.87e+2	**4.40**	2.327e+4	4.34e+2	4.94
	Rank	6	7	5	3	**1**	8	4	2
F23	Mean	4.20	4.79	3.29	2.49	**2.15**	3.39	5.60	2.86
	Rank	6	7	4	2	**1**	5	8	3
F24	Mean	59.22	48.28	45.47	**15.25**	21.25	35.13	18.23	46.12
	Rank	8	7	5	**1**	3	4	2	6
F25	Mean	2.05	1.98	1.20	1.60	**1.12**	1.16	1.30	2.87
	Rank	7	6	3	5	**1**	2	4	8
F26	Mean	8.21	9.28	7.31	2.42	**2.00**	3.65	2.05	7.33
	Rank	7	8	5	3	**1**	4	2	6
F27	Mean	1.50e+3	1.25e+3	1.24e + 3	7.44e+2	9.06 e+2	1.16e+3	**6.91 e+2**	1.19e+3
	Rank	8	7	6	3	2	4	**1**	5
F28	Mean	4.58	4.57	4.33	3.61	**2.93**	3.95	3.71	4.54
	Rank	8	7	5	2	**1**	4	3	6
F29	Mean	1.36	1.33	1.42	**1.15**	1.19	1.16	1.21	1.31
	Rank	7	6	8	**1**	3	2	4	5
F30	Mean	21.12	21.15	21.05	21.40	20.07	**17.08**	21.02	21.32
	Rank	5	6	4	8	2	**1**	3	7

Bold value indicates the algorithm with the best performance among all algorithms

has the best stability. MVO has the worst performance in this model.

5.2 Photovoltaic design problem

The photovoltaic design model is similar to the spring design model described above. The difference is that there are four dimensions in this model: shell depth d_s, head depth d_h, inner radius R, and interface width L. The goal is to find the best solution to minimize the total cost (Mirjalili and Lewis 2016). The detailed description and constraints are as follows:

Variable $\vec{x}=[x_1, x_2, x_3, x_4] = [d_s\ d_h\ R\ L]$

Objective

$$f(\vec{x})_{\min} = 0.6224\mathrm{d}_s RL + 1.7781R^2 d_h + 3.1661 d_s^2 L + 19.84 d_h^2 L$$

Restrictions to

$$g_1(\vec{x}) = -d_s + 0.0193R \le 0,$$
$$g_2(\vec{x}) = -d_h + 0.00954R \le 0,$$
$$g_3(\vec{x}) = -\pi LR^2 - \frac{4}{3}\pi R^3 + 1296000 \le 0,$$
$$g_4(\vec{x}) = L - 240 \le 0$$

Ranges :

$$0 \le x_1 \le 99$$
$$0 \le x_2 \le 99$$
$$10 \le x_3 \le 200$$
$$10 \le x_4 \le 200$$

We conduct 30 experiments on each algorithm. The results of the various algorithms in this optimization task are given in Table 10. ACWOA is better than the PSO algorithm, ranking first, with a minimum of 5345.116. d_s, d_h,R, and L are 1.2588, 0.6222, 65.2252, and 10.000, respectively. The recent state-of-the-art algorithms (CWOA (ranking 6th), HWPSO (ranking 7th), and HIWOA (ranking 3rd)) are better than WOA. The classical intelligent population algorithms (like PSO (ranking 2nd), MVO (ranking 4th), and GWO (ranking 5th)) are effective in this model. Therefore, ACWOA algorithm still has certain practical value in photovoltaic design. Figure 5 shows the box diagram of the experimental results of each algorithm tested on this model. The mean and median of ACWOA are 5388.137 and 5398.5536, respectively. The mean and median of PSO are 5388.317 and 539.804, respectively. ACWOA was the best on average, but PSO was better than ACWOA on median. The traditional WOA algorithm is not good in photovoltaic design model.

Table 9 Best values of the respective algorithms

Function		WOA	CWOA	HWPSO	GWO	ACWOA	PSO	MVO	HIWOA
F1	Best	1.1e−168	2.8e−169	3.55e−79	7.1e−61	**0**	1.3e−10	0.22	2.6e−200
	Rank	4	3	5	6	**1**	7	8	2
F2	Best	1.2e−113	2.6e−115	2.40e−50	1.5e−35	**0**	5.24e−6	0.23	1.0e−133
	Rank	4	3	5	6	**1**	7	8	2
F3	Best	8.33e+4	1.4e+5	1.50	4.5e−19	**0**	4.78	19.43	8.65e−79
	Rank	7	8	4	3	**1**	5	6	2
F4	Best	1.42	6.57	0.52	1.7e−15	**0**	0.54	0.66	**0**
	Rank	7	8	4	3	**1**	5	6	**1**
F5	Best	26.63	26.76	20.49	26.20	**19.58**	24.64	29.40	27.10
	Rank	5	6	2	4	**1**	3	8	7
F6	Best	0.02	0.01	**3.04e−12**	1.19e−5	1.0e−08	1.3e−11	0.21	0.02
	Rank	6	5	**1**	4	3	2	8	6
F7	Best	1.14e−4	1.49e−4	2.74e−3	3.68e−4	**1.38e−5**	0.05	8.0e−3	4.37e−5
	Rank	3	4	6	5	**1**	8	7	2
F8	Best	**0**	**0**	**0**	**0**	**0**	25.87	98.64	**0**
	Rank	**1**	**1**	**1**	**1**	**1**	7	8	**1**
F9	Best	8.88e−16	8.65e−16	4.44e−15	1.5e−14	**7.9e−16**	8.9e−06	0.19	8.12e−16
	Rank	4	3	5	6	**1**	7	8	2
F10	Best	**0**	**0**	**0**	**0**	**0**	4.6e−12	0.49	**0**
	Rank	**1**	**1**	**1**	**1**	**1**	7	8	**1**
F11	Best	1.51e−3	1.65e−3	2.72e−10	0.02	7.2e−10	**3.3e−12**	6.8e−3	9.5e−4
	Rank	5	6	2	8	3	**1**	7	4
F12	Best	0.07	0.03	2.63e−12	0.20	**1.4e−14**	6.85e−13	0.02	0.02
	Rank	7	6	3	8	**1**	2	4	4
F13	Best	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**
	Rank	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**
F14	Best	3.16e−4	3.12e−4	3.07e−4	3.08e−4	**3.05e−4**	8.61e−4	3.09e−4	3.13e−4
	Rank	7	5	2	3	**1**	8	4	6
F15	Best	**−1.0316**	**−1.0316**	**−1.0316**	**−1.0316**	**−1.0316**	**−1.0316**	**−1.0316**	**−1.0316**
	Rank	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**
F16	Best	**0.398**	**0.398**	**0.398**	**0.398**	**0.398**	**0.398**	**0.398**	**0.398**
	Rank	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**
F17	Best	**3.00**	**3.00**	**3.00**	**3.00**	**3.00**	**3.00**	**3.00**	**3.00**
	Rank	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**
F18	Best	**−3.86**	**−3.86**	**−3.86**	**−3.86**	**−3.86**	**−3.86**	**−3.86**	**−3.86**
	Rank	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**
F19	Best	**−3.32**	**−3.32**	**−3.32**	**−3.32**	**−3.32**	**−3.32**	**−3.32**	**−3.32**
	Rank	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**
F20	Best	**−10.1532**	**−10.1532**	**−10.1532**	**−10.1532**	**−10.1532**	**−10.1532**	**−10.1532**	**−10.1532**
	Rank	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**

Bold value indicates the algorithm with the best performance among all algorithms

5.3 The control process of a welding production line

We further prove the performance of the ACWOA algorithm in a practical application (the quality process control of automated welding of car bodies in Geely cars). At present, the traditional scheme is to use a simple control chart to control the hole center, *X, Y, Z* coordinate size. However, there are some disadvantages of this scheme. On the one hand, it does not effectively consider the economic situation. On the other hand, it can only control one-dimensional independent variables, resulting in poor performance. In order to obtain better performance, we design a multivariate Bayesian VSI control chart. Some of the

Table 10 Best values of the respective algorithms in CEC2019

Function	Criteria	WOA	CWOA	HWPSO	GWO	ACWOA	PSO	MVO	HIWOA
F21	Best	1.84e+3	29.93	16.74	**1**	**1**	6.407e+7	3.245e+4	**1**
	Rank	6	5	4	**1**	**1**	8	7	**1**
F22	Best	5.13e+3	3.34e+3	2.13e + 2	38.47	**4.22**	1.858e+4	2.77e+2	4.36
	Rank	7	6	4	3	**1**	8	5	2
F23	Best	1.46	2.62	1.40	1.44	**1.28**	1.41	1.45	1.31
	Rank	7	8	3	5	**1**	4	6	2
F24	Best	23.93	24.97	22.89	4.99	**4.98**	11.95	7.97	22.11
	Rank	7	8	6	2	**1**	4	3	5
F25	Best	1.47	1.51	1.06	1.14	**1.04**	1.07	1.09	2.26
	Rank	6	7	2	5	**1**	3	4	8
F26	Best	5.10	5.84	3.63	1.39	**1.00**	**1.00**	1.16	4.40
	Rank	7	8	5	4	**1**	**1**	3	6
F27	Best	1001.19	765.45	797.45	500.32	382.8	459.40	**238.99**	474.39
	Rank	8	6	7	5	2	3	**1**	4
F28	Best	3.93	3.90	3.78	**2.67**	3.52	3.09	3.11	4.11
	Rank	7	6	5	**1**	4	2	3	8
F29	Best	1.202	1.139	1.146	**1.081**	1.100	1.084	1.105	1.175
	Rank	8	5	6	**1**	3	2	4	7
F30	Best	21.05	21.03	21.00	21.31	20.98	**1.00**	21.01	21.18
	Rank	6	5	3	8	2	**1**	4	7

Bold value indicates the algorithm with the best performance among all algorithms

economic parameters are calculated through empirical data, but the statistical parameters can only be obtained by using population intelligence algorithms. The quality control model is described as follows:

Variable $\overrightarrow{x}=[h_1\, h_2\, n\, p_{s1}.\, p_{s2}]$

Restrictions to $h_1 \geq h_2$

$p_{S1} \geq p_{S2}$

$n \in N^*$

The economic parameters are the results obtained after a lot of experiment, such as fixed cost ($b = 4.9$), variable sampling ($c = 0.6$), cost in the out-of-control state

Objective

$$\mathrm{ECT} = \frac{\text{Expected Cost}}{\text{Expected Time}}$$

$$= \sum_{i=1,2} \left\{ b + \sum_{p \leq p_{R_i}} \pi_{0p}\left[M\left[h_1 - \left(1 - e^{-\theta h_1}\right)/\theta\right] + cn\right] + \sum_{p_{R_i} < p \leq p_{S_i}} \pi_{0p}\left[M\left[h_2 - \left(1 - e^{-\theta h_2}\right)/\theta\right] + cn\right] \right.$$
$$+ \sum_{p > p_{S_i}} \pi_{0p}\left[M\left[h_1 - \left(1 - e^{-\theta h_1}\right)/\theta\right] + cn + A\right] + \sum_{p \leq p_{R_i}} \pi_{1p}(Mh_1 + cn)$$
$$\left. + \sum_{p_{R_i} < p \leq p_{S_i}} \pi_{1p}(Mh_2 + cn) + \sum_{p > p_{S_i}} \pi_{1p}\left[M\left[h_1 - \left(1 - e^{-\theta h_1}\right)/\theta\right] + cn + R\right] \right\}$$
$$\div \sum_{i=1,2} \left\{ \sum_{p \leq p_{R_i}} \left(\pi_{0p} + \pi_{1p}\right)h_1 + \sum_{p_{R_i} < p \leq p_{S_i}} \left(\pi_{0p} + \pi_{1p}\right)h_2 + \sum_{p > p_{S_i}} \pi_{0p}(h_1 + T_0) + \sum_{p > p_{S_i}} \pi_{1p}(h_1 + T_0 + T_1) \right\}$$

Table 11 Standard deviations of the respective algorithms

Function		WOA	CWOA	HWPSO	GWO	ACWOA	PSO	MVO	HIWOA
F1	Std	5.7e−157	4.3e−154	5.8e−74	3.3e−59	**0**	1.40e−8	0.09	**0**
	Rank	3	4	5	6	**1**	7	8	**1**
F2	Std	3.8e−101	3.5e−104	4.4e−50	6.4e−35	**0**	7.60e−4	0.12	3.1e−125
	Rank	4	3	5	6	**1**	7	8	2
F3	Std	7.0e+3	1.15e+4	4.56	7.25e−16	**0**	5.42	16.74	4.10e−75
	Rank	7	8	4	3	**1**	5	6	2
F4	Std	26.62	32.76	0.53	2.59e−14	1.27e−44	0.16	0.18	**0**
	Rank	7	8	6	3	2	4	5	**1**
F5	Std	0.23	0.67	0.46	0.58	**0.176**	56.91	945.20	0.182
	Rank	3	6	4	5	**1**	7	8	2
F6	Std	0.11	0.08	**1.53e−10**	0.43	7.40e−9	4.63e−7	0.07	9.38e−3
	Rank	7	6	**1**	8	2	3	5	4
F7	Std	1.56e−3	4.57e−4	5.11e−3	2.31e−4	**2.22e−5**	0.02	5.03e−3	9.66e−5
	Rank	5	4	7	3	**1**	8	6	2
F8	Std	**0**	**0**	50.83	1.32	**0**	14.11	24.97	**0**
	Rank	**1**	**1**	8	5	**1**	6	7	**1**
F9	Std	2.66e−15	2.13e−15	0.40	1.07e−15	**0**	0.35	0.60	2.49e−15
	Rank	5	3	7	2	**1**	6	8	4
F10	Std	**0**	**0**	6.01e−3	**0**	**0**	0.01	0.10	**0**
	Rank	**1**	**1**	6	**1**	**1**	7	8	**1**
F11	Std	0.06	3.51e−3	0.56	0.02	6.27e−10	**6.47e−11**	1.31	3.81e−4
	Rank	6	4	7	5	2	**1**	8	3
F12	Std	0.09	0.14	5.04e−3	0.15	**3.30e−3**	0.01	0.03	0.02
	Rank	6	7	2	8	**1**	3	5	4
F13	Std	3.85	2.91	2.87	4.17	**1.86e−16**	2.72	1.58e-11	1.10e−8
	Rank	7	6	5	8	**1**	4	2	3
F14	Std	6.08e−4	4.09e−4	6.03e−3	9.11e−3	**2.24e−12**	2.71e−4	0.02	5.59e−4
	Rank	6	3	5	7	**1**	2	8	4
F15	Std	1.00e−10	7.52e−11	**0**	8.86e−9	**0**	**0**	7.08e−8	9.51e−9
	Rank	5	4	**1**	6	**1**	**1**	8	7
F16	Std	1.27e−6	1.26e−7	**0**	1.40e−6	**0**	**0**	6.93e−8	9.81e−5
	Rank	6	5	**1**	7	**1**	**1**	4	8
F17	Std	6.35e−6	1.20e−5	7.69e−16	5.55e−6	1.74e−15	**1.99e−16**	3.96e−7	1.00e−4
	Rank	6	7	2	5	3	**1**	4	8
F18	Std	4.01e−3	2.25e−3	3.42e−3	3.82e−3	**6.12e−16**	8.9e−16	8.07e−7	7.95e−3
	Rank	7	4	5	6	**1**	2	3	8
F19	Std	7.21e−2	8.24e−2	7.92e−2	6.05e−2	**4.76e-2**	5.82e−2	5.88e−2	9.92e−2
	Rank	5	7	6	4	**1**	2	3	8
F20	Std	2.8733	2.3357	2.8745	**1.5156**	1.5294	2.4864	2.8586	2.8094
	Rank	7	3	8	**1**	2	4	6	5

Bold value indicates the algorithm with the best performance among all algorithms

(M = 111), cost of search (A = 115.5), cost of additional repairs (R = 349.5), θ = 0.013. $\pi_{0p} = 0.5$ and $\pi_{1p} = 0.5$ are the state probability and the steady-state probability, respectively. $T_0 = 1$ and $T_1 = 2$ are the investigation time and the expected time, respectively. Some variables that need to be solved are as follows: control variable $p_{Si}(i = 1, 2)$, sample size (n), and sampling period (h_1, h_2). It is worth noting that $p_{Ri} = 0.5 * p_{Si}(i = 1, 2)$ denotes the alarm

Table 12 Standard deviations of the respective algorithms in CEC2019

Function	Criteria	WOA	CWOA	HWPSO	GWO	ACWOA	PSO	MVO	HIWOA
F21	Std	5.34e+6	4.43e+6	5.19 e+6	5.21e+3	**0**	9.39e+7	4.13e + 5	1.39e−15
	Rank	7	5	6	3	**1**	8	4	2
F22	Std	2.54e+3	3.80e+3	3.22e+3	1.98e+2	0.30	4.06e+3	1.05e+2	**0.19**
	Rank	5	7	6	4	2	8	3	**1**
F23	Std	1.50	**0.95**	1.90	1.54	1.26	2.37	2.79	1.05
	Rank	4	**1**	6	5	3	7	8	2
F24	Std	25.12	20.68	14.27	7.96	6.08	15.93	**5.50**	14.78
	Rank	8	7	4	3	2	6	**1**	5
F25	Std	0.65	0.28	0.07	0.76	**0.04**	0.06	0.13	0.48
	Rank	7	5	3	8	**1**	2	4	6
F26	Std	1.96	2.37	1.78	**0.73**	1.02	1.50	0.76	1.86
	Rank	7	8	5	**1**	3	4	2	6
F27	Std	282.55	289.25	311.86	**200.00**	307.08	379.92	237.05	418.46
	Rank	3	4	6	**1**	5	7	2	8
F28	Std	0.34	0.31	0.51	0.37	**0.25**	0.50	0.38	0.27
	Rank	4	3	8	5	**1**	7	6	2
F29	Std	0.14	0.15	0.18	**0.05**	0.06	0.09	0.07	0.08
	Rank	6	7	8	**1**	2	5	3	4
F30	Std	0.06	0.07	0.11	0.05	0.04	8.04	**0.01**	0.09
	Rank	4	5	7	3	2	8	**1**	6

Bold value indicates the algorithm with the best performance among all algorithms

limits. We conduct 30 experiments on each algorithm. The ACWOA is compared with other algorithms, and the results are described in Fig. 6. We can get the best fitness of each algorithm from this figure (18.457 (WOA), 19.15 (CWOA), 17.15 (HWPSO), 21.4144 (GWO), 17.115 (ACWOA), 27.123 (PSO), 19.175 (MVO), and 19.4096 (HIWOA)). ACWOA has the best performance. (The mean is 19.171 and the median is 20.175.) PSO falls into local optimum in this model. Although it has the best stability, it has the worst accuracy.

6 Summary and prospect

Embedding a chaotic map and artificial bee colony algorithm into the WOA algorithm is the main idea of this paper. The feature that the artificial bee swarm algorithm can effectively jump out of the local optimum is added to the WOA, which effectively avoids the possibility that the algorithm falls into local optimum. It is also very effective to add chaotic maps to improve the quality of the initialized population. For another thing, the nonlinear convergence factor and the adaptive weight value are added to the position update of the whale to improve the convergence speed of the algorithm. Finally, the proposed improved algorithm is compared with classical and the recent state-of-the-art six algorithms through 30 standard benchmark functions, two classical mathematical models, and a practical application. It is evaluated by various metrics such as optimal value, mean value, variance value, convergence curve, and penalty function. The final experimental results show that the ACWOA algorithm has certain competitiveness in the optimization problem and beats all other algorithms in most benchmark functions to get the first. In the mathematical model and practical application, ACWOA also shows an outstanding side and has certain practical value. However, there are two shortcomings in ACWOA. One is that the threshold Limit is obtained experimentally and is not intelligent. In different practical problems, Limit will affect the detection ability of the population. How to design adaptive Limit is a challenge. Other one is that there is also the question of which mapping model to choose for initialization. We use the classic Logistic model. However, different mapping models construct different initializations, which will affect the final convergence result. In addition, the method can be further integrated and inserted with other various effective and practical algorithms to obtain a better hybrid algorithm in the future. Meanwhile, the calculation can be applied to

Table 13 Mann–Whitney U test results for 30 functions (significance level: $\alpha = 0.05$)

Function		Pairwise comparison ACWOA versus						
		WOA	CWOA	HWPSO	GWO	PSO	MVO	HIWOA
F1	Symbol	+	+	+	+	+	+	+
	p	8.00e−09	8.007e−09	8.007e−09	8.007e−09	8.007e−09	8.007e−09	8.01e−09
F2	Symbol	+	+	+	+	+	+	+
	p	5.350e−08	5.350e−08	5.350e−08	5.350e−08	5.350e−08	5.350e−08	5.350e−08
F3	Symbol	+	+	+	+	+	+	+
	p	8.01e−09	8.01e−09	8.01e−09	8.01e−09	8.01e−09	8.01e−09	8.01e−09
F4	Symbol	+	+	+	+	+	+	+
	p	4.95e−08	4.95e−08	4.95e−08	4.95e−08	4.95e−08	4.95e−08	4.95e−08
F5	Symbol	+	+	−	+	+	+	+
	p	6.796e−08	6.796e−08	1.235e−07	6.796e−08	1.61e−04	6.796e−08	6.796e−08
F6	Symbol	+	+	−	+	+	+	+
	p	3.577e−24	6.796e-08	6.796e−08	6.796e−08	9.209e−04	6.796e−08	6.796e−08
F7	Symbol	+	+	+	+	+	+	+
	p	2.062e−06	4.540e−06	6.796e−08	6.796e−08	6.796e−08	6.796e−08	8.357e−04
F8	Symbol	≈	≈	+	+	+	+	≈
	p	NaN	NaN	9.429e−06	0.002	8.007e−09	8.007e-09	NaN
F9	Symbol	+	+	+	+	+	+	+
	p	1.377e−04	2.375e−06	2.665e−09	3.835e−09	8.007e−09	8.007e−09	1.113e−07
F10	Symbol	+	≈	+	+	+	+	≈
	p	0.342	NaN	0.342	0.163	8.007e−09	8.007e−09	NaN
F11	Symbol	+	+	+	+	−	+	+
	p	6.796e−08	6.796e−08	0.0011	6.796e−08	1.807e−05	6.796e−08	6.796e−08
F12	Symbol	+	+	+	+	+	+	+
	p	6.796e−08	6.796e−08	0.0499	6.796e−08	5.255e−05	6.796e−08	6.796e−08
F13	Symbol	+	+	+	+	+	≈	≈
	p	1.512e−08	1.512e−08	0.413	1.512e−08	0.057	1.512e−08	1.512e−08
F14	Symbol	+	+	+	+	+	+	+
	p	1.251e−05	8.597e−06	8.357e−04	1.415e−05	1.600e−05	2.062e−06	2.041e−05
F15	Symbol	≈	≈	≈	≈	≈	≈	≈
	p	2.404e−08	2.404e−08	0.040	2.404e−08	0.164	2.404e−08	2.404e−08
F16	Symbol	≈	≈	≈	≈	≈	≈	≈
	p	3.597e−07	4.157e−08	3.597e−07	3.597e−07	5.127e−07	3.597e−07	3.597e−07
F17	Symbol	≈	≈	≈	≈	≈	≈	≈
	p	5.678e−08	5.678e−08	7.230e−05	5.678e−08	5.193e−04	5.678e−08	5.678e−08
F18	Symbol	−	+	+	−	+	+	+
	p	4.205e−08	4.205e−08	0.003	4.205e−08	0.002	4.205e−08	4.205e−08
F19	Symbol	+	+	−	+	+	+	+
	p	0.006	7.578e−04	0.273	0.006	0.294	7.578e−04	0.031
F20	Symbol	+	−	+	−	+	+	+
	p	0.003	0.049	0.002	6.038e−06	0.014	0.013	0.660
F21	Symbol	+	+	+	+	+	+	≈
	p	8.007e−09	8.007e−09	8.007e−09	8.007e−09	8.007e−09	8.007e−09	0.020
F22	Symbol	+	+	+	+	+	+	+
	p	6.70e−08	6.70e−08	6.70e−08	6.70e-08	6.70e−08	6.70e−08	8.204e−07
F23	Symbol	+	+	+	+	+	+	+
	p	0.001	1.997e−04	0.507	0.695	0.022	1.60e−05	0.057

Table 13 (continued)

Function		Pairwise comparison ACWOA versus						
		WOA	CWOA	HWPSO	GWO	PSO	MVO	HIWOA
F24	Symbol	+	+	+	−	+	−	+
	p	1.376e−06	1.803e-06	1.431e−07	6.868e−04	2.745e−04	0.096	7.948e−07
F25	Symbol	+	+	+	+	+	+	+
	p	6.796e−08	6.796e−08	6.610e−05	1.235e-07	0.011	6.015e−07	6.796e−08
F26	Symbol	+	+	+	+	+	+	+
	p	7.898e−08	1.918e−07	6.917e−07	0.968	0.756	0.208	1.065e−07
F27	Symbol	+	+	+	+	+	−	+
	p	2.222e−04	8.597e−06	3.293e−05	0.164	0.002	0.925	6.674e−06
F28	Symbol	+	+	+	+	+	+	+
	p	1.60e−05	2.960e−07	1.251e−05	0.032	0.003	0.925	1.803e−06
F29	Symbol	+	+	+	−	−	+	+
	p	1.997e−04	5.255e−05	1.610e−04	0.756	0.882	0.164	2.062e−06
F30	Symbol	+	+	+	+	−	+	+
	p	1.037e−04	1.794e−04	0.543	1.235e−07	0.006	0.068	2.066e−06
+/≈/−		26/4/0	24/5/1	24/3/3	23/3/4	24/3/3	25/3/2	23/7/0

Table 14 Friedman test results of eight algorithms on 30 functions

	WOA	CWOA	HWPSO	GWO	ACWOA	PSO	MVO	HIWOA	*p* value
Friedman rank	5.398	5.272	4.125	4.431	2.299	4.566	5.407	4.564	1.50e−05

Table 15 Experimental results of each algorithm in the spring design model

Types	Variable			Fitness value	Rank
	d	D	Num		
WOA	0.05828714	0.53721074	5.35101084	0.0134164	7
CWOA	0.05587575	0.46605444	6.92088004	0.0129805	5
HWPSO	0.05733385	0.50833926	5.91236631	0.0132216	6
GWO	0.50000000	0.31736225	14.0542133	0.0127375	2
ACWOA	**0.05231571**	**0.37198223**	**10.4602112**	**0.0126856**	**1**
PSO	0.05444798	0.42679117	8.12682362	0.0128130	3
MVO	0.06546942	0.78728646	2.70678486	0.0158831	8
HIWOA	0.05576711	0.46299336	7.00529556	0.0129667	4

Bold value indicates the algorithm with the best performance among all algorithms

Table 16 Experimental results of each algorithm in the photovoltaic design model

Types	Variable				Fitness value	Rank
	Ts	Th	R	L		
WOA	1.2877	0.6352	65.2252	10.000	5401.026	8
CWOA	1.2876	0.6246	65.2252	10.000	5377.714	6
HWPSO	1.2363	0.6111	64.0582	10.000	5389.554	7
GWO	1.2574	0.6217	65.0558	10.775	5364.024	5
ACWOA	**1.2588**	**0.6222**	**65.2252**	**10.000**	**5345.116**	**1**
PSO	1.5262	0.6709	65.2252	10.000	5345.188	2
MVO	1.2672	0.6217	65.1401	10.405	5357.704	4
HIWOA	1.2667	0.6241	65.2252	10.000	5352.592	3

Bold value indicates the algorithm with the best performance among all algorithms

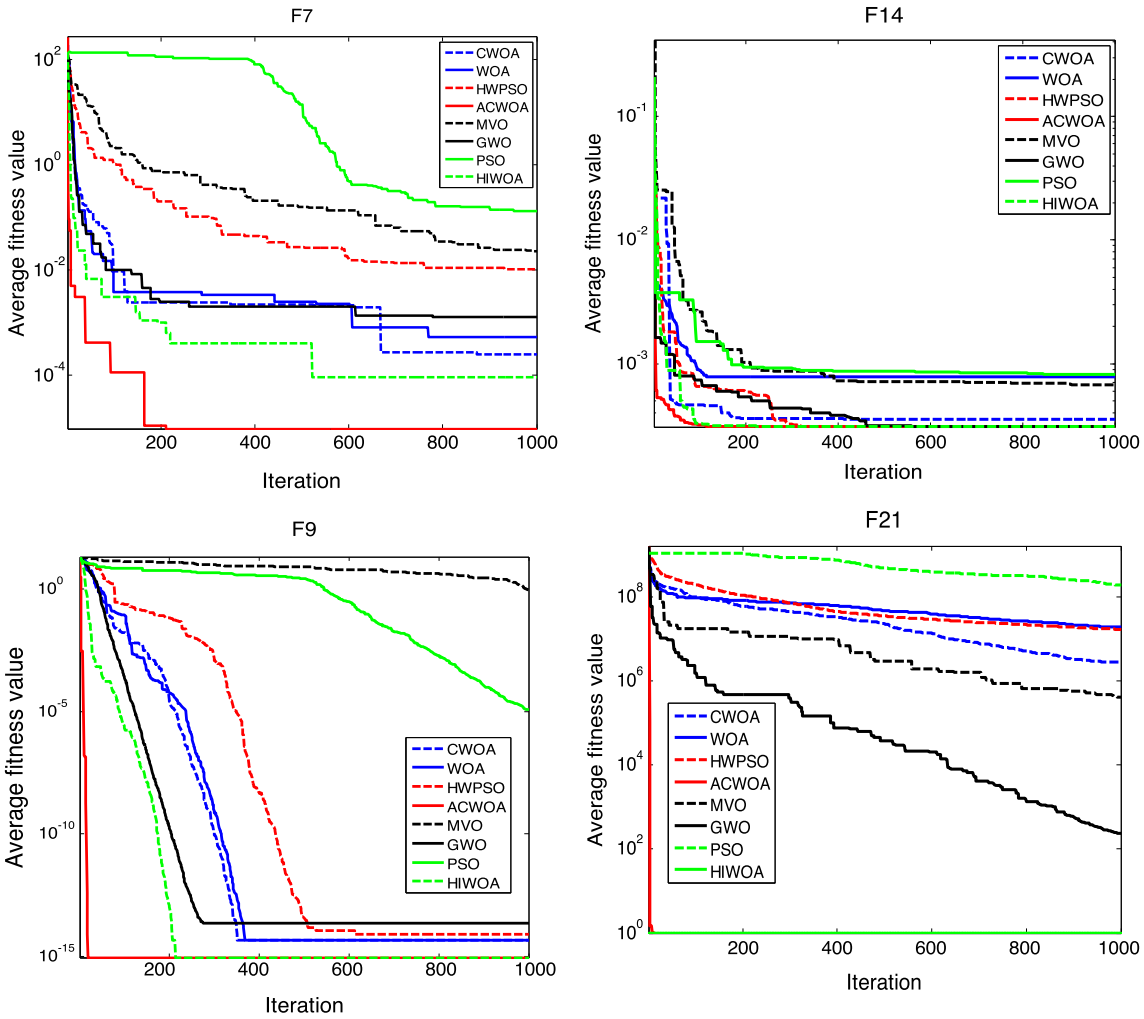

Fig. 3 Convergence curve of partial function

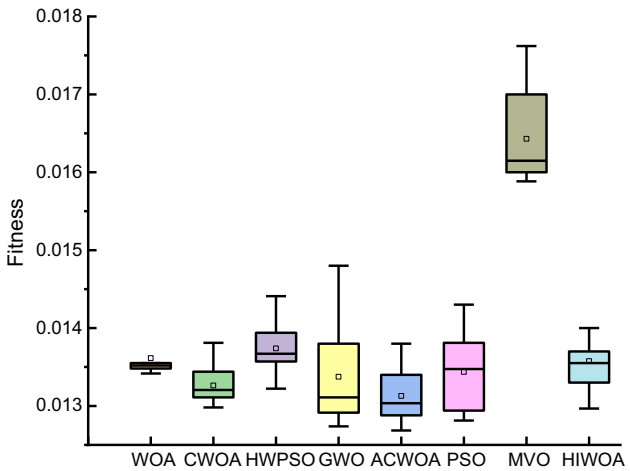

Fig. 4 Box diagram of the spring design model

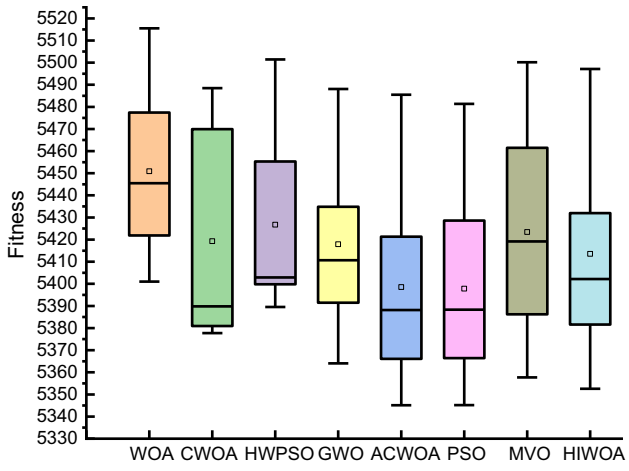

Fig. 5 Box diagram of photovoltaic design model

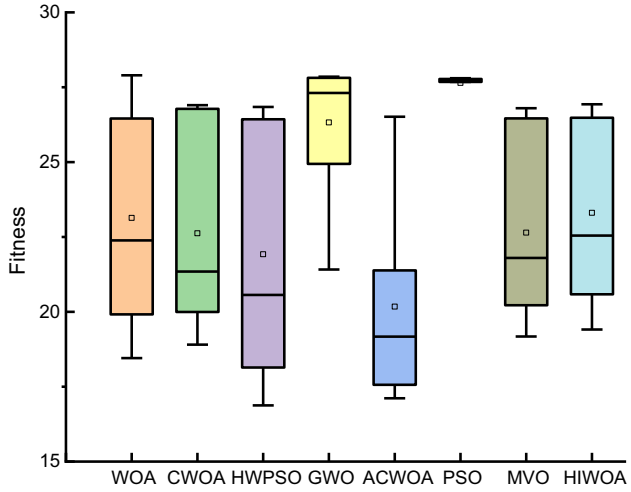

Fig. 6 Box diagram of control process model

multiple practical engineering fields to solve some of the complex environmental optimization problems.

Acknowledgements This work is supported by the National Natural Science Foundation of China under Grant U1813205, Independent Research Project of State Key Laboratory of Advanced Design and Manufacturing for Vehicle Body 71765003, and Hunan Key Laboratory of Intelligent Robot Technology in Electronic Manufacturing Open Foundation Grant 2017TP1011.

Funding Funding was provided by Innovative Research Group Project of the National Natural Science Foundation of China (U1813205).

References

Abd El Aziz M, Eweesc AA, Hassanien AE (2017) Whale optimization algorithm and moth-flame optimization for multilevel thresholding image segmentation. Expert Syst Appl 83:242–256

Abdel-Basset M, Manogaran G, El-Shahat D, Mirjalili S (2018) A hybrid whale optimization algorithm based on local search strategy for the permutation flow shop scheduling problem. Future Gener Comput Syst Int J Esci 85:129–145

Cai ZH, Lou J, Zhao J, Wu K, Liu NJ, Wang YX (2019) Quadrotor trajectory tracking and obstacle avoidance by chaotic grey wolf optimization-based active disturbance rejection control. Mech Syst Signal Process 128:636–654

Cao M, Huang MX, Xu RQ, Lu GN, Chen M (2019) A grey wolf optimizer-cellular automata integrated model for urban growth simulation and optimization. Trans GIS 23:672–687

Chen H, Xu Y, Wang M, Zhao X (2019) A balanced whale optimization algorithm for constrained engineering design problems. Appl Math Model 71:45–59

Coello CAC (2002) Theoretical and numerical constraint-handling techniques used with evolutionary algorithms: a survey of the state of the art. Comput Methods Appl Mech Eng 191:1245–1287

Dhar PL (2017) Introduction to optimum design. Ther Syst Des Simul 2017:385–407

Ding T, Chang L, Li CS, Feng C, Zhang N (2018d) A mixed-strategy-based whale optimization algorithm for parameter identification of hydraulic turbine governing systems with a delayed water hammer effect. Energies 11:2367

Dong XS, Dong WY, Call YL (2018) Ant colony optimisation for coloured travelling salesman problem by multi-task learning. IET Intel Transp Syst 12:774–782

Eberhart RC, Shi Y (2000) Ieee, and Ieee, Comparing inertia weights and constriction factors in particle swarm optimization. IEEE, New York

Fan SKS, Chiu YY (2007) A decreasing inertia weight particle swarm optimizer. Eng Optim 39:203–228

Gandomi AH, Yun GJ, Yang XS, Talatahari S (2013) Chaos-enhanced accelerated particle swarm optimization. Commun Nonlinear Sci Numer Simul 18:327–340

Gharehchopogh FS, Gholizadeh H (2019) A comprehensive survey: Whale optimization algorithm and its applications. Swarm Evol Comput 48:1–24

Gharehchopogh FS, Farnad B, Alizadeh A (2021a) A farmland fertility algorithm for solving constrained engineering problems. Concurr Comput Pract Exp 33:17

Gharehchopogh FS, Maleki I, Dizaji ZA (2021) Chaotic vortex search algorithm: metaheuristic algorithm for feature selection. Evolut Intell 2021(4)

Goldanloo MJ, Gharehchopogh FS (2021) A hybrid OBL-based firefly algorithm with symbiotic organisms search algorithm for solving continuous optimization problems. J Supercomput 2021:1–34

Gupta S, Deep K (2019) An efficient grey wolf optimizer with opposition-based learning and chaotic local search for integer and mixed-integer optimization problems. Arab J Sci Eng 44:7277–7296

He Q, Hu XT, Ren H, Zhang HQ (2015) A novel artificial fish swarm algorithm for solving large-scale reliability-redundancy application problem. ISA Trans 59:105–113

Karaboga D (2010) Artificial bee colony algorithm. Scholarpedia 5:6915

Karlekar NP, Gomathi N (2018) OW-SVM: Ontology and whale optimization-based support vector machine for privacy-preserved medical data classification in cloud. Int J Commun Syst 31:e3700

Kennedy J, Eberhart R (2002) Particle swarm optimization. In: Icnn95-international conference on neural networks.

Khadanga RK, Padhy S, Panda S, Kumar A (2018b) Design and analysis of multi-stage PID controller for frequency control in an islanded micro-grid using a novel hybrid whale optimization-pattern search algorithm. Int J Numer Model Electron Netw Devices Fields 31:e2349

Laskar NM, Guha K, Chatterjee I, Chanda S, Baishnab KL, Paul PK (2019) HWPSO: a new hybrid whale-particle swarm optimization algorithm and its application in electronic design optimization problems. Appl Intell 49:265–291

Li MS, Zhang HJ, Liu L, Chen BS, Guan LX, Wu Y (2018e) A quantitative structure-property relationship model based on chaos-enhanced accelerated particle swarm optimization algorithm and back propagation artificial neural network. Appl Sci Basel 8:1121

Luo J, Shi B (2018) A hybrid whale optimization algorithm based on modified differential evolution for global optimization problems. Appl Intell 49:1982–2000

Mafarja MM, Mirjalili S (2017) Hybrid Whale Optimization Algorithm with simulated annealing for feature selection. Neurocomputing 260:302–312

Mafarja M, Mirjalili S (2018) Whale optimization approaches for wrapper feature selection. Appl Soft Comput 62:441–453

Mehne HH, Mirjalili S (2018) A parallel numerical method for solving optimal control problems based on whale optimization algorithm. Knowl Based Syst 151:114–123

Mirjalili S, Lewis A (2016) The Whale optimization algorithm. Adv Eng Softw 95:51–67
Mirjalili S, Mirjalili SM, Lewis A (2014) Grey Wolf optimizer. Adv Eng Softw 69:46–61
Mirjalili S, Mirjalili SM, Hatamlou A (2015) Multi-verse optimizer: a nature-inspired algorithm for global optimization. Neural Comput Appl 27:495–513
Mohammadzadeh H, Gharehchopogh FS (2021) A multi-agent system based for solving high-dimensional optimization problems: a case study on email spam detection. Int J Commun Syst 34(3):e4670
Mohammadzadeh H, Gharehchopogh FS (2021) A novel hybrid whale optimization algorithm with flower pollination algorithm for feature selection: case study email spam detection. Comput Intell 37:176–209
Mohapatra P, Das KN, Roy S (2017) A modified competitive swarm optimizer for large scale optimization problems. Appl Soft Comput 59:340–362
Niknam T, Amiri B (2010) An efficient hybrid approach based on PSO, ACO and k-means for cluster analysis. Appl Soft Comput 10:183–197
Puchta EDP, Bassetto P, Biuk LH, Itaborahy MA, Converti A, Kaster MD et al (2021) Swarm-inspired algorithms to optimize a nonlinear gaussian adaptive PID controller. Energies 2021:14
Rahnema N, Gharehchopogh FS (2020) An improved artificial bee colony algorithm based on whale optimization algorithm for data clustering. Multimed Tools Appl 79:32169–32194
Rajabioun R (2011) Cuckoo optimization algorithm. Appl Soft Comput J 11:5508–5518
Rashedi E, Nezamabadi-pour H, Saryazdi S (2009) GSA: a gravitational search algorithm. Inf Sci 179:2232–2248
Sahu PR, Hota PK, Panda S (2018c) Modified whale optimization algorithm for fractional-order multi-input SSSC-based controller design. Optim Control Appl Methods 39:1802–1817
Sayed GI, Darwish A, Hassanien AE (2018) A New chaotic whale optimization algorithm for features selection. J Classif 35:300–344
Shao P, Yang L, Tan L, Li GQ, Peng H (2020) Enhancing artificial bee colony algorithm using refraction principle. Soft Comput 24:15291–15306
Singh RP, Dixit M, Silakari S (2015) Image contrast enhancement using GA and PSO: a survey. In: International conference on computational intelligence and communication networks
Sun YJ, Wang XL, Chen YH, Liu ZJ (2018) A modified whale optimization algorithm for large-scale global optimization problems. Expert Syst Appl 114:563–577
Sun WZ, Wang JS, Wei X (2018a) An improved whale optimization algorithm based on different searching paths and perceptual disturbance. Symmet Basel 10:210
Talbi H, Batouche M, and Ieee (2004) Hybrid particle swarm with differential evolution for multimodal image registration
Tang C, Sun W, Wu W, Xue M (2019) A hybrid improved whale optimization algorithm. In: IEEE 15th international conference on control and automation (ICCA), 2019, https://doi.org/10.1109/ICCA.2019.8900003.
Xue Y, Jiang JM, Zhao BP, Ma TH (2018) A self-adaptive artificial bee colony algorithm based on global best for global optimization. Soft Comput 22:2935–2952
Yan ZH, Sha JX, Liu B, Tian W, Lu JP (2018) An ameliorative whale optimization algorithm for multi-objective optimal allocation of water resources in Handan, China. Water 10:87
Yang CH, Yang HS, Chuang LY (2019) PBMDR: a particle swarm optimization-based multifactor dimensionality reduction for the detection of multilocus interactions. J Theor Biol 461:68–75
Zaman HRR, Gharehchopogh FS (2021) An improved particle swarm optimization with backtracking search optimization algorithm for solving continuous optimization problems. Eng Comput. https://doi.org/10.1007/s00366-021-01431-6
Zhang YD, Wang SH, Dong ZC, Phillip P, Ji GL, Yang JQ (2015) Pathological brain detection in magnetic resonance imaging scanning by wavelet entropy and hybridization of biogeography-based optimization and particle swarm optimization. Prog Electromag Res Pier 152:41–58
Zhang K, Huang Q, Zhang Y (2019) Enhancing comprehensive learning particle swarm optimization with local optima topology. Inf Sci 471:1–18