Complex & Intelligent Systems (2023) 9:767–795
https://doi.org/10.1007/s40747-022-00827-1

ORIGINAL ARTICLE

An enhanced whale optimization algorithm with improved dynamic opposite learning and adaptive inertia weight strategy

Di Cao[1] · Yunlang Xu[2] · Zhile Yang[3] · He Dong[1] · Xiaoping Li[1]

Received: 19 December 2021 / Accepted: 7 July 2022 / Published online: 1 August 2022

Abstract

Whale Optimization Algorithm (WOA), as a newly proposed swarm-based algorithm, has gradually become a popular approach for optimization problems in various engineering fields. However, WOA suffers from the poor balance of exploration and exploitation, and premature convergence. In this paper, a new enhanced WOA (EWOA), which adopts an improved dynamic opposite learning (IDOL) and an adaptive encircling prey stage, is proposed to overcome the problems. IDOL plays an important role in the initialization part and the algorithm iterative process of EWOA. By evaluating the optimal solution in the current population, IDOL can adaptively switch exploitation/exploration modes constructed by the DOL strategy and a modified search strategy, respectively. On the other hand, for the encircling prey stage of EWOA in the latter part of the iteration, an adaptive inertia weight strategy is introduced into this stage to adaptively adjust the prey's position to avoid falling into local optima. Numerical experiments, with unimodal, multimodal, hybrid and composition benchmarks, and three typical engineering problems are utilized to evaluate the performance of EWOA. The proposed EWOA also evaluates against canonical WOA, three sub-variants of EWOA, three other common algorithms, three advanced algorithms and four advanced variants of WOA. Results indicate that according to Wilcoxon rank sum test and Friedman test, EWOA has balanced exploration and exploitation ability in coping with global optimization, and it has obvious advantages when compared with other state-of-the-art algorithms.

Keywords Whale optimization · Inertia weight · Dynamic opposite learning · Global optimization

Introduction

As engineering optimization problems have become more complex, previous deterministic methods are less effective to deal with complex practical problems. The metaheuristic algorithms (MAs) inspired by natural phenomena are stochastic optimization strategies that do not require prior knowledge of the problem, which can be seen as a supplement to deterministic optimization techniques [1]. Different from deterministic methods iterating in one direction to the solution, MAs executes randomly and can find the approximate optimal solution of the problem through the parallel iterative optimization method of multiple search populations [2,3], and it has been applied in practical engineering in many different fields [4].

Early researches on MAs with neighborhood searches such as hill-climbing [5] and simulated annealing (SA) [6] were proposed. As optimization problems become more complex, research on MAs is also advancing with the times. Under such circumstances, evolution-basd MAs and swarm-based MAs have also been developed [7]. There are some classic MAs in the early days: genetic algorithms (GA) [8], particle swarm optimization (PSO) [9] ant colony algorithm (ACO) [10], etc. Then new advanced MAs have been proposed in recent years: brain storm optimization (BSO) [11], bat algorithm (BA) [12], ant-lion optimization (ALO) [13], artificial bee colony (ABC) [14], biogeography-based optimization (BBO) [15], teaching-learning-based optimization (TLBO) [16], gray wolf optimization (GWO) [17], sine cosine algorithm (SCA) [18] moth-ame optimization (MFO)

✉ Xiaoping Li
lixiaoping@hust.edu.cn

1 State Key Laboratory of Digital Manufacturing Equipment and Technology, Huazhong University of Science and Technology, Wuhan 430074, China

2 State Key Laboratory of ASIC and System, School of Microelectronics, Fudan University, Shanghai 200433, China

3 Shenzhen Institutes of Advanced Technology, Chinese Academy of Sciences, Shenzhen 518055, China

[19] and social group optimization (SGO) [20], etc. They have been widely used in many engineering applications, e.g., pattern recognition [21,22], production scheduling [23, 24], control tuning [25,26], time series forecasting [27,28]. Among these recently proposed swarm-based MAs, WOA is a competitive algorithm inspired by humpback whales' hunting behavior developed by Mirjalili and Lewis in [29]. The validation of the reference function experiments shows that WOA has better performance and fewer parameter settings against common algorithms [29], and it has been successfully used in flexible job shop scheduling [30], natural language processing [31] and optimal control [32], etc.

Although WOA can achieve better solutions than other algorithms in some problems, WOA still suffers from some shortcomings. Similar to most common MAs, it's a challenge for WOA to balance exploration and exploitation stages better [33]. WOA can split the exploration and exploitation stages which contributes to escape from local optima [29]. The balance mechanism completely relies on the internal parameter a which is only determined by the number of iterations without considering the current convergence situation [34]. It leads to a poor trade-off between exploration and exploitation. Besides, due to the encircling prey mechanism used in the exploitation stage, WOA can easily converge to a local optima [35] and not update the search agents in an effective way [36]. Once the current prey is a local optimal solution, WOA is easy to fall into premature convergence. In view of this, more attempts have been made to enhance WOA, and it is very popular to combine it with a variety of learning strategies such as Lévy flight, inertia weight and opposition-based learning [37]. Among these strategies, Lévy flight was widely applied in various fields for optimization [38]. Ling *et al.* [39] proposed Lévy flight trajectory-based WOA (LWOA) to avoid falling into the local optimal by adding stochastic process at the end of the classical WOA. Sun *et al.* [40] introduced a modified WOA (MWOA) utilizing Lévy flight's occasional long step-change feature to enhance the exploration ability when encircling preys. Yu *et al.* [41] exchanged the internal parameters of WOA with Lévy flight to tune the automatic carrier landing system controller. On the other hand, inertia weight is another learning strategy which can be flexibly embedded in each stage of WOA to improve the optimization effect and speed. Hu *et al.* [42] proposed an improved WOA based on inertia weight (IWOA), which introduced linearly decreasing inertia weight into the encircling prey stage to make IWOA convergence faster and more accurately. Chen *et al.* [43] proposed a random spare strategy and a double weighting strategy for the improvement of WOA (RDWOA), where two adaptive weights were used. Besides, the opposition-based learning (OBL) strategy is a competitive method for improving MAs which has been widely used for WOA. Since OBL was proposed by Tizhoosh [44], improvements to OBL have attracted more attentions. There are many existing variants of OBL, including quasi-opposition-based learning (QOBL) strategy [45], generalized opposition-based learning (GOBL) strategy [46], quasi-reflective based learning (QRBL) strategy [47] and elite opposition-based learning (EOBL)strategy [48], etc. All these are closer to the global optimum than OBL, and the improvement of WOA based on these variants is also worthy of research. HS Alamri *et al.* [49] proposed an opposition-based WOA (OWOA) to expand the initial population's search space better. Based on OBL and GOBL, Luo *et al.* [50] proposed an EOBL-based WOA to enhance the convergence rate of WOA. In terms of application, an improved WOA with EOBL (EOWOA) was established in [51] to solve the problem of complexity and poor accuracy for parameter estimation of Muskingum model. Chen *et al.* [52] presented a WOA with a chaos mechanism based on quasi-opposition (OBC-WOA), which can improve convergence speed and enhance the global search ability. Kumar and Chaparala [53] presented a QOBL-based chaotic WOA to cluster the nodes in the wireless sensor network. Although OBL can be used to raise exploitation of WOA, the disadvantage that the OBL search space is symmetric leads to its weak exploration capability.

For this issue, the DOL strategy is presented in [54], which has been successfully used for improving MAs [55–57]. The search space of DOL is asymmetric and can be adjusted dynamically, which indicates that DOL has enough diversity to enhance the exploration capability of WOA. Although DOL has been applied to improve some common meta-heuristic algorithms, it still suffers from the problem of poor modes balance which relies on fixed, pre-tuned internal weights. Hence, this paper proposes an enhanced WOA (EWOA) to improve the search ability of WOA by introducing an improved DOL (IDOL) and a modified encircling prey stage with adaptive inertia weight (AIW). Among them, IDOL is a newly proposed enhanced learning strategy based on DOL, which plays an important role in the initialization part and the algorithm iterative process of EWOA. IDOL has two modes: in mode-one, the DOL-based stage is used to enhance the exploitation ability; in mode-two, an Lévy flight trajectory-based searching for prey stage is adopted to enhance the exploration ability. In particular, for the trade-off between exploration and exploitation, the two modes of IDOL are switched by an adaption switching rule by evaluating the optimal solution in the current population. On the other hand, AIW adjusted by current agents' fitness is introduced into encircling pray stage to adaptive reform current prey position, which further enhances the search ability of EWOA's exploitation phase and avoids premature convergence.

The main contributions of this paper are as follows:

- An improved DOL (IDOL) strategy is proposed and embedded in EWOA to achieve a better trade-off between exploration and exploitation. IDOL has two modes and can be adaptively switched with concerning the current convergence situation, and it has been introduced into WOA for performance improvement.
- The adaptive inertia weight (AIW), which is adaptively adjusted according to the current agent fitness situation, has been introduced into the encircling prey stage to avoiding local optima when this stage is executed.
- 30 benchmark problems selected from standard 16 functions and CEC 2014 as well as three well-known engineering problems are utilized to evaluate the improvements for EWOA. Some WOA variants and other advanced MAs are also used for comparison to confirm the advantages of the proposed EWOA.

The remainder of this paper is arranged as bellows. "Preliminaries of WOA and DOL" Section describes the concepts of WOA and DOL strategy. The proposed enhanced WOA algorithm was introduced in "The presented EWOA" section in detail. In "Experiment results and discussion" section, EWOA and compared MAs are evaluated by benchmark functions, and then the analysis of experiment results are presented. Besides, in this section the actual engineering problems are also used for test. The main conclusion of this study is presented in "Conclusion" section.

Preliminaries of WOA and DOL

The structure of a canonical WOA

The WOA simulates the hunting behavior of whale bubbles which including three stages: encircling prey, bubble-net attacking and search for prey. Firstly, the whale gradually acquires relevant information about the prey by searching for the prey. Then, the whale keeps approaching the prey by surrounding the prey and spiraling close to the prey, and the final prey found is the optimal solution of the algorithm.

Encircling prey

In this stage, whales identify the position of their prey and surround them. Since the optimal position in the search space is not known in advance, the WOA algorithm assumes that the current optimal individual position is the target prey, and other individuals continue to update their positions by approaching the prey. So that the whales constantly close to the prey by Eqs. (1) and (2).

$$D = \left|C \cdot X^*(t) - X(t)\right| \tag{1}$$

$$X(t+1) = X^*(t) - A \cdot D \tag{2}$$

$$A = 2a \cdot r - a \tag{3}$$

$$C = 2 \cdot r \tag{4}$$

where X^* is the position vector of the optimum solution acquired so far, X is the position vector, t denotes the current iteration, A and C are coefficient vectors calculated as Eqs. (3) and (4), respectively, the components of a is linearly decreased from 2 to 0, and r is a random number in (0,1).

Bubble-net attacking method

In the bubble-net attacking phase, the algorithm simulates that when a whale finds its prey, it continuously spirals close to the prey with the prey as the center. In this way, the prey is approached slowly and unconsciously to achieve the purpose of catching prey. As the whale spirals around, it firstly calculates the distance between it and the prey, and then approaches the prey in a spiral way. The mathematical model is as follows.

$$X(t+1) = D' \cdot e^{bl} \cdot \cos(2\pi l) + X^*(t) \tag{5}$$

where $D' = |CX^*(t) - X(t)|$ denotes the distance of the i^{th} whale to the optimum solution acquired so far, b represents a constant defining the spiral shape, l is a random number in $[-1, 1]$. It is noting that when the whale moves around the prey in the outer circle, it will also reduce the enclosure radius.

Search for prey

The whales use the value A to control whether they are in the hunting stage or the encircling stage. When $|A| > 1$ at this time, the whale cannot obtain the prey's effective information, and it needs to make continuous attempts to find a trace of the prey's clues through random ways. Its mathematical model of modified hunting strategy is shown as Eqs. (6) and (7), where X^{rand} is a randomly selected whale position vector.

$$D = \left|C \cdot X^{rand}(t) - X(t)\right| \tag{6}$$

$$X(t+1) = X^{rand}(t) - A \cdot D \tag{7}$$

The mechanism of DOL

The idea of OBL is that the search space can be dynamically expanded from a candidate to its opposite candidate, which makes it easier to approach the optimal value. However, due to the fixed location definition, it will inevitably converge

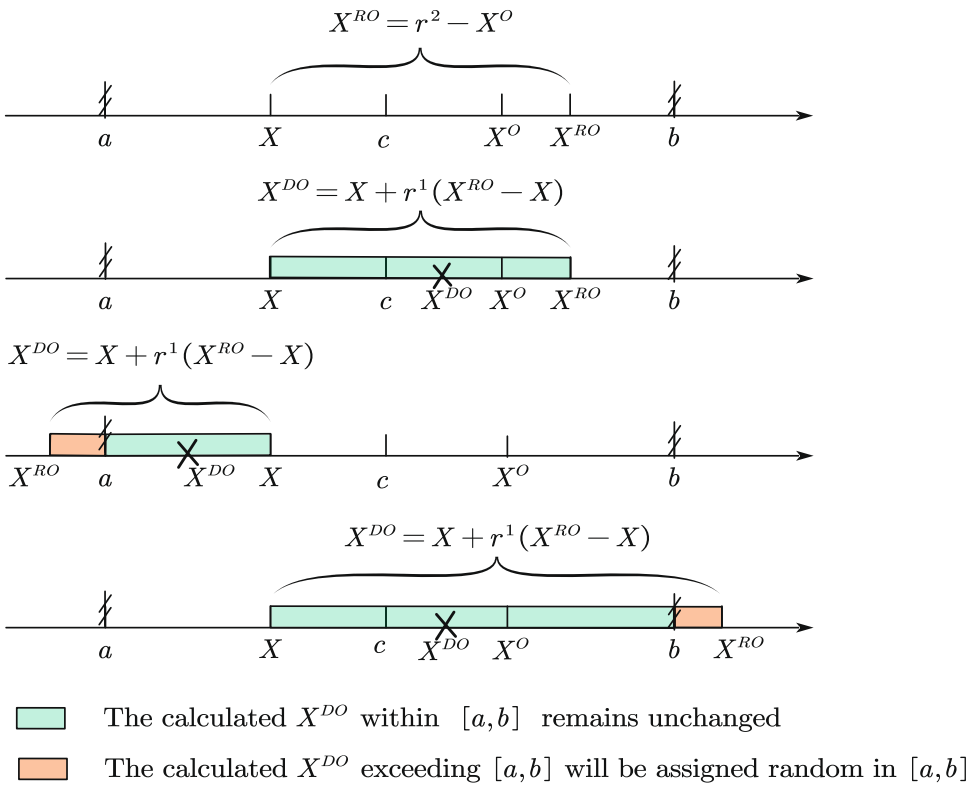


Fig. 1 The search range of X^{DO} under all possible relative positions of X^{RO} and the boundary $[a, b]$ [54]

to a local optimum. Given this deficiency, a DOL strategy is proposed [54]. The dynamic form of DOL strategy in one-dimensional space can be expressed as formulas (8) and Fig. 1.

$$\begin{aligned} X^O &= a + b - X \\ X^{RO} &= r^1 X^O \\ X^{DO} &= X + r^2 \left(X^{RO} - X \right) \end{aligned} \tag{8}$$

where X^O, X^{RO} and X^{DO} define the opposite number, random opposite number and the dynamic opposite number, respectively. r^1 and r^2 represent random values among (0,1). a denotes the low boundary and b denotes the high boundary while c is the center of [a,b]. The positions of all possible solutions of X^{RO} and X^{DO} relative to interval of [a,b] are illustrated in Fig. 1. If the calculation of X^{DO} by Eq. (8) exceeds the boundary, a new X^{DO} is reassigned a random number in [a, b]. By changing X^O to X^{RO}, the search space becomes asymmetric and is conducive to improving exploration capabilities.

The presented EWOA

This section mainly focuses on the improvement of WOA. Firstly, a novel improved dynamic learning strategy basing DOL (IDOL) is proposed. Its exploration-exploitation modes consist of modified search strategy and DOL strategy, while adaptive mode switching rules are adopted to balance two modes. IDOL is executed in two stages of EWOA, including population initialization at the beginning of evolution and generation jumping. The AIW is then introduced to the encircling prey stage of EWOA, which enhances the exploration ability to avoid falling into a local optimum in the second half of the iteration. Two independent improvements for canonical WOA are named IDOLWOA and AIWWOA, respectively.

IDOL strategy

It can be seen from the mechanism of DOL that as the population converges iteratively, the search space gradually shrinks. To solve this problem, DOL introduces a preset fixed weight w_d shown in Eq. (9), so that it can not only maintain the original convergence of OBL, but also improve the exploration ability. When w_d is set larger, the difference between X^{DO} and X also becomes larger and the diversity of populations increases accordingly. Nevertheless, the increase of w_d will lead to low convergence speed and weak exploitation capability.

$$X^{DO} = X + r^2 w_d \left(X^{RO} - X \right) \tag{9}$$

Different from DOL, IDOL can adaptively switch between the two modes to solve the above problems, instead of adopting a single mode based on a fixed pre-set weight w_d in DOL. Mode-one is utilized to improve the exploitation and mode-two is used to raise exploration of IDOL, respectively. The exploration-exploitation trade-off of IDOL is achieved by adopting an adaptive switching rule, which can be adjusted according to the current fitness situation.

Mode-one:DOL

IDOL improved the exploitation capability of WOA based on DOL by expanding the search space of group, where the weight of DOL is set to be 1. Hence, X^{IDOL} is defined as Eq. (10).

$$X^{IDOL} = X + r^3\left(r^4 X^O - X\right) \tag{10}$$

where X^O is obtained by Eq. (8) and r^3, r^4 are random numers among (0,1).

Mode-two:modified search strategy

This mode is inspired by the WOA searching for prey stage, which uses the information interaction between random individuals to expand the search range. To further enhance the mutation capability of this mode, the Lévy flight operator has also been introduced into this mode. Lévy distribution random number is widely used to improve MAs due to its occasional long-step mutation [58] . The simple power-law vision of Lévy flight is shown in (11).

$$L(s) \sim |s|^{-1-\beta}, 0 < \beta \leq 2 \tag{11}$$

where $\beta = 1.5$ and s is a random step size, which can be obtained approximately using Mantegna's method [59] to generate random numbers with a step length similar to the Lévy distribution by Eq. (12). Both $\mu = N\left(0, \sigma_\mu^2\right)$ and $\nu = N\left(0, \sigma_\nu^2\right)$ obey normal stochastic distributions, where the value of σ_u and σ_v can be calculated as Eq. (13).

$$s = \frac{\mu}{|\nu|^{1/\beta}} \tag{12}$$

$$\sigma_\mu = \left[\frac{\Gamma(1+\beta) \times \sin(\pi \times \beta/2)}{\Gamma(1+\beta/2) \times \beta \times 2^{(\beta-1)/2}}\right]^{1/\beta}$$
$$\sigma_\nu = 1 \tag{13}$$

Based on Eq. (7), the original operator A is replaced by the Lévy flight operator s with higher mutation performance to form equation (14).

$$X^{IDOL} = X^r - r^5 s\left(X^{rand} - X\right) \tag{14}$$

where r^5 is a random number among (0,1).

Adaptive IDOL mode switching rules

Since the algorithm will stagnate with iterations, it is necessary to switch modes to improve the search ability when stagnation. The IDOL mode switching rule plays an important role, which is formed by the iterative convergence parameter J and the mode switching threshold T. J and T are initialized to 0 at the beginning of operation, and then with iteration, whenever a better solution cannot be obtained after a round of iteration, J is accumulated following $J = J + 1$; when obtain better fitness, the iteration convergence parameter J is reset to 0; and when J exceeds T, the mode will be switched with J being reset to 0. Furthermore, the convergence speed of the algorithm will gradually decrease with the number of iterations increasing. Therefore, the threshold for switching modes should be increased to provide more iteration times for the current mode to prevent frequent switching when the iteration enters the later stage. In view of this, whenever switching modes, $T = T + \Delta T$. Combining prior knowledge and experiment, ΔT is set to 5. Under this parameter setting, IDOL-based WOA can obtain better performance.

AIW-based adaptive encircling prey stage

In canonical WOA's encircling the prey stage, all agents approach the optimal agent at a random distance based on the A obtained by Eq. (3), and this value decreases linearly with iteration. However, due to the gradual decrease of the step size, the agent's exploration performance will deteriorate at this stage. For increasing the diversity of the group and avoiding falling into local optima, this paper introduces the AIW into encircling stage. Multiply the AIW with the target prey of the current agent to form a new target, and adjust the AIW according to the fitness of the agents. The improved encircling prey stage is described as Eq. (15):

$$X_i(t+1) = w_i(t) \cdot X^*(t) - A \cdot \left|C \cdot X^*(t) - X_i(t)\right| \tag{15}$$

Among them, the adjustment of convergence is determined by the AIW operator $w_i(t)$, $w_i(t) \in [0, 1]$ which can be adjusted adaptively according to the current agent position. When the agent position in the solution space is more suitable, the AIW is maintained or increased. Specifically, when the agent's fitness is better than the mean value of the whole population, the weight operator $w_i(t)$ of this agent is

adjusted to the maximum value, so that they can converge to the current prey as soon as possible. On the other hand, $w_i(t)$ of the current agent will be assigned smaller value when its fitness is worse than average. The worse the performance is, the smaller the weight value is, which will weaken the prey's impact on this group' position update. Once the current prey is a local optima, then the search agent far away from the prey has a greater chance of escape from that to find global optima. Based on that, AIW strategy helps the group to have greater search ability and overcome premature convergence. At this point, the specific solution of AIW is determined following a mapping function basing the position proportion of the current agent's fitness. There is a modified Versoria function [60] that can be used as a mapping function between fitness and AIW. Its curve is nonlinear while its computational complexity is lower than similarly shaped sigmoid functions and other nonlinear mapping functions. It has been successfully applied to adaptive inertial weight particle swarm optimization (AWPSO) [61,62]. Based on the modified Versoria mapping function, $w(t)$ can be formulated as Eq. (16).

$$w_i(t) = \begin{cases} 1 - \frac{1}{(\varphi \cdot (a_i(t) - 1/2)^2 + 2)} & \text{if} \quad a_i(t) \le 0.5 \\ \frac{1}{(\varphi \cdot (a_i(t) - 1/2)^2 + 2)} & \text{otherwise} \end{cases} \tag{16}$$

where $a_i(t), a_i(t) \in [0, 1]$ represents the current agent's ranking in all agents between the minimum fitness and the average fitness, which is described as Eq. (17), and φ is a regularization coefficient set to 300 in subsequent experiments.

$$\begin{aligned} f_{\text{ave}}(t) &= \sum_{i=1}^{N_p} f_i(t)/N_p \\ f_{\min}(t) &= \min\left\{f_1(t), f_2(t), ..., f_{Np}(t)\right\} \\ a_i(t) &= \left(f_i(t) - f_{\min}(t)\right) / \left(f_{\text{ave}}(t) - f_{\min}(t)\right) \end{aligned} \tag{17}$$

where $f_i(t)$ means the fitness of the i agent at the t iteration; $f_{\text{ave}}(t)$ and $f_{\min}(t)$ denote the average value and minimum value of all Np size populations' fitness.

The formation of EWOA

By embedding the IDOL strategy into AIWWOA, EWOA is formulated. In EWOA, the IDOL is embedded into the both population initialization and generation jumping processes. By comparing the fitness of the group generated by IDOL and the original group, the best group can be selected through the greedy rule.

IDOL initialization

In the initial generation, the population is generated randomly. To obtain a better initial population to speed up the convergence rate, the IDOL mode-one, which is inspired by the DOL's dynamic asymmetric reconstruction space, is utilized after the random initialization of $X_{i,j}$ described as Eq. (18).

$$X_{i,j}^{IDOL} = X_{i,j} + r_i^3 \left(r_i^4 X_{i,j}^O - X_{i,j}\right) \tag{18}$$

where $i = 1, 2, \ldots, Np$ is the population size, $j = 1, 2, \ldots, D$ is the dimension of agents, r^1 and r^2 are random numbers in (0,1), and $X_{i,j}^O$ is the opposite position of $X_{i,j}$ inside the search space. After the IDOL initialization process, the updated agent may exceed the boundary, so it is necessary to randomly generate a position for these individuals within the interval.

$$X_{i,j}^{IDOL} = \text{rand}\left(a_j, b_j\right), \text{ if } X_{i,j}^{IDOL} < a_j \| X_{i,j}^{IDOL} > b_j \tag{19}$$

where a_j and b_j are boundaries of search space. After the random initialization and IDOL initialization, Np fittest agents are selected from $X \bigcup X^{IDOL}$.

Improvement by AIW

The WOA is improved by the AIW, which utilizes Eq. (15) to update the positions of agents in the encircling stage. This equation includes the key factor: $w_i(t)$, introduced in Eq. (16).

IDOL generation jumping

By applying the adaptive learning strategy to the WOA algorithm generation, the appropriate IDOL mode is switched according to the current optimal solution's update situation. This additional generation operation may lead to more accurate convergence or selection mutation to further increase population diversity, which enable the algorithm to find a better solution.

In each iteration, the population can be updated through IDOL. The convergence state parameter (J) is used to determine which mode IDOL executes. The IDOL is used to generate the jumping process while optimizing the position of the population, which is described as Eq. (20).

$$X_{i,j}^{IDOL} = \begin{cases} X_{i,j} + r_i^3 \left(r_i^4 \left(a_j + b_j - X_{i,j} \right) - X_{i,j} \right) & \text{if mode} > 0 \\ X_{i,j}^{rand} - r_i^5 s_i \left(X_{i,j}^{rand} - X_{i,j} \right) & \text{otherwise} \end{cases} \tag{20}$$

where $X_{i,j}^{rand}$ is a randomly selected individual at the current generation ($i = 1, 2, \ldots, Np$, $j = 1, 2, \ldots, D$), $[a_j, b_j]$ is the search space of j_{th} dimension, r_i^3, r_i^4, r_i^5 denote random numbers among (0,1), and s_i is a random step length vector generated by Eq. (12). Randomly select a mode at the beginning Eq. (21) firstly, switch mode by mode $= -$mode if $J > T$ when convergence suffers from stagnation and reaches the threshold.

$$\text{mode} = \text{sign}(\text{rand}(0, 1) - 0.5) \tag{21}$$

After the IDOL process, for improving the performance of IDOL, search space's boundaries are updated as Eq. (22):

$$\begin{aligned} a_j &= \min \left\{ X_{1,j}, X_{2,j}, ..., X_{Np,j} \right\} \\ b_j &= \max \left\{ X_{1,j}, X_{2,j}, ..., X_{Np,j} \right\} \end{aligned} \tag{22}$$

Finally, Np better individuals will be selected from $X \bigcup X^{IDOL}$. The IDOL genration jumping process is described in Algorithm 2.

Algorithm procedure

The pseudo-code of EWOA is shown as Algorithm 1, which is detailed described as follows:

(1) Initialization basing IDOL: In this step, EWOA generates a dynamic opposition group of the original group according to the formula of DOL as Eq. 18, and all agents are evaluated together to obtain the optimal Np ones.
(2) Update position with AIW-based WOA: In this step, EWOA updates the positions of each agent using either Eqs. 5, 6, or 15. If $p > 0.5$, agents updating position as Eq. 5, otherwise one of the searching for prey stage and encircling the prey stage will be implemented according to $|A|$. Specifically, If $|A| < 1$, the encircling prey period adopts the adaptive inertia weight strategy as Eq. 15. Otherwise, Eq. 6 will be used to update its position;
(3) Apply the IDOL generation jumping stage as Algorithm 2: In this step, two modes of IDOL are introduced to enhance the WOA, it contains exploration and exploitation modes with an adaptive switching rule. Details of Algorithm 2 are shown as follows.
(4) EWOA execution termination: The proposed EWOA algorithm repeats steps 2 and 3 maximal evolutionary iteration times or can be ended when obtaining the theoretically best solution.

IDOL generation jumping is a key stage of Algorithm 1, and it is shown in Algorithm 2. The details are as follows:

(1) Update position following current IDOL mode: In this step, two IDOL modes are supplied to update positions as Eq.(20). They can both enhance exploration and exploitation ability of EWOA. When convergence suffers stagnation, proper mode will be conducted by judging the parameters of switching rule.
(2) Fitness assessment: Best Np number search agents can be selected after checking boundaries and evaluating the $2Np$ size group consists of IDOL generation and original group.
(3) Update IDOL parameters and modes: Modes can be switched adaptively referring the convergence situation by updating and judging the parameters of IDOL after each iteration. Parameter J records the times of stagnation during operating current mode, as it sums one if current mode cannot obtain better solution. T denotes the threshold of modes switching rule. Modes will not be switched until $J > T$. Ones mode switched, T is added ΔT and J is reset to 0.

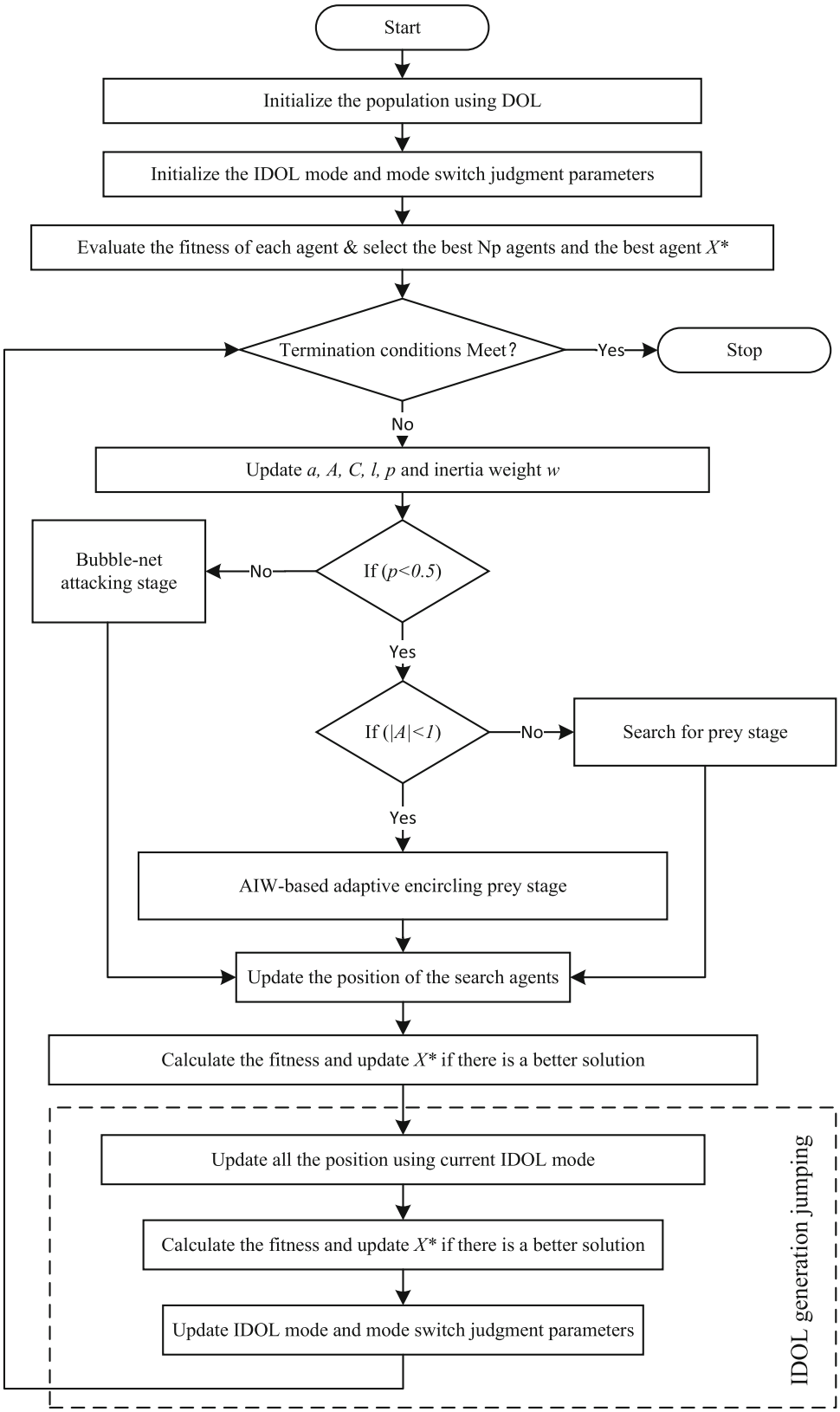
The workflow of the proposed EWOA can be summarized as follows and depicted in Fig. 2.

(1) Initial the population using DOL and set parameters of IDOL generation jumping preparing to start iterating.
(2) In each iteration, parameters a, A, C, l, p and w update for each search agent. And all the agents' position are changed through one of the stages including Bubble-net attacking stage Search for prey stage and AIW-based adaptive encircling prey stage basing these parameters. Calculate fitness and update optimal agent X^*.
(3) The IDOL generation jump stage adopts two modes and switching rules to be carried out after second step. Np number of optimal individuals are obtained through this stage and best agent X^* is updated. The modes and parameters J and T will be updated according to the switching rules.
(4) Repeat second and third steps until the theoretical optimal value is found or the maximum number of iterations is reached.

Experiment results and discussion

In this section, the performance of EWOA are verified by experiments on 30 benchmark functions and three engineering problems. The effective variants of WOA and many current advanced MAs are added to the comparison, and the accuracy and convergence of the results are analyzed afterward.

Fig. 2 EWOA flowchart



Algorithm 1 The EWOA Algorithm

Randomly generate an initial population $X_i (i = 1, 2, ..., Np)$
for $i = 1; i \leq Np; i++$ **do**
  $r_i^3 = \text{rand}(0, 1)$, $r_i^4 = \text{rand}(0, 1)$
  for $j = 1; i \leq D; j++$ **do**
    $X_{i,j}^{IDOL} = X_{i,j} + r_i^3 \cdot (r_i^4 \cdot (a_j + b_j - X_{i,j}) - X_{i,j})$
    Check the boundaries
  end for
end for
Select Np number of the fittest individuals from $\{X \bigcup X^{IDOL}\}$
Set IDOL parameters $J, T = 0$
Initialize mode of IDOL using Eq. (21)
Set the generation $t = 0$
while $t <$ Maximal evolutionary iteration **do**
  for $i = 1; i \leq Np; i++$ **do**
    Update a,A,C,l,P,w
    for $j = 1; i \leq D; j++$ **do**
      if $P \leq 0.5$ **then**
        if $|A| \leq 1$ **then**
          Implement the encircling prey using Eq. (15)
        else
          Implement the search for prey using Eq. (6);
        end if
      else
        Implement the bubble-net attacking using Eq. (5);
      end if
    end for
    Check the boundaries
    $t++$
  end for
  Implement the IDOL generation jumping algorithm as Algorithm 2
  $t++$
end while

Algorithm 2 The IDOL Generation Jumping Algorithm

for $i = 1; i \leq Np; i++$ **do**
  $r_i^3 = \text{rand}(0, 1)$, $r_i^4 = \text{rand}(0, 1)$,$r_i^5 = \text{rand}(0, 1)$
  for $j = 1; i \leq D; j++$ **do**
    Update IDOL generation jumping stage using Eq. (20)
    Check the boundaries
  end for
end for
Select Np number of the fittest individuals from $\{X \bigcup X^{IDOL}\}$
Update X^* if there is a better solution
if $X^*(t) < X^*(t-1)$ **then**
  $J = 0$
else
  $J = J + 1$
end if
if $J > T$ **then**
  $T = T + \Delta T$
  $J = 0$
  mode=-mode
end if

Test functions

The functions F1–F16 and F17–F30 used as benchmark functions in the optimization literature [63–66] and CEC 2014 [67] are divided into five categories: unimodal functions (F1–F6), multimodal functions (F7–F11), fixed-dimensional multimodal functions (F12–F16), hybrid functions (F17–F22) and composition functions (F23–F30). Unimodal functions are commonly used to evaluate the exploitation capabilities of optimization algorithms while the exploration capabilities of algorithms can be evaluated through multimodal and fixed-dimensional multimodal functions. Both hybrid functions and composition functions belong to multimodal functions, which have higher complexity after conversion and hybridization with other meta-functions. When solving complex problems, they can test the comprehensive capabilities of the algorithm for exploration, exploitation and the trade-off between them. The specific description of 30 functions is shown in Appendix 6 in the Table 12, where "HF" and "CF" represent hybrid functions and composition functions from CEC 2014, respectively, and "N" represents the number of basic functions that make up HF or CF. Besides, three engineering problems are used for evaluating EWOA's ability to solve practical problems.

Parameter settings

To evaluate the performance of EWOA, numerical experiments on functions shown in Table 12 are performed. The three sub-variants of the proposed EWOA introduced in the Table 1, namely AIWWOA, DOLWOA and IDOLWOA, are added to the test. Then three classic MAs including TLBO, MFO and GWO, and three advanced MAs including the opposite learning artificial bee colony (OABC) [68], weighted PSO (wPSO) [69] and elite TLBO (ETLBO) [70], are compared with EWOA. Besides, there are four variants of WOA, including OWOA, IWOA, LWOA and RDWOA are also involved in comparison. These compared advanced MAs include the two improvement strategies that are also involved in EWOA, the opposition operator or inertia weight, to further prove that we can introduce these strategies into EWOA with the best effect.

The parameters of these compared MAs and numerical tests' settings are as follows. For IDOLWOA and EWOA, the threshold increment step size ΔT is set to 5; for DOLWOA, the DOL weight w_{dol} has been pre-tuned and set to the optimal value 12 ; for IWOA, varying step is 0.1 from 0 to 1; for RDWOA weight adjustment factor s is set to 0; for ETLBO, the number of elites is set to 5; for wPSO, cognitive constant $C1 = 2$, social constant $C2 = 24$, inertia constant w decrease from 0.9 to 0.4; for OABC jump rate J_r are set to 0.5; The rest algorithms' parameter settings are subject to the original reference.

A large number of experiments can more accurately quantify the algorithm's optimization ability in different types of problems and evaluate the comprehensive performance of the algorithms. In detail, for F1–F16 tests, set the population size and number of iterations to 30 and 2000, respectively; for F17–F30 (CEC 2014) tests and CEC 2019 tests, set the

Table 1 Comparison of core operations in WOA, sub-variants of EWOA and EWOA

	WOA	DOLWOA	AIWWOA	IDOLWOA	EWOA
Population initialization	Random number	DOL	Random number	DOL	DOL
Basic exploitation search	Encircling prey	Encircling prey	AIW-encircling prey	Encircling prey	AIW-encircling prey
Additional operations	None	DOL	None	IDOL	IDOL

Table 2 Wilcoxon rank-sum tests between EWOA and other WOA variants

	EWOA	AIWWOA	IDOLWOA	DOLWOA	WOA
F1	**0.00E+00**	**0.00E+00**	**0.00E+00**	3.26E–260	9.75E–307
F2	**0.00E+00**	**0.00E+00**	2.82E–225	1.12E–143	3.04E–210
F3	**0.00E+00**	**0.00E+00**	3.09E–248	9.64E–68	2.06E+03
F4	**0.00E+00**	**0.00E+00**	7.33E–183	9.85E–98	4.85E–07
F5	2.47E+01	2.64E+01	**2.44E+01**	2.55E+01	2.64E+01
F6	5.38E–05	**2.60E–05**	5.83E–05	8.15E–04	3.18E–05
F7	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**
F8	**8.88E–16**	**8.88E–16**	**8.88E–16**	**8.88E–16**	**8.88E–16**
F9	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**
F10	**5.98E–05**	6.02E–04	1.30E–03	8.52E–04	1.20E–03
F11	2.85E–02	1.82E–02	8.72E–02	8.40E–02	**1.77E–02**
F12	3.08E–04	4.25E–04	3.16E–04	**3.08E–04**	6.02E–04
F13	**−3.86E+00**	**−3.86E+00**	**−3.86E+00**	**−3.86E+00**	**−3.86E+00**
F14	**−3.30E+00**	−3.25E+00	−3.29E+00	−3.29E+00	−3.26E+00
F15	**−1.02E+01**	−8.29E+00	−8.62E+00	−9.81E+00	−8.78E+00
F16	**−1.05E+01**	−7.93E+00	−9.82E+00	**−1.05E+01**	−8.54E+00
F17	2.54E+06	3.51E+06	**2.23E+06**	2.34E+06	4.75E+06
F18	**2.41E+03**	1.70E+04	3.42E+03	4.73E+03	1.19E+04
F19	7.99E+01	8.62E+01	**5.66E+01**	5.94E+01	8.20E+01
F20	5.85E+03	2.06E+04	7.74E+03	**4.93E+03**	1.89E+04
F21	**4.06E+05**	1.23E+06	4.42E+05	5.93E+05	2.14E+06
F22	**6.03E+02**	7.30E+02	6.85E+02	6.42E+02	6.87E+02
F23	**2.00E+02**	**2.00E+02**	**2.00E+02**	**2.00E+02**	3.58E+02
F24	**2.00E+02**	**2.00E+02**	**2.00E+02**	**2.00E+02**	2.05E+02
F25	**2.00E+02**	**2.00E+02**	**2.00E+02**	**2.00E+02**	2.23E+02
F26	1.77E+02	1.34E+02	1.77E+02	1.24E+02	**1.04E+02**
F27	**2.00E+02**	**2.00E+02**	**2.00E+02**	**2.00E+02**	6.54E+02
F28	**2.00E+02**	**2.00E+02**	**2.00E+02**	**2.00E+02**	3.84E+03
F29	**2.00E+02**	**2.00E+02**	**2.00E+02**	4.11E+04	1.97E+07
F30	**2.00E+02**	**2.00E+02**	7.51E+04	1.66E+05	2.55E+05
+/−/=	N/A	12/1/17	9/3/18	16/2/12	22/1/7
ARV	**1.50**	2.60	2.27	2.63	3.83

The best one of the mean results of all algorithms run on the benchmark function (in bold)

population size and number of iterations to 100 and 3000, respectively. And we performed 30 individual demonstrations of all experiments to establish a statistical sample of the algorithm. Besides, in the subsequent data display, the result of the CEC 2014 test is the final value of the evolution minus the theoretically optimal solution.

Accuracy analysis

In this section, the performance of EWOA is verified by comparing with WOA variants and other algorithms over all the benchmark functions. Table 2 lists the effect comparison between the improved optimized variants proposed in

this paper, and Tables S1, S2 and S3 provided in the supplementary file depict the optimization effects of unimodal and multimodal functions with variety of different dimensions. Tables 13, 14, 15, 16 and 17 in Appendix 7 represent the statistic results of 11 compared algorithms with EWOA on all 30 functions. In the presented data, "Mean" and "Std" represent the mean and standard deviation of the results of 30 independent runs, and "Ranking" is the average ranking of "Mean". Non-parametric tests for the performance of multiple algorithms, including the two-tailed Wilcoxon rank-sum tests [71] with a significance level of 0.05 and the Friedman test [72] with the Holm procedure, are shown in Tables 3, 4, 5 and 6.

To demonstrate the effect of independent improvement on WOA, i.e., IDOL and AIW, and the effect of improvement from DOL to IDOL, the performance comparison results of all the variants proposed in this paper, i.e., WOA, DOLWOA, AIWWOA, IDOLWOA and EWOA, are presented in Table 2. It lists the mean best fitness and the corresponding ARV (average ranking value) over all 30 functions (F1–F30) obtained by EWOA and other algorithms. The three columns below the algorithm names in Table 2 are the average optimal fitness of the algorithm, the statistical results of the Wilcoxon detection between EWOA and 4 algorithms, and the ranking based on the statistical results, respectively. The symbols "+" and "−" respectively indicate that EWOA is significantly better and worse than other algorithms through the Wilcoxon rank-sum tests; "=" indicates that there is no significant difference between them. Among them, EWOA is the optimal algorithm, IDOLWOA is better than DOLWOA, and all the improved algorithms are better than the canonical WOA. In detail, it can be observed AIWWOA perform better than WOA in solving complex problems from the test results of composition functions benefit from its more exploratory mechanism to encircle prey. Compared with DOLWOA, IDOLWOA can maintain outstanding exploitation ability in handling unimodal function, while avoiding local optima when calculating complex functions. The combination of IDOL and AIW in EWOA have all the above advantages.

In order to study the optimization performance of EWOA in various dimensional problems, a multi-dimensional test was performed. The test included 6 unimodal functions and 5 multimodal functions, and compares with the other 11 functions in the mean and standard value of optimization results. In the subsequent tests in this paper, the default 30 dimensions are used. In this part of the different dimensional tests, the dimensions are set to 10, 50 and 100, and they are executed independently 30 times. From Tables S1, S2 and S3 in the supplementary material, it can be observed that EWOA can have strong competitiveness in both low-dimension and high-dimension tests, and rank first in the number of best solutions. It not only rank high among WOA variants, but also performs better than other optimization algorithms.

The results of experiments on unimodal, multimodal and fixed dimension functions are depicted in Tables 13, 14 and 15 given in Appendix 7. EWOA behaves best on unimodal functions F1–F4, multimodal functions F7–F9, second or third on F5, F6, F10, F11, and ranked first in the test of two functions, F15 and F16, while F13 and F14 ranked second and third. The hybrid and composition functions F17–F30 of CEC 2014 are more complex than unimodal and multimodal functions and are closer to optimization problems in the real world. For solving these problems, algorithms should appropriately balance exploration and exploitation. According to the results in Tables 16 and 17 given in Appendix 7, EWOA ranks first in terms of F18, F23–F25 and F27-30. Obviously, it performs best in the optimization problem of composition functions.

The overall significance analysis results on functions F1–F30 are also presented in this section. The optimization results of all algorithms on the test function are divided into two groups for comparison, the comparison with the WOA variant and the comparison with other advanced algorithms. To detect the significance of differences between algorithms, Wilcoxon rank-sum tests at the 5% level were conducted. Detailed data of the Wilcoxon rank-sum test results are shown in the Tables 3 and 4. where the values of p and h represent the difference significance indicators of EWOA compared with other algorithms. Specifically, if p is less than 5%, h will set to 1, which represents significance of the difference; otherwise h is set to 0, which means there is no significant difference between them. Statistical results of Wilcoxon rank-sum tests are shown in Table 5 When the difference in comparison is significant, the signs "Better" and "Worse" represent the times that EWOA performs better or worse on all functions, respectively.The sign "Same" means there is no significant difference between EWOA and compared algorithms. Besides, "Difference" denotes the value of "Better" minus "Worse". Learn from the Wilcoxon test results, the value of "Better" is larger than 15 when compared to all compared algorithms except ETLBO, and it can be observed that the "Dierence" value is not less than 10 when compared with others. Moreover, the average excellent rate of EWOA can reach 66.7% $\left(\sum_{i=1}^{11} \text{Better}_i/(30 \times 11)\right)$. In order to facilitate the visual comparison of the ranking of the algorithms, the two groups of ranking comparison are graphically depicted in Fig. 3. According to the Friedman test results in the fourth column of Table 6, EWOA has average rankings of 3.4062, 4.3214 and 3.85 on Simple (F1–F16) problems, complex (F17–F30) problems and overall 30 problems. The rankings of EWOA is smaller than all compared algorithms, and most of the null hypotheses are rejected based on Holm's

Table 3 The results of Wilcoxon rank-sum tests for EWOA with WOA variants

Function		EWOA versus				
		WOA	OWOA	IWOA	LWOA	RDWOA
F1	*p*	1.21E+112	1.21E+112	NaN	NaN	1.21E+112
	h	1	1	0	0	1
F2	*p*	1.21E+112	1.21E+112	1.13E+112	NaN	1.21E+112
	h	1	1	1	0	1
F3	*p*	1.21E+112	1.21E+112	NaN	NaN	1.21E+112
	h	1	1	0	0	1
F4	*p*	1.21E+112	1.21E+112	1.18E+112	NaN	1.21E+112
	h	1	1	1	0	1
F5	*p*	3.20E+109	5.60E+107	3.02E+111	9.92E+111	1.20E+108
	h	1	1	1	1	1
F6	*p*	8.50E–02	6.07E+111	1.70E–02	7.29E–03	2.25E–04
	h	0	1	1	1	1
F7	*p*	NaN	NaN	NaN	NaN	NaN
	h	0	0	0	0	0
F8	*p*	NaN	NaN	NaN	NaN	NaN
	h	0	0	0	0	0
F9	*p*	NaN	NaN	NaN	NaN	NaN
	h	0	0	0	0	0
F10	*p*	3.82E+110	3.50E+109	3.69E+111	3.69E+111	2.38E+107
	h	1	1	1	1	1
F11	*p*	1.91E–01	7.70E–04	7.39E+111	1.70E+108	8.20E+107
	h	0	1	1	1	1
F12	*p*	1.46E+110	3.33E–01	3.02E+111	3.02E+111	3.26E–01
	h	1	0	1	1	0
F13	*p*	3.34E+111	5.46E+109	3.02E+111	3.02E+111	7.38E–02
	h	1	1	1	1	0
F14	*p*	3.08E+108	1.25E+107	1.33E+110	2.02E+108	5.32E–03
	h	1	1	1	1	1
F15	*p*	3.02E+111	6.52E+109	3.02E+111	3.02E+111	1.42E+106
	h	1	1	1	1	1
F16	*p*	3.02E+111	1.60E+107	3.02E+111	3.02E+111	9.31E+106
	h	1	1	1	1	1
F17	*p*	1.86E–03	9.07E–03	1.44E–03	2.44E+109	5.75E–02
	h	1	1	1	1	0
F18	*p*	2.96E–05	1.03E+106	4.50E+111	3.02E+111	1.86E+109
	h	1	1	1	1	1
F19	*p*	9.71E–01	3.95E–01	9.23E–01	5.87E–04	5.46E+106
	h	0	0	0	1	1
F20	*p*	1.43E+108	1.19E+106	3.69E+111	5.49E+111	1.12E–01
	h	1	1	1	1	0
F21	*p*	1.58E–04	4.21E–02	1.87E–05	1.21E+110	6.97E–03
	h	1	1	1	1	1
F22	*p*	1.96E–01	3.18E–01	9.51E+106	1.00E–03	8.07E–01
	h	0	0	1	1	0
F23	*p*	1.21E+112	1.21E+112	NaN	NaN	NaN
	h	1	1	0	0	0

Table 3 continued

Function		EWOA versus				
		WOA	OWOA	IWOA	LWOA	RDWOA
F24	*p*	1.21E+112	1.21E+112	NaN	NaN	1.93E+110
	h	1	1	0	0	1
F25	*p*	1.93E+110	5.85E+109	NaN	NaN	NaN
	h	1	1	0	0	0
F26	*p*	1.76E+109	1.28E+110	8.82E–03	2.06E–03	3.87E+107
	h	1	1	1	1	1
F27	*p*	1.21E+112	1.21E+112	NaN	NaN	1.10E–02
	h	1	1	0	0	1
F28	*p*	1.21E+112	1.21E+112	NaN	NaN	4.19E–02
	h	1	1	0	0	1
F29	*p*	1.21E+112	1.21E+112	NaN	8.75E+107	1.70E+108
	h	1	1	0	1	1
F30	*p*	1.21E+112	1.21E+112	NaN	3.45E+107	1.21E+112
	h	1	1	0	1	1

Table 4 The results of Wilcoxon rank sum tests for EWOA with other advanced algorithms

Function		EWOA versus					
		TLBO	GWO	MFO	ETLBO	wPSO	OABC
F1	*p*	1.21E+112	1.21E+112	1.21E+112	NaN	1.21E+112	1.21E+112
	h	1	1	1	0	1	1
F2	*p*	1.21E+112	1.21E+112	1.21E+112	1.21E+112	1.21E+112	1.21E+112
	h	1	1	1	1	1	1
F3	*p*	1.21E+112	1.21E+112	1.21E+112	8.15E–02	1.21E+112	1.21E+112
	h	1	1	1	0	1	1
F4	*p*	1.21E+112	1.21E+112	1.21E+112	1.21E+112	1.21E+112	1.21E+112
	h	1	1	1	1	1	1
F5	*p*	3.02E+111	1.41E+109	3.15E–02	3.02E+111	3.02E+111	4.80E+107
	h	1	1	1	1	1	1
F6	*p*	3.02E+111	3.82E+110	3.02E+111	4.57E+109	3.02E+111	3.02E+111
	h	1	1	1	1	1	1
F7	*p*	1.21E+112	3.34E–01	1.21E+112	NaN	1.21E+112	1.21E+112
	h	1	0	1	0	1	1
F8	*p*	1.21E+112	8.70E+114	1.21E+112	9.65E+106	1.21E+112	1.21E+112
	h	1	1	1	1	1	1
F9	*p*	1.21E+112	4.19E–02	1.21E+112	NaN	1.21E+112	1.21E+112
	h	1	1	1	0	1	1
F10	*p*	3.02E+111	3.02E+111	6.63E–01	3.02E+111	3.02E+111	2.71E–02
	h	1	1	0	1	1	1
F11	*p*	3.02E+111	6.12E+110	7.96E–03	3.02E+111	3.02E+111	2.71E–02
	h	1	1	1	1	1	1
F12	*p*	8.89E+110	8.68E–03	3.02E+111	1.73E+107	5.23E+110	3.02E+111
	h	1	1	1	1	1	1
F13	*p*	5.04E+111	2.15E+110	1.21E+112	1.58E+111	3.02E+111	1.72E+112
	h	1	1	1	1	1	1
F14	*p*	8.89E+110	6.05E+107	1.60E–04	8.48E+109	4.98E+111	2.68E+111
	h	1	1	1	1	1	1

Table 4 continued

Function		EWOA versus					
		TLBO	GWO	MFO	ETLBO	wPSO	OABC
F15	*p*	3.02E+111	5.49E+111	1.00E+00	3.82E+110	3.02E+111	1.82E–03
	h	1	1	0	1	1	1
F16	*p*	3.02E+111	3.34E+111	3.29E–04	5.97E+109	3.02E+111	2.48E+108
	h	1	1	1	1	1	1
F17	*p*	7.22E+106	7.29E–03	3.52E+107	2.52E–01	7.20E–05	1.54E–01
	h	1	1	1	0	1	0
F18	*p*	3.02E+111	1.70E–02	5.32E–03	1.17E+109	1.86E–01	1.61E–06
	h	1	1	1	1	0	1
F19	*p*	1.55E+109	2.13E–04	6.72E+110	7.84E–01	3.35E+108	3.02E+111
	h	1	1	1	0	1	1
F20	*p*	2.92E+109	4.42E–06	8.48E+109	3.40E–01	4.50E+111	4.92E–01
	h	1	1	1	0	1	0
F21	*p*	2.64E–01	1.09E–01	1.03E+106	7.77E+109	2.61E+110	1.99E–02
	h	0	0	1	1	1	1
F22	*p*	5.59E–01	1.43E–05	1.44E–02	7.98E–02	1.86E–03	8.15E+111
	h	0	1	1	0	1	1
F23	*p*	1.21E+112	1.21E+112	1.21E+112	NaN	1.21E+112	1.21E+112
	h	1	1	1	0	1	1
F24	*p*	1.21E+112	1.21E+112	1.21E+112	NaN	1.21E+112	1.21E+112
	h	1	1	1	0	1	1
F25	*p*	1.21E+112	8.24E+113	1.21E+112	3.34E–01	1.21E+112	1.21E+112
	h	1	1	1	0	1	1
F26	*p*	2.15E–01	9.21E–01	1.76E+109	5.80E–02	7.72E+111	7.88E+112
	h	0	0	1	0	1	1
F27	*p*	1.21E+112	1.21E+112	1.21E+112	NaN	1.21E+112	1.21E+112
	h	1	1	1	0	1	1
F28	*p*	1.21E+112	1.21E+112	1.21E+112	NaN	1.21E+112	1.21E+112
	h	1	1	1	0	1	1
F29	*p*	1.21E+112	1.21E+112	1.21E+112	3.13E–04	1.21E+112	1.21E+112
	h	1	1	1	1	1	1
F30	*p*	1.21E+112	1.21E+112	1.21E+112	3.34E–01	1.21E+112	1.21E+112
	h	1	1	1	0	1	1

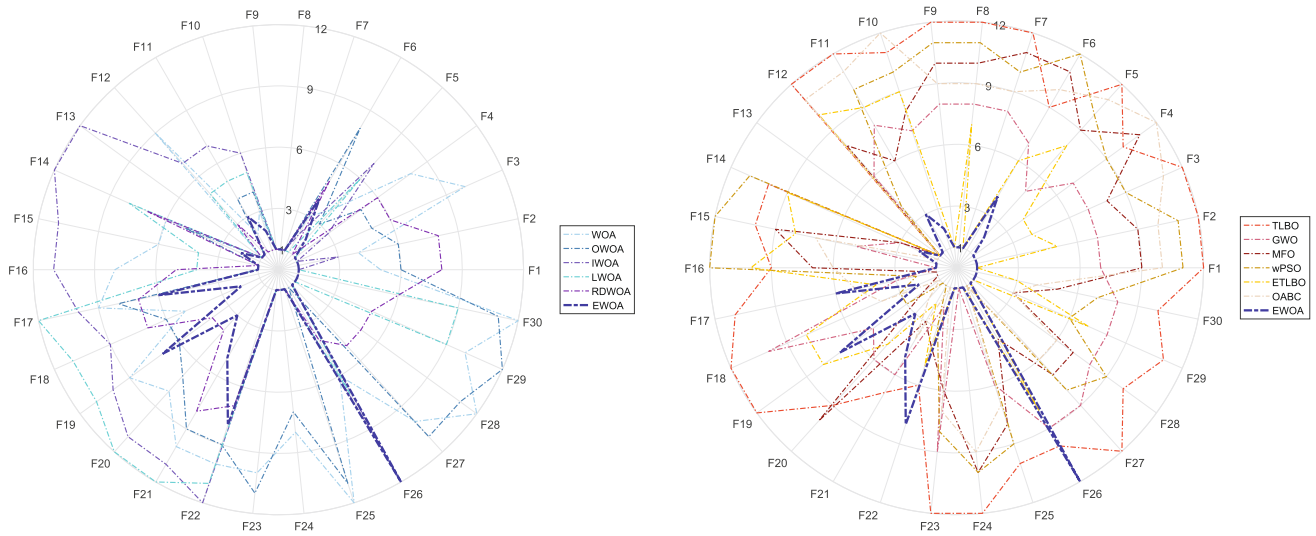
Table 5 Statistical results of Wilcoxon rank-sum tests

	WOA	OWOA	IWOA	LWOA	RDWOA	TLBO	GWO	MFO	ETLBO	wPSO	OABC
Better	22	23	15	16	15	26	24	21	13	23	22
Worse	1	1	2	2	5	1	3	7	2	6	6
Same	7	6	13	12	10	3	3	2	15	1	2
Difference	21	22	13	14	10	25	21	14	11	17	16

Table 6 Friedman tests on unimodal/multimodal functions /fix-dimension functions F1–F16, hybrid functions /composition functions F17–F30 and all functions F1–F30

	i	Algorithms	Average ranking	$z = (R_0 - R_i)/SE$	Unadjusted p	p_{Holm}
Simple (Unimodal/multimodal) F1–F16	12	TLBO	10.8125	5.809941	0	0
	11	wPSO	9.9375	5.123534	0	0.000003
	10	OABC	9.0625	4.437128	0.000009	0.000082
	9	MFO	8.2188	3.775236	0.00016	0.001279
	8	ETLBO	6.5625	2.475966	0.013288	0.093013
	7	IWOA	6.5625	2.475966	0.013288	0.093013
	6	GWO	5.4688	1.617958	0.105672	0.528358
	5	WOA	5.4375	1.593444	0.111061	0.528358
	4	LWOA	4.625	0.956066	0.339039	1.017116
	3	OWOA	4.0938	0.539319	0.589667	1.179333
	2	RDWOA	3.8125	0.318689	0.749963	1.179333
	1	**EWOA**	3.4062			
Complex (Hybrid /composition) F17-30	12	TLBO	10.3214	4.402796	0.000011	0.000118
	11	WOA	9.3571	3.695204	0.00022	0.002197
	10	OWOA	8.6429	3.171062	0.001519	0.013669
	9	LWOA	7.75	2.515884	0.011873	0.094988
	8	GWO	6.6071	1.677256	0.093492	0.654447
	7	IWOA	6.5	1.598634	0.109902	0.659411
	6	MFO	5.1786	0.628971	0.529368	2.646841
	5	wPSO	5.1786	0.628971	0.529368	2.646841
	4	RDWOA	5.1429	0.602764	0.546666	2.646841
	3	ETLBO	4.6071	0.209657	0.833935	2.646841
	2	OABC	4.3929	0.052414	0.958199	2.646841
	1	**EWOA**	4.3214			
All F1–F30	12	TLBO	10.5833	7.23276	0	0
	11	wPSO	7.7167	4.153466	0.000033	0.000327
	10	WOA	7.2833	3.687992	0.000226	0.002034
	9	OABC	6.8833	3.258323	0.001121	0.008966
	8	MFO	6.6667	3.025585	0.002482	0.017371
	7	IWOA	6.55	2.900265	0.003728	0.022371
	6	OWOA	6.2333	2.560111	0.010464	0.052319
	5	LWOA	6.1	2.416888	0.015654	0.062615
	4	GWO	6.0167	2.327373	0.019945	0.062615
	3	ETLBO	5.6667	1.951413	0.051008	0.102016
	2	RDWOA	4.45	0.644503	0.519249	0.519249
	1	**EWOA**	3.85			

adjusted p-value. Through the overall data evaluation, i.e. the Wilcoxon rank-sum tests and the Friedman tests, EWOA is the top average ranking among all 12 comparison algorithms. In addition to the above 30 test problems, tests based on CEC 2019 benchmark functions are also performed. Detailed test functions information, accuracy analysis and convergence behavior are included in the supplementary file. The same Wilcoxon rank-sum and Friedman tests were used to evaluate the optimization ability of EWOA. Among them, the metrics that passed the Friedman test are summarized in Table 7. The "Average ranking" of EWOA is 1.4 which is superior than others. It indicates that the EWOA algorithm outperforms compared WOA variants, which further verifies the improvements of EWOA.



(a) Comparison with WOA variants. (b) Comparison with other advanced algorithms.

Fig. 3 Rank of mean best fitness of EWOA and other compared algorithms

Table 7 Friedman tests on CEC 2019

i	Algorithms	Average ranking	$z = (R_0 - R_i)/SE$	Unadjusted p	p_{Holm}
6	IWOA	5.3	4.661392	0.000003	0.000016
5	LWOA	4.2	3.34664	0.000818	0.003272
4	RDWOA	3.8	2.868549	0.004124	0.012371
3	OWOA	3.2	2.151411	0.031444	0.062887
2	WOA	3.1	2.031889	0.042165	0.062887
1	**EWOA**	1.4			

Actually, the above evaluation results can be predicted from the improved mechanism of EWOA. The comparison results with WOA show that EWOA maintains and improves the exploitation ability of WOA well in the unimodal function test, which benefits from the dynamic shrinking search space and opposite search properties of the IDOL strategy. In the test of complex functions, especially the hybrid and composition functions based on benchmark functions, EWOA can obtain more competitive results. The reason is that once a better solution cannot be found in the prey search stage, in the later stage of the iteration, the exploitation-based WOA will shrink the population to the vicinity of the optimal solution obtained in the previous stage. It is easy to get stuck in a local optimum. Differently, EWOA introduces AIW strategy to improve this phenomenon in the stage of encircling the prey. Groups that are far away from the current prey correspond to a smaller weight value, that is, they are less affected by the current prey to improve the diversity of the group, while groups that are close to the current prey correspond to a larger weight value and continue to maintain inertial encirclement. In addition, during the proposed IDOL generation jump stage, the appropriate mode is adaptively switched according to the current number of convergence stagnation to better balance exploration and exploitation. According to the overall data, EWOA adopts AIW and IDOL strategies to significantly improve the ability of exploration and exploitation, and has stronger global search ability, especially on the test of the most complex composition functions. EWOA can also significantly outperform WOA variants and other classical heuristics as well as some of their advanced variants.

Convergence analysis

Table 8 shows the comparison result (SP) of the algorithm convergence speed measured by the successful performance. The index is taken from [73] and is defined as the average FES and success rate (SR) of successful operations. SR is the ratio of successful runs when algorithms reache the value-to-reach (VTR) to the total number of runs. The SP value of the two algorithms and the algorithm's convergence speed determine relative SP (RSP). Among all the expriments on F1–F16, the dimension is set to 30, and the VTR is set to 10^{-8}.

Table 8 indicates that EWOA has the most vital ability to reach the theoretically optimal solution. The 10/16 test function can reach 100% to obtain the optimal solution, and the average success rate of all test functions is also the high-

Table 8 Comparison of the convergence speed of algorithms

Functions	WOA		OWOA		IWOA		LWOA		RDWOA		TLBO	
	SR	RSP	SR	RSP	SR	RSP	SR	RSP	SR	RSP	SR	RSP
$F1$	1	10.63	1	4.87	1	1	1	0.77	1	8.18	0	×
$F2$	1	8.51	1	4.39	1	1	1	0.9	1	8.12	0	×
$F3$	0	×	1	12.42	1	1	1	0.75	1	8.73	0	×
$F4$	0.93	25.04	1	6.04	1	1	1	1.07	1	8.25	0	×
$F5$	0	×	0	×	0	×	0	×	0	×	0	×
$F6$	0	×	0	×	0	×	0	×	0	×	0	×
$F7$	1	11.69	1	5.65	1	1	1	0.96	1	10.93	0	×
$F8$	1	8.82	1	4.61	1	1	1	0.92	1	7.76	0	×
$F9$	1	11.27	1	5.55	1	1	1	1.04	1	9.68	0	×
$F10$	0	×	0	×	0	×	0	×	0.83	0	0	×
$F11$	0	×	0	×	0	×	0	×	1	0	0	×
$F12$	0.03	1378.01	1	1.23	0	1	0	×	0.93	1.41	0.07	331.55
$F13$	1	1	1	1	0.2	1	0.67	2.25	0.93	1.15	1	1
$F14$	0.5	2.78	0.53	2.44	0	1	0.07	156.25	0.43	3.7	0	×
$F15$	0	×	0.97	1.07	0	1	0	×	0.9	1.23	0	×
$F16$	0	×	0	×	0	1	0	×	0.83	0	0	×

Functions	GWO		MFO		ETLBO		wPSO		OABC		EWOA
	SR	RSP	SR	RSP	SR	RSP	SR	RSP	SR	RSP	SR
$F1$	1	3.88	1	50.32	1	2.18	0	×	0.97	46.18	1
$F2$	1	3.35	1	37.35	1	2.12	0	×	0.03	39074.89	1
$F3$	1	8.23	0	×	1	1.51	0	×	0	×	1
$F4$	1	5.75	0	×	1	1.99	0	×	0	×	1
$F5$	0	×	0	×	0	×	0	×	0	×	0
$F6$	0	×	0	×	0	×	0	×	0	×	0
$F7$	0.97	9.2	0	×	1	2.45	0	×	0.33	608.14	1
$F8$	1	3.62	17	1556.34	1	2.04	0	×	0	×	1
$F9$	0.87	18.06	0.3	718.03	1	2.29	0	×	1	6894.2	1
$F10$	0	×	0.53	0	0	×	0	×	0.67	0	0
$F11$	0	×	0.5	0	0	×	0	×	0.33	0	0
$F12$	0.93	1.61	0	×	13	77.57	0.97	0.6	0	×	0.97
$F13$	1	1	1	1	1	1	17	36	1	1	1
$F14$	0.7	1.42	0.73	1.29	1	69.44	0	×	0.97	0.74	0.83
$F15$	0.87	1.33	0.5	4	0.03	900	0	×	0.73	1.86	1
$F16$	0	×	0.77	0	0.03	3.89	0	×	0.9	0	0.07

est. At the same time, by comparing the iteration numbers needed for achieving the average optimal solution, it can be found that in the tests of functions other than F11, the number of iterations required by the comparison algorithm is mostly greater than that of EWOA, which shows that EWOA has faster convergence rate.

Figures 4 and 5 illustrate the convergence trend of all compared algorithms and EWOA on the benchmark function tests. Two characteristic behaviors of EWOA can be observed in most convergence graphs. The fitness value decreases rapidly at the initial stage of the iteration, and a competitive value can be obtained at the end of the convergence. By observing the convergence of all tested functions, EWOA can converge to a small fitness faster than other algorithms at the beginning of optimization in almost most cases, which benefits from the exploration ability of IDOL through the initialization and jumping generation stages. For a special curve like F30, when EWOA encounters a convergence stagnation and mainly performs exploitation in the second half of the iteration, EWOA can still break the convergence stag-

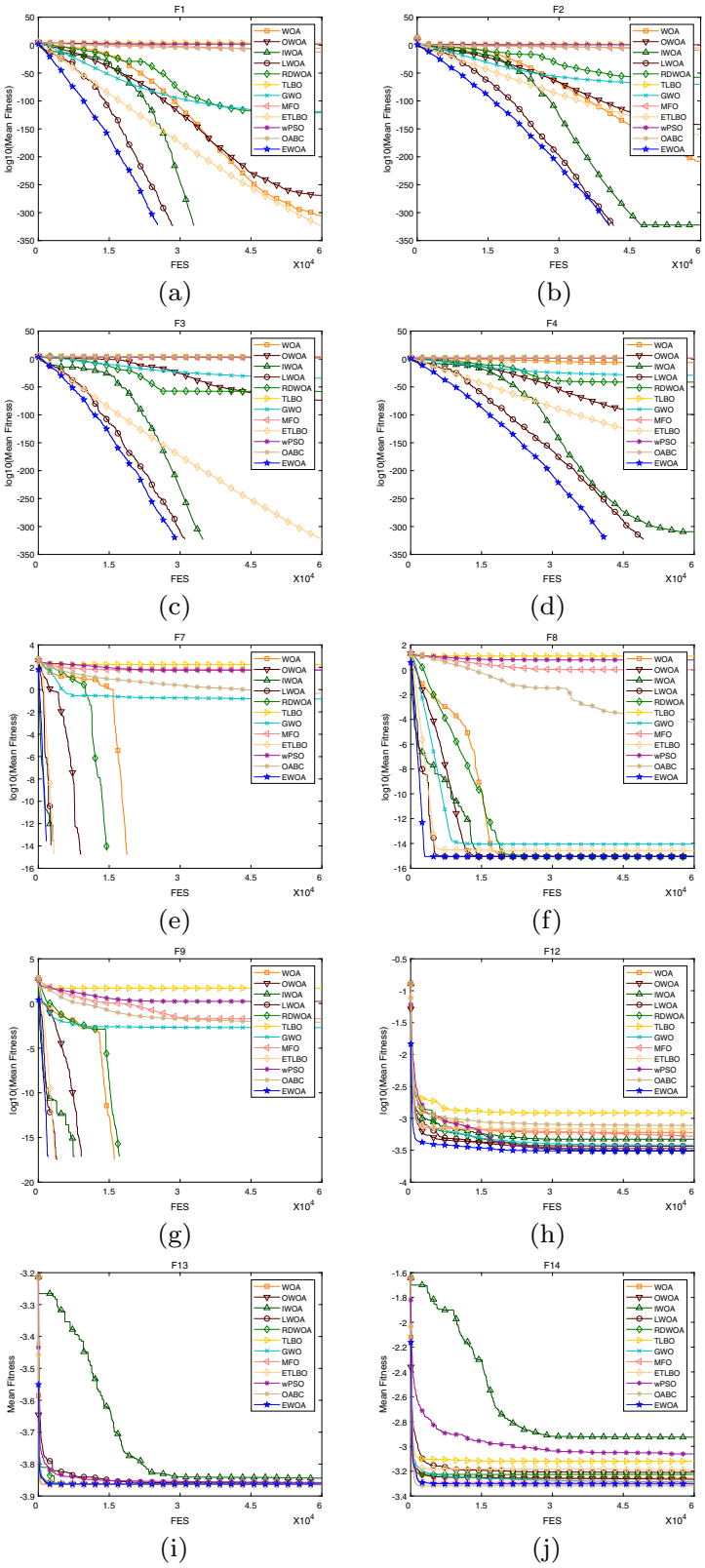


Fig. 4 Convergence trends of EWOA and advanced algorithms for some selected unimodal/multimodal/fixed dimension functions

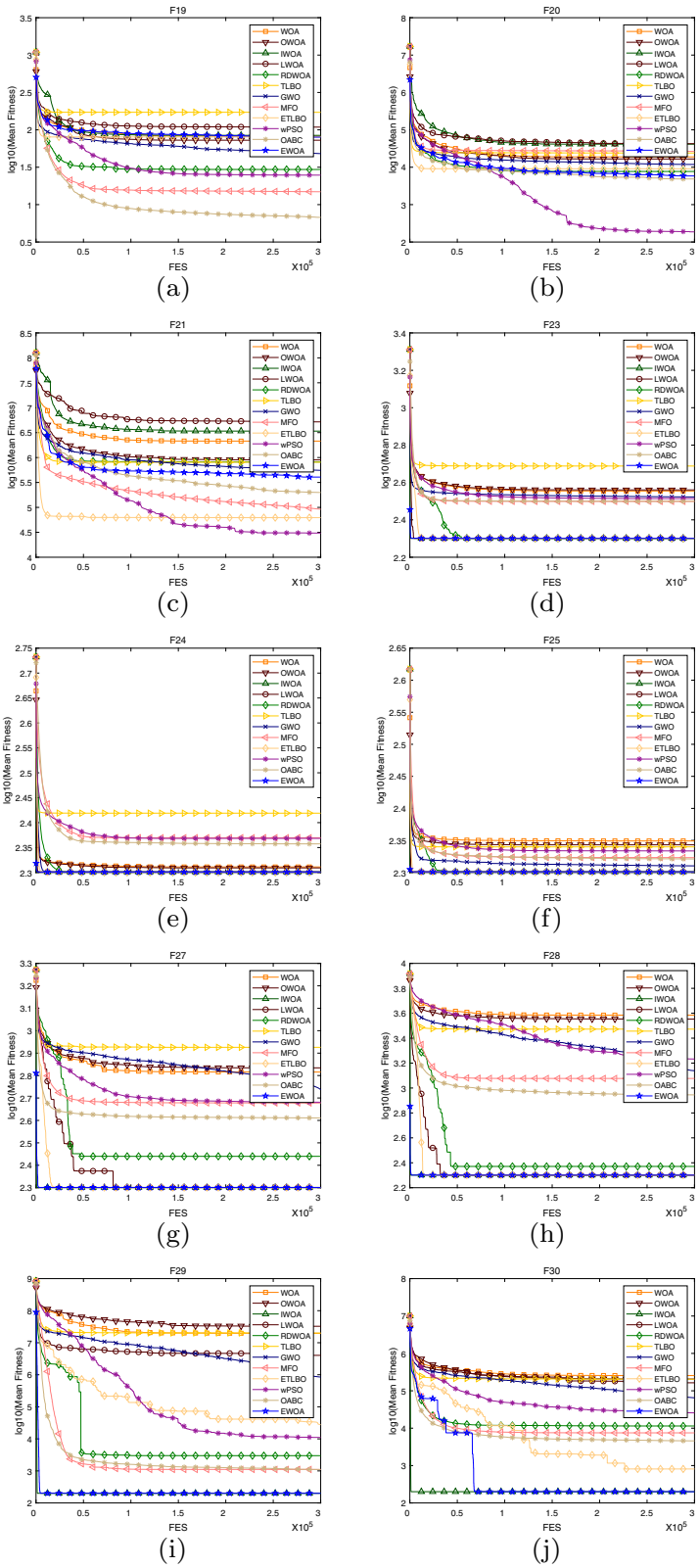


Fig. 5 Convergence trends of EWOA and advanced algorithms for some selected hybrid and composition functions

nation and continue to reduce the fitness. This is because the adaptive inertia weight in AIW-based encircling prey play an important role in enabling the population to ensure diversity to escape the trap of local optima. On the other hand, from the final convergence results, EWOA gets the optimal fitness value at the end of the iteration on F1–F4, F7–F9, F15

Table 9 Results of the CD problem

Algorithm	Optimum variables					Optimum cost
	$x1$	$x2$	$x3$	$x4$	$x5$	
WOA	5.6698	5.3581	4.7643	3.2432	2.8364	13.3833
OWOA	5.6697	5.4496	4.5692	3.1789	3.71	13.4157
IWOA	6.283	5.6289	3.9962	3.5053	2.2624	13.491
LWOA	5.903	5.1835	4.4491	3.6491	2.3424	13.3985
RDWOA	6.1424	5.464	4.4371	3.1984	2.3352	13.4296
SA	12.8653	40.9188	37.761	41.1744	33.0597	14.9933
SCA	9.1362	6.8693	6.118	6.074	1.9859	13.9316
cfPSO	6.6945	4.1532	5.3589	6.4033	3.2079	17.0873
SaDE	6.077	5.3005	4.4597	3.5703	2.092	13.3812
EWOA	6.2079	4.8906	4.4663	3.7409	2.4098	13.367

and F16 and ranks among the top three on F5–F6, F10–12 and F14. For the tests on hybrid and the composite functions F18, F23–F25 and F27–F30, EWOA has the best final convergence results. The significant advantage of the final converged results reflects the EWOA algorithm's ability to balance exploration and exploitation well. It benefits from IDOL's enhancements in exploration and exploitation modes and corresponding adaptive switching rules.

Engineering problems experiments

To verify the effectiveness of EWOA when solving engineering constraint optimization problems, the EWOA and nine other algorithms including WOA, OWOA, IWOA, LWOA, RDWOA, SA, SCA, cfPSO [74] and SaDE [75] are applied to three well known standard engineering problems which are optimization design problems of tension/compression spring, cantilever beam and infinite impulse response (IIR) filter. Among them, the IIR filter design is a continuous optimization problem while the rest are constrained problems. A general death penalty function is used to deal with these constraints so that the algorithm automatically discards infeasible solutions during the optimization process. When solving these engineering optimization problems, each algorithm runs independently 30 times to select the best optimization design result.

Cantilever design problem

The structural cantilever design problem (CD) [76] is to minimize the weight of cantilever beams. A cantilever beam is composed of five hollow blocks, a variable represents each hollow block and the thickness is constant. Therefore, the problem has 5 design variables and 1 vertical displacement constraint.

$$\begin{aligned} &Consider\ \mathbf{x} = [x_1\ \ x_2\ \ x_3\ \ x_4\ \ x_5],\\ &Minimize\ \ f(\mathbf{x}) = 0.6224(x_1 + x_2 + x_3 + x_4 + x_5),\\ &Subject\ g_1(\mathbf{x}) = 61/x_1^3 + 37/x_2^3 + 19/x_3^3\\ &\qquad\qquad +7/x_4^3 + 1/x_5^3 \le 1.\\ &Variable\ ranges\ 0.01 \le x_1, x_2, x_3, x_4, x_5 \le 100. \end{aligned} \tag{23}$$

The results of the CD problem are depicted in Table 9, EWOA obtains the optimal when the five parameters are set to 6.2079, 4.8906, 4.4663, 3.7409, 2.4098 and outperforms other algorithms.

Tension/compression spring design problem

The purpose of the Tension/Compression Spring Design (TCSD) problem is to design a spring with the lightest weight [77]. This design includes three design variables: the wire diameter (d), the mean coil diameter (D) and the length (or number of coils) (N), and needs to meet the four constraints of deflection, shear stress, variable frequency and refractive index. The mathematical model of this problem is as follows:

$$\begin{aligned} &Consider\ \ \mathbf{x} = [x_1\ \ x_2\ \ x_3] = [d\ \ D\ \ N],\\ &Minimize\ \ f(\mathbf{x}) = (x_3 + 2)\, x_2 x_1^2,\\ &Subject\ \ g_1(\mathbf{x}) = 1 - \frac{x_2^3 x_3}{71785 x_1^4} \le 0,\\ &g_2(\mathbf{x}) = \frac{4x_2^2 - x_1 x_2}{12566\left(x_2 x_1^3 - x_1^4\right)}\\ &+\frac{1}{5108 x_1^2} - 1 \le 0,\\ &g_3(\mathbf{x}) = 1 - \frac{140.45 x_1}{x_2^2 x_3} \le 0,\\ &g_4(\mathbf{x}) = \frac{x_1 + x_2}{1.5} - 1 \le 0 \end{aligned}$$

Table 10 Results of the TCSD problem

Algorithm	Optimum variables			Optimum cost
	d	D	N	
WOA	0.0526	0.3794	10.0891	0.012698479
OWOA	0.0542	0.4196	8.3883	0.012774336
IWOA	0.0514	0.3507	11.6652	0.012680419
LWOA	0.05	0.3172	14.0608	0.012736428
RDWOA	0.0524	0.3744	10.3251	0.012674646
SA	0.0625	0.584	8.2141	0.012923956
SCA	0.0945	1.0364	8.431	0.013027366
cfPSO	0.0544	0.4147	8.8689	0.013342345
SaDE	0.0507	0.3317	12.994	0.012771669
EWOA	0.0522	0.3678	10.6985	0.012670417

$$\begin{aligned} &Variable\ ranges \quad 0.05 \le x_1 \le 2.00,\\ &0.25 \le x_2 \le 1.30,\\ &2.00 \le x_3 \le 15.0. \end{aligned} \tag{24}$$

The results of EWOA are compared with other solutions and reported in Table 10. EWOA can generate the optimum cost of 0.012670417 when d, D and N are set to 0.0522, 0.3678 and 10.6985, which is superior to all other algorithms.

Infinite impulse response filter design problem

IIR filters have been widely used in system identification. When the process model is unknown or more complex, the simple finite impulse response (FIR) adaptive filter may not be able to fit the real model accurately. In these cases, it is natural to use IIR adaptive filters to model unknown systems [78]. In terms of system identification, when the filter order is determined, the key task is to configure appropriate filter parameters to minimize the difference between the filter and the real system output value. In this section, a fourth-order IIR is used to model a superior-order plant. The unknown plant and the IIR model hold the following transfer functions [79]:

$$\begin{aligned} H_p\left(z^{-1}\right) &= \frac{1-0.4z^{-2}-0.65z^{-4}+0.26z^{-6}}{1-0.77z^{-2}-0.8498z^{-4}+0.6486z^{-6}}\\ H_M\left(z^{-1}\right) &= \frac{b_0+b_1z^{-1}+b_2z^{-2}+b_3z^{-3}+b_4z^{-4}}{1+a_1z^{-1}+a_2z^{-2}+a_3z^{-3}+a_4z^{-4}} \end{aligned} \tag{25}$$

These nine design variables are the simplest form of the transfer function of the fourth-order IIR model, which are represented by $a_1, a_2, a_3, a_4, b_0, b_1, b_2, b_3, b_4$. The white noise sequence is taken as the input of the system $u(t)$ and set 100 sample cases. $d(t)$ and $y(t)$ represent the output of the actual system and IIR model respectively.

$$\begin{aligned} &Consider \quad \mathbf{x} = [a_1\ \ a_2\ \ a_3\ \ a_4\ \ b_0\ \ b_1\ \ b_2\ \ b_3\ \ b_4],\\ &Minimize \quad f(\mathbf{x}) = \frac{1}{100}\sum_{t=1}^{100}(d(t)-y(t))^2,\\ &Variable\ ranges \quad -1 \le a_1, a_2, a_3, a_4, b_0, b_1, b_2, b_3, b_4 \le 1,\\ &Where \quad u(t) \sim N(0,1),\\ &d(t) = 0.77d(t-2) + 0.8498d(t-4)\\ &\quad +0.6486d(t-6) + u(t) - 0.4u(t-2)\\ &\quad -0.65u(t-4) + 0.26u(t-6),\\ &y(t) = \sum_{i=1}^{4} a_i \cdot y(t-i) + \sum_{j=0}^{4} b_j \cdot u(t-j). \end{aligned} \tag{26}$$

Statistics of the best solution data obtained by 10 algorithms in solving the IIR design problem is depicted in Table 11. It can be observed that EWOA can obtain the optimal fitting residual of 0.00105, which is lower than other algorithms.

Table 11 Results of the IIR problem

Algorithm	Optimum variables									Optimal result
	$a1$	$a2$	$a3$	$a4$	$b0$	$b1$	$b2$	$b3$	$b4$	
WOA	0.9668	−0.0003	0.4065	0.004	−0.2353	−0.0043	−0.0001	−0.0105	−0.8079	0.00685
OWOA	0.9717	−0.0574	0.1114	−0.0272	−0.204	−0.0227	−0.2492	0.0163	−0.6204	0.00837
IWOA	0.6788	0	0.1533	0.0012	0.1731	0	0.0031	0.0315	0	0.10746
LWOA	0.9629	−0.0065	−0.5549	−0.0129	0.1415	−0.0297	−0.9266	0.0142	0.0037	0.02592
RDWOA	0.9955	0.1317	−0.0997	−0.1441	−0.0918	0.139	−0.4124	−0.1422	−0.4705	0.01364
SA	0.7264	0.4308	−0.6538	−0.2054	−0.078	0.5316	−0.0629	0.4164	0.5558	0.15976
SCA	0.7109	0.1616	0.1171	0.2524	0.0881	0.1309	−0.3104	0.0599	−0.3625	0.03398
cfPSO	0.9928	0.0124	0.078	0.0004	−0.2324	0.0095	−0.2457	−0.0069	−0.6322	0.01804
SaDE	0.8765	−0.1241	−0.0343	0.0331	−0.1883	−0.1939	−0.3235	0.1608	−0.5581	0.04836
EWOA	0.9844	0.0003	0.3733	0.0071	−0.3368	0.005	0.0024	0.0037	−0.826	0.00105

According to the above engineering problems' optimization results, EWOA can get the best optimal results in the CD, TCSD and IIR problems. It means that the optimal value of EWOA is better than WOA and its variants under the same iterative conditions. In optimizing specific engineering problems, EWOA is also competitive with other classic and advanced optimization algorithms.

Conclusion

This paper proposes an enhanced WOA (EWOA), which uses new IDOL and AIW strategies to overcome the problems encountered by canonical WOA, including the balance between exploitation and exploration, and easy to fall into local optima. Both IDOL and AIW strategies' mechanisms to improve EWOA are presented in detail. In the proposed EWOA, IDOL plays an important role in population initialization and generation stages. This strategy is inspired by DOL, which not only has the characteristics of asymmetric and dynamically adjusting, but also works better in balancing exploration and exploitation. Inside IDOL, two modes are instructed to enhance the performance of exploration and exploitation, and the adaptive switching rules can choose the appropriate mode according to convergence situation. In another improvement method AIW, the inertia weight can be adaptively adjust referring to the current agents' fittness, which increases the diversity of population in exploitation stage of EWOA.

The experiments consist of benchmark functions and engineering problems are used to evaluate the performance of EWOA. In combination with DOL, IDOL, AIW strategies, respectively, the inter-variants of EOWA, i.e. DOLWOA, IDOLWOA and AIWWOA's are constructed and compared. The test results indicate that EWOA which combined with IDOL and AIW can achieve the best performance. Besides, all the combinations perform better than canonical WOA and the improvement from DOL to IDOL is significant. Canonical WOA, 3 sub-variants of EWOA and 10 advanced MAs are also for comparison, 30 functions including unimodal, multimodal, hybrid and composition are used for numerical testing to valid the exploitation, exploration and comprehensive capabilities of the EWOA. The results indicate that EWOA has consistently high performance in solving complex problems of different dimensions. Non-parametric tests' results, i.e. Wilcoxon rank-sum tests and Friedman tests indicate that whether solving simple (F1–F16) or complex test functions (F17–F30) the comprehensive performance of EWOA is better than other algorithms. In addition to the default dimension 30, the variable dimension tests are also conducted. The test results of unimodal/multimodal functions in dimensions 10, 50, 100 indicate that EWOA outperforms other compared MAs when solving different dimensional problems. Convergence analysis depicts that the proportion of the theoretical optimal solution obtained by the EWOA is the highest and the number of iterations required for the average convergence except for individual test functions is the smallest. Besides, the engineering optimization results present that EWOA is very competitive in solving three practical engineering design problems.

Finally, there is still some potential research work to be done for IDOL and EWOA in the future. Given IDOL is first proposed and conducive to balance the exploration and exploitation, it has the potential to be extended to other MAs which suffering problem of trade-off. For the improvement of EWOA, the application of EWOA in multi-objective problems is also worth exploring.

Supplementary Information The online version contains supplementary material available at https://doi.org/10.1007/s40747-022-00827-1.

Acknowledgements This work was supported by the National Key R&D Program of China (No.2018YFB1700500).

Declarations

Conflicts of interest The authors declare that they have no conflict of interest.

Appendix A

See Table 12.

Table 12 The test functions

ID	Function	Dim	Range	$f_{\min}$
Unimodal functions				
$F1$	$f_1(x) = \sum_{i=1}^{n} x_i^2$	30	[–100,100]	0
$F2$	$f_2(x) = \sum_{i=1}^{n} \lvert x_i \rvert + \prod_{i=1}^{n} \lvert x_i \rvert$	30	[–10, 10]	0
$F3$	$f_3(x) = \sum_{i=1}^{n} \left(\sum_{j-1}^{i} x_j \right)^2$	30	[–100, 100]	0
$F4$	$f_4(x) = \max_i \{ \lvert x_i \rvert , 1 \le i \le n \}$	30	[–100, 100]	0
$F5$	$f_5(x) = \sum_{i=1}^{n-1} \left[100 \left(x_{i+1} - x_i^2 \right)^2 + (x_i - 1)^2 \right]$	30	[–30, 30]	0
$F6$	$f_6(x) = \sum_{i=1}^{n} i x_i^4 +$random [0,1)	30	[–1.28, 1.28]	0
Multimodal functions				
$F7$	$f_7(x) = \sum_{i=1}^{n} \left[x_i^2 - 10 \cos (2\pi x_i) + 10 \right]$	30	[–5.12, 5.12]	0
$F8$	$f_8(x) = -20 \exp \left(-0.2 \sqrt{\frac{1}{n} \sum_{i=1}^{n} x_i^2} \right) - \exp \left(\frac{1}{n} \sum_{i=1}^{n} \cos (2\pi x_i) \right) + 20 + e$	30	[–32, 32]	0
$F9$	$f_9(x) = \frac{1}{4000} \sum_{i=1}^{n} x_i^2 - \prod_{i=1}^{n} \cos \left(\frac{x_i}{\sqrt{i}} \right) + 1$	30	[–600, 600]	0
$F10$	$f_{10}(x) = \frac{\pi}{n} \left\{ 10 \sin (a y_1) + \sum_{i=1}^{n-1} (y_i - 1)^2 \left[1 + 10 \sin^2 (\pi y_{i+1}) \right] + (y_n - 1)^2 \right.$ $\left. + \sum_{i=1}^{n} \mu (x_i, 10, 100, 4) \right\}$	30	[–50, 50]	0
$F11$	$f_{11}(x) = 0.1 \left\{ \sin^2 (3\pi x_i) + \sum_{i=1}^{n} (x_i - 1)^2 \left[1 + \sin^2 (3\pi x_i + 1) \right] \right.$ $\left. + (x_n - 1)^2 \left[1 + \sin^2 (2\pi x_n) \right] \right\} + \sum_{i=1}^{n} \mu (x_i, 5, 100, 4)$	30	[–50, 50]	0
Fixed dimension functions				
$F12$	$f_{12}(x) = \sum_{i=1}^{11} \left[a_i - \frac{x_1 \left(b_i^2 + b_i \right)_2)}{b_i^2 + b_i x_3 + x_4} \right]^2$	4	[–5, 5]	0.00030
$F13$	$f_{13}(x) = - \sum_{i=1}^{4} c_i \exp \left(- \sum_{j=1}^{3} a_{ij} \left(x_j - p_{ij} \right)^2 \right)$	3	[1, 3]	–3.86
$F14$	$f_{14}(x) = - \sum_{i=1}^{4} c_i \exp \left(- \sum_{j=1}^{6} a_{ij} \left(x_j - p_{ij} \right)^2 \right)$	6	[0, 1]	–3.32
$F15$	$f_{15}(x) = - \sum_{i=1}^{5} \left[(X - a_i) (X - a_i)^T + c_i \right]^{-1}$	4	[0, 10]	–10.1532
$F16$	$f_{16}(x) = - \sum_{i=1}^{10} \left[(X - a_i) (X - a_i)^T + c_i \right]^{-1}$	4	[0, 10]	–10.5363
Hybrid Functions (HF)				
$F17$	HF 1 (N = 3)	30	[–100, 100]	1700
$F18$	HF 2 (N = 3)	30	[–100, 100]	1800
$F19$	HF 3 (N = 4)	30	[–100, 100]	1900
$F20$	HF 4 (N = 4)	30	[–100, 100]	2000
$F21$	HF 5 (N = 5)	30	[–100, 100]	2100
$F22$	HF 6 (N = 5)	30	[–100, 100]	2200
Composition Functions (CF)				
$F23$	CF 1 (N = 3)	30	[–100, 100]	2300
$F24$	CF 2 (N = 3)	30	[–100, 100]	2400
$F25$	CF 3 (N = 5)	30	[–100, 100]	2500
$F26$	CF 4 (N = 5)	30	[–100, 100]	2600
$F27$	CF 5 (N = 5)	30	[–100, 100]	2700
$F28$	CF 6 (N = 3)	30	[–100, 100]	2800
$F29$	CF 7 (N = 3)	30	[–100, 100]	2900
$F30$	CF 8 (N = 5)	30	[–100, 100]	3000

Appendix B

See Tables 13,14,15,16 and 17.

Table 13 Results on unimodal functions F1–F6

		F1	F2	F3	F4	F5	F6
WOA	Mean	9.75E–307	3.04E–210	2.06E+03	4.85E–07	2.64E+01	3.18E–05
	Std	0.00E+00	0.00E+00	1.90E+03	2.58E–06	4.42E–01	3.42E–05
OWOA	Mean	7.92E–270	5.85E–143	1.68E–74	5.09E–100	2.54E+01	1.02E–03
	Std	0.00E+00	1.75E–142	9.02E–74	1.63E–99	4.03E–01	6.68E–04
IWOA	Mean	**0.00E+00**	6.90E–323	**0.00E+00**	4.10E–310	2.80E+01	2.18E–05
	Std	0.00E+00	0.00E+00	0.00E+00	0.00E+00	3.36E–01	1.78E–05
LWOA	Mean	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	2.71E+01	**2.08E–05**
	Std	0.00E+00	0.00E+00	0.00E+00	0.00E+00	4.10E–01	1.75E–05
RDWOA	Mean	1.76E–120	2.18E–58	9.09E–59	4.03E–42	**1.76E+01**	2.51E–04
	Std	9.57E–120	1.19E–57	4.98E–58	2.21E–41	1.08E+01	3.05E–04
TLBO	Mean	5.58E+03	3.06E+01	8.50E+03	2.44E+01	2.22E+06	3.37E–03
	Std	1.41E+03	4.76E+00	3.32E+03	3.89E+00	1.34E+06	1.80E–03
GWO	Mean	4.11E–122	9.03E–71	4.12E–35	6.29E–30	2.66E+01	3.62E–04
	Std	7.98E–122	1.59E–70	1.29E–34	2.26E–29	6.35E–01	2.03E–04
MFO	Mean	2.58E–13	4.51E–10	2.68E+01	3.90E+01	6.37E+01	8.87E–02
	Std	6.18E–13	6.73E–10	2.45E+01	5.04E+00	7.64E+01	5.60E–02
ETLBO	Mean	**0.00E+00**	2.25E–162	4.90E–324	8.55E–159	2.89E+01	2.92E–04
	Std	0.00E+00	1.15E–161	0.00E+00	2.07E–158	2.84E–02	1.64E–04
wPSO	Mean	6.95E+01	3.93E+00	1.59E+03	1.63E+01	2.94E+03	5.38E–01
	Std	3.63E+01	1.79E+00	6.16E+02	3.04E+00	1.94E+03	3.08E–01
OABC	Mean	2.58E–06	1.07E–05	4.27E+03	6.65E+01	3.07E+04	4.51E–02
	Std	1.42E–05	1.77E–05	3.62E+03	1.48E+01	1.46E+05	1.42E–02
EWOA	Mean	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	2.47E+01	5.38E–05
	Std	0.00E+00	0.00E+00	0.00E+00	0.00E+00	6.76E–01	5.72E–05
	Rank	1	1	1	1	2	4

Table 14 Results on multimodal functions F7–F11

		F7	F8	F9	F10	F11
WOA	Mean	**0.00E+00**	**8.88E–16**	**0.00E+00**	1.20E–03	1.77E–02
	Std	0.00E+00	0.00E+00	0.00E+00	2.31E–03	2.52E–02
OWOA	Mean	**0.00E+00**	**8.88E–16**	**0.00E+00**	1.66E–03	9.08E–02
	Std	0.00E+00	0.00E+00	0.00E+00	3.87E–03	1.30E–01
IWOA	Mean	**0.00E+00**	**8.88E–16**	**0.00E+00**	3.43E–02	3.67E–01
	Std	0.00E+00	0.00E+00	0.00E+00	2.87E–02	2.02E–01
LWOA	Mean	**0.00E+00**	**8.88E–16**	**0.00E+00**	8.96E–03	2.61E–01
	Std	0.00E+00	0.00E+00	0.00E+00	7.63E–03	3.84E–01
RDWOA	Mean	**0.00E+00**	**8.88E–16**	**0.00E+00**	**2.59E–05**	**2.76E–03**
	Std	0.00E+00	0.00E+00	0.00E+00	9.55E–05	5.13E–03
TLBO	Mean	1.77E+02	1.28E+01	5.19E+01	8.77E+04	1.60E+06
	Std	2.38E+01	1.03E+00	1.22E+01	1.35E+05	1.28E+06

Table 14 continued

		F7	F8	F9	F10	F11
GWO	Mean	1.47E–01	8.82E–15	1.98E–03	3.71E–02	4.58E–01
	Std	8.03E–01	2.22E–15	5.99E–03	2.26E–02	1.97E–01
MFO	Mean	5.90E+01	1.02E+00	1.97E–02	1.49E–01	2.79E–01
	Std	1.63E+01	1.21E+00	2.45E–02	2.69E–01	8.79E–01
ETLBO	Mean	**0.00E+00**	2.66E–15	**0.00E+00**	7.42E–01	2.91E+00
	Std	0.00E+00	1.81E–15	0.00E+00	2.02E–01	2.02E–01
wPSO	Mean	5.58E+01	6.18E+00	1.71E+00	9.95E+00	4.33E+01
	Std	1.14E+01	1.03E+00	3.95E–01	4.87E+00	2.41E+01
OABC	Mean	4.65E–01	5.91E–05	8.03E–03	2.49E+05	3.95E+05
	Std	1.19E+00	1.35E–04	1.10E–02	9.45E+05	1.78E+06
EWOA	Mean	**0.00E+00**	**8.88E–16**	**0.00E+00**	5.98E–05	2.85E–02
	Std	0.00E+00	0.00E+00	0.00E+00	2.96E–04	4.29E–02
	Rank	1	1	1	2	3

Table 15 Results on fixed dimension functions F12–F16

		F12	F13	F14	F15	F16
WOA	Mean	6.02E–04	−3.86E+00	−3.26E+00	−8.78E+00	−8.54E+00
	Std	3.78E–04	9.88E–06	6.71E–02	2.29E+00	2.64E+00
OWOA	Mean	3.08E–04	−3.86E+00	−3.27E+00	−1.02E+01	−1.05E+01
	Std	2.18E–08	1.51E–06	6.11E–02	2.89E–05	2.05E–05
IWOA	Mean	4.71E–04	−3.84E+00	−2.93E+00	−4.86E+00	−4.77E+00
	Std	1.04E–04	3.35E–02	3.44E–01	1.41E–01	7.50E–01
LWOA	Mean	3.66E–04	−3.86E+00	−3.21E+00	−1.01E+01	−1.02E+01
	Std	7.63E–05	3.96E–03	9.53E–02	2.40E–03	1.75E+00
RDWOA	Mean	3.13E–04	−3.86E+00	−3.23E+00	−1.02E+01	−1.00E+01
	Std	3.18E–05	2.00E–03	1.63E–01	2.45E–03	1.89E+00
TLBO	Mean	1.22E–03	−3.86E+00	−3.12E+00	−6.15E+00	−8.31E+00
	Std	3.65E–03	1.07E–10	1.43E–01	2.19E+00	2.65E+00
GWO	Mean	3.81E–04	−3.86E+00	−3.29E+00	−9.64E+00	−1.05E+01
	Std	2.84E–04	5.23E–06	5.55E–02	1.55E+00	2.31E–05
MFO	Mean	5.24E–04	**−3.86E+00**	−3.29E+00	−6.98E+00	−8.92E+00
	Std	1.92E–04	2.71E–15	5.35E–02	3.34E+00	3.01E+00
ETLBO	Mean	6.84E–04	−3.86E+00	−3.19E+00	−7.02E+00	−8.13E+00
	Std	1.04E–03	2.55E–13	1.22E–01	2.81E+00	3.10E+00
wPSO	Mean	3.38E–04	−3.86E+00	−3.06E+00	−4.02E+00	−4.35E+00
	Std	1.69E–04	3.03E–03	6.47E–02	8.44E–01	9.46E–01
OABC	Mean	7.76E–04	**−3.86E+00**	**−3.32E+00**	−8.72E+00	−9.83E+00
	Std	3.83E–04	2.27E–15	2.17E–02	2.45E+00	2.15E+00
EWOA	Mean	**3.08E–04**	−3.86E+00	−3.30E+00	**−1.02E+01**	**−1.05E+01**
	Std	3.21E–08	2.84E–08	4.51E–02	4.23E–06	2.59E–06
	Rank	1	5	2	1	1

Table 16 Results on hybrid functions F17–F22

		F17	F18	F19	F20	F21	F22
WOA	Mean	4.75E+06	1.19E+04	8.20E+01	1.89E+04	2.14E+06	6.87E+02
	Std	3.96E+06	1.44E+04	4.42E+01	9.83E+03	2.71E+06	2.09E+02
OWOA	Mean	4.45E+06	2.32E+04	7.27E+01	1.66E+04	8.97E+05	6.49E+02
	Std	3.88E+06	4.87E+04	3.77E+01	8.83E+03	9.93E+05	1.96E+02
IWOA	Mean	5.51E+06	4.23E+06	8.44E+01	4.14E+04	3.39E+06	9.01E+02
	Std	5.06E+06	2.14E+07	4.58E+01	1.83E+04	4.08E+06	2.98E+02
LWOA	Mean	1.46E+07	6.98E+07	1.09E+02	4.27E+04	5.20E+06	8.09E+02
	Std	8.73E+06	6.68E+07	3.31E+01	2.45E+04	4.49E+06	2.30E+02
RDWOA	Mean	3.01E+06	4.96E+04	2.95E+01	7.71E+03	8.16E+05	5.91E+02
	Std	2.10E+06	3.08E+04	3.16E+01	4.54E+03	6.55E+05	1.86E+02
TLBO	Mean	7.65E+06	9.15E+07	1.72E+02	2.40E+04	8.06E+05	5.76E+02
	Std	5.50E+06	1.31E+08	4.92E+01	1.18E+04	8.77E+05	1.46E+02
GWO	Mean	7.32E+05	5.59E+06	4.81E+01	1.19E+04	5.54E+05	3.66E+02
	Std	6.14E+05	1.48E+07	2.86E+01	4.05E+03	1.20E+06	1.72E+02
MFO	Mean	**2.32E+05**	**1.20E+03**	1.49E+01	2.65E+04	9.15E+04	4.66E+02
	Std	1.42E+05	1.68E+03	2.05E+01	1.87E+04	5.70E+04	2.32E+02
ETLBO	Mean	1.54E+06	5.40E+05	8.03E+01	9.09E+03	6.22E+04	5.26E+02
	Std	1.84E+06	1.33E+06	3.38E+01	8.54E+03	5.40E+04	1.73E+02
wPSO	Mean	4.87E+05	4.63E+03	2.48E+01	**1.87E+02**	**3.05E+04**	4.58E+02
	Std	5.71E+05	1.69E+04	2.21E+01	1.03E+02	3.78E+04	1.51E+02
OABC	Mean	2.30E+06	1.00E+04	**6.75E+00**	4.86E+03	1.99E+05	**2.00E+02**
	Std	9.65E+05	6.53E+03	9.83E–01	2.11E+03	1.04E+05	7.77E+01
EWOA	Mean	2.54E+06	2.41E+03	7.99E+01	5.85E+03	4.06E+05	6.03E+02
	Std	3.03E+06	2.51E+03	3.52E+01	3.93E+03	3.75E+05	1.70E+02
	Rank	6	2	7	3	5	8

Table 17 Results on composition functions F23–F30

		F23	F24	F25	F26	F27	F28	F29	F30
WOA	Mean	3.58E+02	2.05E+02	2.23E+02	1.04E+02	6.54E+02	3.84E+03	1.97E+07	2.55E+05
	Std	1.66E+01	3.90E+00	1.18E+01	1.79E+01	2.62E+02	7.18E+02	2.24E+07	1.61E+05
OWOA	Mean	3.63E+02	2.04E+02	2.20E+02	1.00E+02	6.83E+02	3.57E+03	3.26E+07	2.07E+05
	Std	1.82E+01	2.31E+00	1.23E+01	1.01E–01	2.77E+02	6.19E+02	3.09E+07	1.66E+05
IWOA	Mean	**2.00E+02**	**2.00E+02**	**2.00E+02**	1.50E+02	**2.00E+02**	**2.00E+02**	**2.00E+02**	**2.00E+02**
	Std	0.00E+00	0.00E+00	0.00E+00	4.98E+01	0.00E+00	0.00E+00	0.00E+00	0.00E+00
LWOA	Mean	**2.00E+02**	**2.00E+02**	**2.00E+02**	1.27E+02	**2.00E+02**	**2.00E+02**	4.04E+06	1.64E+05
	Std	0.00E+00	0.00E+00	0.00E+00	4.38E+01	0.00E+00	0.00E+00	3.94E+06	1.48E+05
RDWOA	Mean	**2.00E+02**	**2.00E+02**	**2.00E+02**	1.01E+02	2.75E+02	2.35E+02	2.99E+03	1.14E+04
	Std	0.00E+00	3.25E–03	0.00E+00	1.51E–01	2.27E+02	1.88E+02	2.23E+03	1.02E+04
TLBO	Mean	4.87E+02	2.62E+02	2.19E+02	1.50E+02	8.43E+02	2.97E+03	2.01E+07	2.04E+05
	Std	5.43E+01	6.61E+00	2.54E+00	4.82E+01	2.36E+02	6.08E+02	2.32E+07	1.19E+05
GWO	Mean	3.33E+02	**2.00E+02**	2.04E+02	1.47E+02	5.46E+02	1.38E+03	8.47E+05	6.49E+04
	Std	1.09E+01	7.68E–04	5.73E+00	4.96E+01	1.02E+02	3.41E+02	1.98E+06	3.78E+04
MFO	Mean	3.15E+02	2.34E+02	2.10E+02	1.01E+02	4.76E+02	1.19E+03	1.11E+03	7.50E+03
	Std	1.20E–01	6.79E+00	2.76E+00	1.35E–01	1.37E+02	1.71E+02	2.53E+02	6.32E+03
ETLBO	Mean	**2.00E+02**	**2.00E+02**	**2.00E+02**	1.38E+02	**2.00E+02**	**2.00E+02**	2.78E+04	8.11E+02
	Std	0.00E+00	0.00E+00	8.30E–14	4.68E+01	0.00E+00	0.00E+00	7.81E+04	3.29E+03

Table 17 continued

		F23	F24	F25	F26	F27	F28	F29	F30
wPSO	Mean	3.26E+02	2.34E+02	2.16E+02	1.00E+02	4.80E+02	1.71E+03	1.10E+04	2.57E+04
	Std	3.39E+00	7.10E+00	2.72E+00	1.15E–01	1.18E+02	3.88E+02	2.60E+04	1.77E+04
OABC	Mean	3.16E+02	2.28E+02	2.09E+02	**1.00E+02**	4.09E+02	8.81E+02	1.11E+03	4.57E+03
	Std	5.54E–01	8.18E–01	1.42E+00	6.18E–02	4.85E+00	4.73E+01	3.17E+02	1.48E+03
EWOA	Mean	**2.00E+02**	**2.00E+02**	**2.00E+02**	1.77E+02	**2.00E+02**	**2.00E+02**	**2.00E+02**	**2.00E+02**
	Std	0.00E+00	0.00E+00	0.00E+00	4.20E+01	0.00E+00	0.00E+00	0.00E+00	0.00E+00
	Rank	1	1	1	12	1	1	1	1

References

1. Yang XS (2010) Nature-inspired metaheuristic algorithms. Luniver press, Beckington
2. Beheshti Z, Shamsuddin SMH (2013) A review of population-based meta-heuristic algorithms. Int J Adv Soft Comput Appl 5(1):1–35
3. Neapolitan RE, Naimipour K (2004) Foundations of algorithms using Java pseudocode. Jones & Bartlett Learning
4. Qiao W, Moayedi H, Foong LK (2020) Nature-inspired hybrid techniques of iwo, da, es, ga, and ica, validated through a k-fold validation process predicting monthly natural gas consumption. Energy and Buildings p 110023
5. Selman B, Gomes CP (2006) Hill-climbing search. Encyclo Cogn Sci 81:82
6. Bertsimas D, Tsitsiklis J et al (1993) Simulated annealing. Stat Sci 8(1):10–15
7. Rana N, Abd Latiff MS, Chiroma H, et al. (2020) Whale optimization algorithm: a systematic review of contemporary applications, modifications and developments. Neural Computing and Applications pp 1–33
8. Whitley D (1994) A genetic algorithm tutorial. Stat Comput 4(2):65–85
9. Eberhart R, Kennedy J (1995) Particle swarm optimization. In: Proceedings of the IEEE international conference on neural networks, Citeseer 4:1942–1948
10. Dorigo M, Birattari M, Stutzle T (2006) Ant colony optimization. IEEE Comput intell magaz 1(4):28–39
11. Shi Y (2011) Brain storm optimization algorithm. In: International conference in swarm intelligence, Springer, pp 303–309
12. Yang XS (2011) Bat algorithm for multi-objective optimisation. Int J Bio-Inspir Comput 3(5):267–274
13. Mirjalili S (2015) The ant lion optimizer. Adv Eng Soft 83:80–98
14. Karaboga D, Basturk B (2008) On the performance of artificial bee colony (abc) algorithm. Appl Soft Comput 8(1):687–697
15. Simon D (2008) Biogeography-based optimization. IEEE Trans Evol Comput 12(6):702–713
16. Rao RV, Savsani VJ, Vakharia D (2011) Teaching-learning-based optimization: a novel method for constrained mechanical design optimization problems. Comput-Aided Des 43(3):303–315
17. Emary E, Zawbaa HM, Grosan C, Hassenian AE (2015) Feature subset selection approach by gray-wolf optimization. In: Afro-European conference for industrial advancement, Springer, pp 1–13
18. Mirjalili S (2016) Sca: a sine cosine algorithm for solving optimization problems. Knowl Based Syst 96:120–133
19. Mirjalili S (2015) Moth-flame optimization algorithm: A novel nature-inspired heuristic paradigm. Knowl Based Syst 89:228–249
20. Satapathy S, Naik A (2016) Social group optimization (sgo): a new population evolutionary optimization technique. Complex Intell Syst 2(3):173–203
21. Senaratne R, Halgamuge S, Hsu A (2009) Face recognition by extending elastic bunch graph matching with particle swarm optimization. J Mult 4(4):204–214
22. Cao K, Yang X, Chen X, Zang Y, Liang J, Tian J (2012) A novel ant colony optimization algorithm for large-distorted fingerprint matching. Patt Recog 45(1):151–161
23. Zobolas G, Tarantilis CD, Ioannou G (2008) Exact, heuristic and meta-heuristic algorithms for solving shop scheduling problems. In: Metaheuristics for scheduling in industrial and manufacturing applications, Springer, pp 1–40
24. Gao K, Huang Y, Sadollah A, Wang L (2019) A review of energy-efficient scheduling in intelligent production systems. Comp and Intell Sys pp 1–13
25. Serrano-Pérez O, Villarreal-Cervantes MG, González-Robles JC, Rodríguez-Molina A (2019) Meta-heuristic algorithms for the control tuning of omnidirectional mobile robots. Eng Optim
26. Yu Y, Xu Y, Wang F, Li W, Mai X, Wu H (2020) Adsorption control of a pipeline robot based on improved pso algorithm. Comp and Intell Sys pp. 1–7
27. Aladag CH, Yolcu U, Egrioglu E, Dalar AZ (2012) A new time invariant fuzzy time series forecasting method based on particle swarm optimization. Appl Soft Comput 12(10):3291–3299
28. Yang Y, Duan Z (2020) An effective co-evolutionary algorithm based on artificial bee colony and differential evolution for time series predicting optimization. Comp and Intell Sys 6:299–308
29. Mirjalili S, Lewis A (2016) The whale optimization algorithm. Adv Eng Soft 95:51–67
30. Luan F, Cai Z, Wu S, Jiang T, Li F, Yang J (2019) Improved whale algorithm for solving the flexible job shop scheduling problem. Mathematics 7(5):384
31. Pandey AC, Tikkiwal VA (2021) Stance detection using improved whale optimization algorithm. Comp and Intell Sys 7(3):1649–1672
32. 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
33. Zhang H, Tang L, Yang C, Lan S (2019) Locating electric vehicle charging stations with service capacity using the improved whale optimization algorithm. Adv Eng Info 41:100901
34. Saidala RK, Devarakonda N (2018) Improved whale optimization algorithm case study: clinical data of anaemic pregnant woman. In: Data engineering and intelligent computing, Springer, pp 271–281
35. Abdel-Basset M, El-Shahat D, El-Henawy I, Sangaiah AK, Ahmed SH (2018) A novel whale optimization algorithm for cryptanalysis in merkle-hellman cryptosystem. Mob Netw Appl 23(4):723–733

36. Xu Z, Yu Y, Yachi H, Ji J, Todo Y, Gao S (2018) A novel memetic whale optimization algorithm for optimization. In: International Conference on Swarm Intelligence, Springer, pp 384–396
37. Gharehchopogh FS, Gholizadeh H (2019) A comprehensive survey: Whale optimization algorithm and its applications. Swarm Evol Comput 48:1–24
38. Kamaruzaman AF, Zain AM, Yusuf SM, Udin A (2013) Levy flight algorithm for optimization problems-a literature review. In: Applied Mechanics and Materials. Trans Tech Publ 421:496–501
39. Ling Y, Zhou Y, Luo Q (2017) Lévy flight trajectory-based whale optimization algorithm for global optimization. IEEE access 5:6168–6186
40. Sun Y, Wang X, Chen Y, Liu Z (2018) A modified whale optimization algorithm for large-scale global optimization problems. Expert Syst Appl 114:563–577
41. Yu Y, Wang H, Li N, Su Z, Wu J (2017) Automatic carrier landing system based on active disturbance rejection control with a novel parameters optimizer. Aerosp Sci Technol 69:149–160
42. Hu H, Bai Y, Xu T (2017) Improved whale optimization algorithms based on inertia weights and theirs applications. Int J Circuits Syst Signal Process 11:12–26
43. Chen H, Yang C, Heidari AA, Zhao X (2020) An efficient double adaptive random spare reinforced whale optimization algorithm. Expert Syst Appl 154
44. Tizhoosh HR (2005) Opposition-based learning: a new scheme for machine intelligence. In: International conference on computational intelligence for modelling, control and automation and international conference on intelligent agents, web technologies and internet commerce (CIMCA-IAWTIC'06), IEEE, vol 1, pp 695–701
45. Rahnamayan S, Tizhoosh HR, Salama MM (2007) Quasi-oppositional differential evolution. In: 2007 IEEE congress on evolutionary computation, IEEE, pp 2229–2236
46. Wang H, Wu Z, Rahnamayan S, Liu Y, Ventresca M (2011) Enhancing particle swarm optimization using generalized opposition-based learning. Inf Sci Inf. Sci. 181(20):4699–4714
47. Ergezer M, Simon D, Du D (2009) Oppositional biogeography-based optimization. In: 2009 IEEE international conference on systems, man and cybernetics, IEEE, pp 1009–1014
48. Zhou X, Wu Z, Wang H (2012) Elite opposition-based differential evolution for solving large-scale optimization problems and its implementation on gpu. In: 2012 13th International Conference on Parallel and Distributed Computing. Applications and Technologies, IEEE, pp 727–732
49. Alamri HS, Alsariera YA, Zamli KZ (2018) Opposition-based whale optimization algorithm. Adv Sci Lett 24(10):7461–7464
50. Luo J, He F, Yong J (2020) An efficient and robust bat algorithm with fusion of opposition-based learning and whale optimization algorithm. Intell Data Anal 24(3):581–606
51. Qiang Z, Qiaoping F, Xingjun H, Jun L (2020) Parameter estimation of muskingum model based on whale optimization algorithm with elite opposition-based learning. In: IOP Conference Series: Materials Science and Engineering, IOP Publishing, vol 780, p 022013
52. Chen H, Li W, Yang X (2020) A whale optimization algorithm with chaos mechanism based on quasi-opposition for global optimization problems. Expert Syst Appl 158
53. Kumar M, Chaparala A (2019) Obc-woa: opposition-based chaotic whale optimization algorithm for energy efficient clustering in wireless sensor network. Intelligence 250:1
54. Xu Y, Yang Z, Li X, Kang H, Yang X (2020) Dynamic opposite learning enhanced teaching-learning-based optimization. Knowl-Based Syst 188:104966
55. Xu Y, Yang X, Yang Z, Li X, Wang P, Ding R, Liu W (2021) An enhanced differential evolution algorithm with a new oppositional-mutual learning strategy. Neurocomputing 435:162–175
56. Dong H, Xu Y, Li X, Yang Z, Zou C (2021) An improved antlion optimizer with dynamic random walk and dynamic opposite learning. Knowl-Based Syst 216:106752
57. Zhang L, Hu T, Yang Z, Yang D, Zhang J (2021) Elite and dynamic opposite learning enhanced sine cosine algorithm for application to plat-fin heat exchangers design problem. Neural Comput Appl pp 1–14
58. Yang XS, Deb S (2009) Cuckoo search via lévy flights. In: 2009 World congress on nature & biologically inspired computing (NaBIC), IEEE, pp 210–214
59. Mantegna RN (1994) Fast, accurate algorithm for numerical simulation of levy stable stochastic processes. Phys Rev E 49(5):4677
60. Yu X, Liu J, Li H (2009) An adaptive inertia weight particle swarm optimization algorithm for iir digital filter. In: 2009 International Conference on Artificial Intelligence and Computational Intelligence, IEEE, vol 1, pp 114–118
61. Qin Z, Yu F, Shi Z, Wang Y (2006) Adaptive inertia weight particle swarm optimization. In: International conference on Artificial Intelligence and Soft Computing, Springer, pp 450–459
62. Nickabadi A, Ebadzadeh MM, Safabakhsh R (2011) A novel particle swarm optimization algorithm with adaptive inertia weight. Appl Soft Comput 11(4):3658–3670
63. Yao X, Liu Y, Lin G (1999) Evolutionary programming made faster. IEEE Trans Evol Comput 3(2):82–102
64. Digalakis JG, Margaritis KG (2001) On benchmarking functions for genetic algorithms. Int J Comput Math 77(4):481–506
65. Molga M, Smutnicki C (2005) Test functions for optimization needs. Test Funct Optim Need 101:48
66. Yang XS (2010) Firefly algorithm, stochastic test functions and design optimisation. Int J Bio-Ins Comput 2(2):78–84
67. Liang JJ, Qu BY, Suganthan PN (2013) Problem definitions and evaluation criteria for the cec 2014 special session and competition on single objective real-parameter numerical optimization. Zhengzhou University, Zhengzhou China and Technical Report, Nanyang Technological University, Singapore, Computational Intelligence Laboratory, pp. 635
68. El-Abd M (2011) Opposition-based artificial bee colony algorithm. In: Proceedings of the 13th annual conference on Genetic and evolutionary computation, pp. 109–116
69. Bansal JC, Singh P, Saraswat M, Verma A, Jadon SS, Abraham A (2011) Inertia weight strategies in particle swarm optimization. In: 2011 Third world congress on nature and biologically inspired computing, IEEE, pp. 633–640
70. Niu Q, Zhang H, Li K (2014) An improved tlbo with elite strategy for parameters identification of pem fuel cell and solar cell models. Int J Hydrog Energy 39(8):3837–3854
71. García S, Fernández A, Luengo J, Herrera F (2010) Advanced nonparametric tests for multiple comparisons in the design of experiments in computational intelligence and data mining: Experimental analysis of power. Inf Sci 180(10):2044–2064
72. Derrac J, García S, Molina D, Herrera F (2011) A practical tutorial on the use of nonparametric statistical tests as a methodology for comparing evolutionary and swarm intelligence algorithms. Swarm Evol Comput 1(1):3–18
73. Park SY, Lee JJ (2015) Stochastic opposition-based learning using a beta distribution in differential evolution. IEEE Trans Cybern 46(10):2184–2194
74. Clerc M, Kennedy J (2002) The particle swarm-explosion, stability, and convergence in a multidimensional complex space. IEEE Trans Evol Comput 6(1):58–73
75. Qin AK, Suganthan PN (2005) Self-adaptive differential evolution algorithm for numerical optimization. In: 2005 IEEE congress on evolutionary computation, IEEE, vol 2, pp 1785–1791
76. Mirjalili S, Mirjalili SM, Hatamlou A (2016) Multi-verse optimizer: a nature-inspired algorithm for global optimization. Neural Comput Appl 27(2):495–513

77. Arora JS (2004) Introduction to optimum design. Elsevier, Amsterdam
78. Kukrer O (2011) Analysis of the dynamics of a memoryless nonlinear gradient iir adaptive notch filter. Sig Process 91(10):2379–2394
79. Cuevas E, Gálvez J, Hinojosa S, Avalos O, Zaldívar D, Pérez-Cisneros M (2014) A comparison of evolutionary computation techniques for iir model identification. J Appl Math 2014