scientific reports

Check for updates

OPEN

Hybrid strategy enhanced crayfish optimization algorithm for breast cancer prediction

Yu-Jiong Li

Crayfish Optimization Algorithm (COA) suffers from degradation of diversity, insufficient exploratory capability, a propensity to become caught in local optima, and an imprecise search engine for optimization. To address these issues, the current research introduces a hybrid strategy enhanced crayfish optimization algorithm (MSCOA). Initially, a chaotic inverse exploration initialization method is utilized to establish the population's position with high diversity, significantly enhancing the global exploration capability. Second, an adaptive t-distributed feeding strategy was employed to define the connection between feeding behavior and temperature, increasing population variety and enhanced the algorithm's local search effectiveness. Finally, an adaptive ternary optimization mechanism is introduced in the exploration phase: a curve growth acceleration factor is used to collaboratively guide global and individual optimal information, while a hybrid adaptive cosine exponential weigh dynamically adjusts the search intensity. Additionally, an inverse worst individual variant reinforcement approach is employed to enhance the population evolution efficiency. In the hybrid test sets of CEC2005 and CEC2019, MSCOA shows improved convergence accuracy compared to the traditional COA algorithm, and the Wilcoxon test ($p < 0.05$) confirms its superiority over five other comparison algorithms. MSCOA outperforms other algorithms in terms of robustness, convergence speed, and solution accuracy, although there is still room for further improvement. When combined with Extreme Learning Machine (ELM) and applied to the Wisconsin breast cancer dataset, the MSCOA-ELM model achieved 100% accuracy and F1 score, a 28.9% improvement over the baseline ELM, demonstrating the algorithm's efficiency and generalization ability in solving practical optimization problems.

Keywords Crayfish optimization algorithm, Chaotic initialization, T-distribution feeding strategy, Ternary optimization mechanism, ELM, Cancer prediction

Optimization is widely regarded as the most efficient method for determining the best solution to various real-world problems in complex domains. All scientific fields face the constant challenge of optimization, and hundreds of Metaheuristic Algorithms (MA) have been devised to handle the growing complexity of optimization issues. A wide class of algorithms called MA is used to tackle challenging optimization issues. Numerous domains, including engineering design[1], machine learning[2], the medical profession[3], community detection[4], atmospheric sciences[5], and others, heavily rely on these metaheuristic techniques. MAs are often categorized into many groups according to where they got their inspiration: evolutionary algorithms[6], swarm intelligence (SI)[7], physics-based algorithms[8], and human behavior-based algorithms[9]. Among them, SI is particularly popular in scientific research and practical applications due to the simplicity and efficiency of its mathematical model.

SI is a type of stochastic optimization algorithms that tackle complicated optimization issues by mimicking the collective behavioral traits of a natural collection of organisms, such as Spider wasp optimizer[10], Genghis khan shark optimizer[11], Puma optimizer[12], Dung beetle optimizer[13], Gazelle optimization algorithm[14], Grey wolf optimizer[15] and Frilled Lizard Optimization[16] etc. These algorithms are used in computer science[17], engineering science[18], Mathematics[19], Environmental Sciences[20], biomedical science[21] and many other fields have demonstrated powerful problem solving capabilities.

In recent years, the integration of metaheuristic algorithms and machine learning has achieved remarkable progress in various fields. Researchers have explored how metaheuristic optimization algorithms can enhance machine learning models. Jovanovic et al.[22] proposed a hybrid approach combining machine learning and

School of Information, Shanxi University of Finance and Economics, 030000 Taiyuan, China. email: pbxy66@163.com

swarm intelligence optimization to tackle the problem of credit card fraud detection. Han et al.[23] utilized a hybrid method based on Convolutional Neural Networks (CNN) and Extreme Learning Machine (ELM) for the optimal and efficient modeling of Proton-exchange Membrane Fuel Cells (PEMFC), further optimizing the model using an improved Honey Badger Algorithm (IHBA) to achieve optimal results. Ewees et al.[24] employed the Honey Badger Optimization (HBO) algorithm to optimize Long Short-Term Memory (LSTM) networks, boosting wind power forecasting performance. Additionally, Ma et al.[25] combined the Multi-Verse Optimization (MVO) algorithm with Support Vector Regression (SVR) for geohazard modeling, achieving great success. Jovanovic et al.[26] tuned the multi-headed LSTM structure using a modified Particle Swarm Optimization (PSO) algorithm to enhance fuel price forecasting accuracy. These studies highlight that the fusion of metaheuristic optimization and machine learning can optimize model structures and improve prediction and classification performance. In the future, this research domain holds great potential, particularly in intelligent scheduling optimization, automated hyperparameter tuning, and deep learning-driven adaptive search strategies, which could further strengthen the synergy between optimization algorithms and machine learning.

The new swarm intelligence algorithm Crayfish Optimization Algorithm (COA)[27] was put out by Jia et al. in 2023, inspired by crayfish behaviors such as burrow scrambling, foraging, and heat avoidance. COA mimics crayfish behaviors through a two-phase strategy: in the exploration phase, it mimics crayfish searching for habitats to enhance global search ability, while in the exploitation phase, it mimics burrow scrambling and foraging behaviors to achieve local optimization. The algorithm is dynamically adjusted based on temperature changes, with crayfish searching for burrows to avoid the heat when the temperature exceeds 30 °C and foraging when it falls below 30°C.

Motivations behind the present work

Although COA possesses numerous advantages, it also has several shortcomings:

- Decline in population diversity.
- Tendency to fall into local optima.
- Insufficient exploration capability.
- Lack of adaptability due to static parameters.

Contribution of this study

To efficiently address these limitations of COA, this study proposes a hybrid strategy enhanced crayfish optimization algorithm (MSCOA). To evaluate its performance, MSCOA is combined with Extreme Learning Machine (ELM) and applied to breast cancer prediction. The key contributions of this study are as follows:

- The chaotic inverse exploration initialization strategy is proposed to initialize the population position by introducing Logistic-Tent chaotic mapping with variational quasi-inverse learning to generate more diverse and uniformly distributed solutions and increase the algorithm's effectiveness for global search.
- The adaptive t-distributed feeding strategy is used to describe the relationship between feeding amount and temperature, which effectively increases the algorithm's potential to search locally and successfully expands the possibility for adaption in the simulated real-world environment.
- Curve growth acceleration factor C_1 is proposed to optimize burrow positioning, ensuring the integration of individual and global optimal solutions while making better use of population information.
- Hybrid adaptive cosine exponential weights ω are introduced into the crayfish position update formula, allowing the algorithm to dynamically adjust search intensity and balance global exploration and local convergence.
- Inverse worst individual variance strengthening strategy applies inverse perturbation to poor solutions, guiding the population out of local optima.
- A novel method, the adaptive ternary optimization mechanism, is proposed by integrating C_1, ω, and the inverse worst individual variance strengthening strategy into the position update process during the Summer Resort Stage.
- The MSCOA-ELM model is developed and applied to breast cancer prediction, demonstrating the algorithm's potential for practical applications.

To clearly highlight the main contributions of MSCOA, they are summarized in the following numbered list:

1. Introduction of a chaotic inverse exploration initialization strategy.
2. Implementation of an adaptive t-distributed feeding strategy.
3. Proposal of a curve growth acceleration factor for optimized burrow positioning.
4. Integration of hybrid adaptive cosine exponential weights in position updates.
5. Development of an inverse worst individual variance strengthening strategy.
6. Creation of a novel adaptive ternary optimization mechanism.
7. Application of MSCOA-ELM to breast cancer prediction, demonstrating real-world utility.

This paper's fundamental format is as follows: Sect. Introduction provides a succinct synopsis of the topics and issues discussed throughout the paper. Section Motivations behind the present work introduces previous research on COA improvements and its practical applications. Section Contribution of this study provides a comprehensive summary of COA. Section 4 proposes MSCOA, detailing the algorithmic flow and the specifics of the MSCOA algorithm. Section 5 presents the experimental results and conducts analysis and discussion. Section 6 integrates MSCOA with ELM (MSCOA-ELM) for breast cancer prediction. Finally, Sect. 7 summarizes the paper.

LITERATURE REVIEW

Owing to its quickness, flexibility, and adaptability, COA been applied across a wide range of fields. For instance, Hussam et al.[28] developed a hybrid COASaDE optimizer for handling difficult optimization issues and engineering design issues. In order to address FS issues, Shaymaa et al.[29] presented a successful IBCOA approach. Sait et al.[30] explored the application of COA in engineering design optimization with the assistance of Artificial Neural Network, and Shikoun et al.[31] offer a novel BinCOA for feature selection.

COA has also been used for PV parameter extraction[32], cybersecurity[33], PV parameter estimation[34], skidney tumor segmentation and classification[35], and friction behavior modeling in complex machinery[36].

Moreover, Yakubu et al.[37] used COA to generate an optimal resource allocation in both the fog and cloud layers that reduces the delay and execution time of the system. Jia et al.[38] used the COA approach to deftly find the crucial hyperparameters of the model. Jiang et al.[39] showcased the BiLSTM approach for time-series forecasting, optimized by the enhanced crayfish optimization algorithm (ICOA). Huang et al.[40] suggest an innovative fault diagnostic methodology for distribution transformers and COA is used to optimize parameters. Almutairi et al.[41] leveraged a hierarchical optimization strategy that integrates particle swarm optimization with crayfish optimization for more effective solutions.

Nevertheless, COA's performance in complex optimization tasks is hindered by challenges such as local optima, diminished population diversity, and insufficient exploration capacity during the optimization process.

To mitigate these issues, Zhong et al.[42] proposed HRCOA, which combines ROA's exploitation operators with COA's summer resort operators to streamline the algorithm. However, this method struggles with inefficiencies in high-dimensional problems. In an attempt to improve the algorithm's performance, Zhang et al.[43] presented ECOA, which introduces four enhancement measure, yet its convergence accuracy remains inadequate for complex multimodal functions. Jia et al.[44] presented MCOA, a mechanism for updating the environment based on the crayfish's survival habit. The MCOA uses a ghost-based antagonistic learning strategy along with the water quality factor to guide the crayfish in its search for better environments. However, this approach still has limitations in balancing global exploration and local exploitation. Finally, Wang et al.[45] suggested IMCOA to address the disconnect between global exploration and local exploitation phases in COA, particularly during the summer vacation and tournament phases, but it fails to completely eliminate premature convergence.

Crayfish optimization algorithm

Crayfish (scientific name: Procambarus clarkii) is a kind of freshwater crustacean[46] that is indigenous to North America, which has been brought to several regions of the world due to its adaptability. Temperature significantly impacts its survival[47], influencing behavior such as burrowing to avoid heat or foraging when conditions are favorable.

As omnivores, crayfish may eat a wide range of foods, including grains, insects, and plant material[48]. When Crayfish feed, they usually claw large food items, tear it off, and send it to the second and third walking feet for grasping and nibbling. For small amounts of food, their second and third walking feet are used to grab and bite directly. And crayfish typically utilize pincers or fast concealment as a defensive tactic to keep other crayfish from stealing them[49]. As seen in Fig. 1, The behavior of freshwater crayfish when foraging, avoiding summer heat, and competing is the inspiration's origin for the COA, which are applied in the algorithm's exploration and development phases.

Temperature and feeding

Changes in environmental temperature cause crayfish to exhibit different behavioral stages. Temperature is defined as Eq. (1). Within a suitable temperature range (20°C to 35°C), crayfish engage in foraging behavior and their feeding amount is regulated by temperature, showing an approximate normal distribution trend. These are the particular equations:

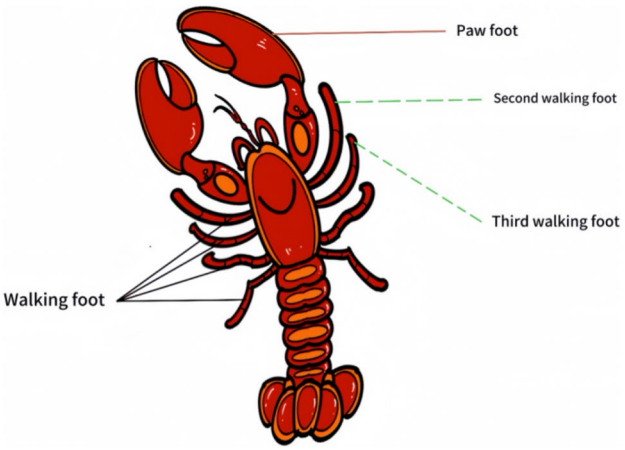


Fig. 1. Structure diagram of crayfish.

$$temp = rand \times 15 + 20 \tag{1}$$

$$P = C_1 \times \left(\frac{1}{\sqrt{2 \times \pi} \times \sigma} \times \exp\left(-\frac{(temp - \mu)^2}{2\sigma^2} \right) \right) \tag{2}$$

where μ indicates the most ideal outside temperature for crayfish foraging. By adjusting C_1 and σ, the quantity of food that crayfish eat at various temperatures can be controlled.

Summer resort stage

When external temperatures exceed 30 °C, crayfish instinctively retreat into their burrows to escape the heat. The expression for the burrow position is denoted as Eq. (3).

$$X_{shade} = (X_G + X_L)/2 \tag{3}$$

where X_L stands for the ideal spot in the existing population and X_G signifies the optimal position attained throughout the iterative process. The rivalry between crayfish for burrows is arbitrary. Equation (3) states that when rand <0.5, the crayfish straight to escape the heat because there aren't any other crawfish battle for it.

$$X_{i,j}^{t+1} = X_{i,j}^{t} + C_2 * rand * \left(\mathrm{X_{shade}} - X_{i,j}^{t} \right) \tag{4}$$

where t is the amount of cycles that are currently in progress, $X_{i,j}^{t}$ symbolizes the j-th dimensional position of the i-th crayfish in the t-th iteration, and C_2 is a coefficient that decreases linearly from 2 to 0. This is how it is calculated.

$$C_2 = 2 - (t/T) \tag{5}$$

Competition stage

The presence of rand >0.5 and an ambient temperature over 30 °C suggests that other crayfish are also considering the burrow. The crayfish now fight over who gets to keep the burrow. Every crayfish X will modify its location based on where another crayfish X is located. The formula is as follows.

$$X_{i,j}^{t+1} = X_{i,j}^{t} - X_{z,j}^{t} + \mathrm{X_{shade}} \tag{6}$$

where B is the crayfish population size and z denotes another randomly obtained crayfish individual with the following Eq.

$$z = round\,(rand * (B - l)) + l \tag{7}$$

Foraging stage

Crayfish will approach the food source if the outside temperature is at or below 30°C. When they locate the food, they assess it. If the food is too large, they will tear it apart using their claws and alternate between their second and third legs to feed. However, if the food is of moderate size, the crayfish will directly approach and consume it. The food's position is denoted as $\mathrm{X_{food}}$, which is represented as $\mathrm{X_{food}} = \mathrm{X_G}$. The food's size Q can be described by Eq. (8).

$$Q = C_3 \times rand \times fitness_i / fitness_{food} \tag{8}$$

where, $\mathrm{C_3}$ denotes the food factor, which is a constant value indicating the largest food size, with a specific value of 3, $\mathrm{fitness_i}$ denotes the fitness value of the i-th crayfish and $\mathrm{fitness_{food}}$ denotes the fitness value of the food.

The food is considered excessively large when the food size Q >($\mathrm{C_3}$+1)/2. At this time, the crayfish begins tearing it with the first claw. The following is the mathematical equation:

$$X_{food} = exp(-\frac{1}{Q}) * X_{food} \tag{9}$$

The food is alternately picked up and brought to the mouth using the second and third paws as it is shredded, becoming smaller. Equation (10) combines sine and cosine functions to simulate the alternating motion of the second and third claws.

$$X_{i,j}^{t+1} = X_{i,j}^{t} + X_{food} * p * (cos\,(2 * \pi * rand) - sin\,(2 * \pi * rand)) \tag{10}$$

When the food size $Q \leq (\mathrm{C_3} + 1)/2$, the crayfish simply travels towards the food position and feeds, as Eq. (11) illustrates.

$$X_{i,j}^{t+1} = \left(X_{i,j}^{t} - X_{food} \right) * p + p * rand * X_{i,j}^{t} \tag{11}$$

Hybrid strategy enhanced crayfish optimization algorithm

To solve the problem of COA, this paper proposes MSCOA. To address the issue of insufficient population diversity in COA, where traditional random initialization leads to uneven solution distribution and low global search efficiency, this study proposes a Chaotic Reverse Initialization Strategy. To overcome the fixed step size in the COA feeding phase, which limits local search capability and makes static parameters less adaptable to dynamic optimization environments, this paper introduces the t-Distribution Feeding Strategy. To tackle the problem of excessive reliance on the current best solution, which can easily lead to local optima and premature convergence, we propose the Inverse Elite Variant Reinforcement Strategy. Lastly, to address the lack of a dynamic weight mechanism in COA, which hinders the flexible adjustment of search intensity, this paper introduces the Curve Growth Bootstrap Factor and Hybrid Adaptive Cosine Exponential Weighting.

Chaotic reverse initialization strategy

In intelligent optimization algorithms, the quality of the original population has a non-negligible effect on the algorithm's performance, and can directly impact both the convergence rate and global search capability[50]. The random initialization adopted by the basic COA algorithm may lead to the distribution of the initial population being too scattered or concentrated, which lacks effective guidance of the search space. An inadequately distributed population may prematurely converge to local optima, reducing the overall search effectiveness. To address this, a chaotic inverse initialization strategy is adopted to optimize the population initialization process. This is achieved by incorporating Logistic-Tent chaotic mapping along with quasi-inverse learning, enhancing the algorithm's global search efficiency.

Logistic-Tent chaotic mapping Good randomness, traversability, and non-repetition characterize chaotic motion. Chaotic mapping has been a popular technique in optimization algorithms in recent years. The algorithm's performance may be further enhanced by incorporating chaotic behavior, which can further increase the population variety and global search efficiency[51].

Logistic mapping is a kind of single-peaked mapping, which fully carries over the benefits of unpredictability and sensitivity to beginning values that are present in chaotic systems. This is how it is expressed mathematically:

$$X_{n+1} = a\times X_n(1 - X_n) \tag{12}$$

where X_n is the outcome of the iteration $X_n \in (0,1)$, and a is a parameter $a \in (0,4)$.

The Tent chaotic system, another one-dimensional mapping, is known for its uniform probability distribution and strong autocorrelation. Its mathematical form is as follows:

$$X_{n+1} = \begin{cases} b\times X_n\,, & 0 \le X_n < 0.5 \\ b\times (1 - X_n)\,, & 0.5 < X_n \le 1 \end{cases} \tag{13}$$

where X_n is the iteration result $X_n \in (0,1)$, and a is a control parameter $b \in (0,4)$ and it is $b \neq 0.5$ when the system is chaotic.

By combining these two chaotic systems, Zhu et al.[52] produced a Logistic-Tent composite chaotic system. The benefits of both Tent and Logistic mapping are combined in the Logistic-Tent chaotic mapping, which promotes the stability and diversity of chaotic behavior. diversity of chaotic behavior. It is capable of generating highly complex and unpredictable sequences with good traversal properties. Here is the Logistic-Tent mapping expressed mathematically:

$$X_{n+1} = \begin{cases} (k\times X_n(1 - X_n) + (4 - r)X_n/2)\ mod\ 1, & X_n < 0.5 \\ (k\times X_n(1 - X_n) + (4 - r)(1 - X_n)/2)\ mod\ 1, & X_n \ge 0.5 \end{cases} \tag{14}$$

where X_n is the iteration result $X_n \in (0,1)$, and k is a control parameter $k \in (0,4)$.

In Fig. 2, the histograms of these three chaotic mappings after 500 iterations are shown, where it is observed that the Logistic-Tent mapping has a more uniform distribution between [0,1] than the other two typical mappings. This means that using Logistic-Tent mapping for population initialization can more thoroughly cover the search space and raise the likelihood of identifying the global optimum. Secondly, Lyapunov exponent is an important measure of chaotic performance, and this index is maximized to 1.2937 for Logistic-Tent cascade

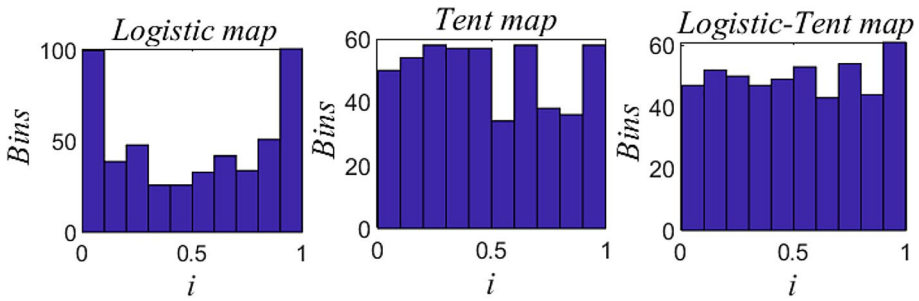


Fig. 2. Three kinds of chaotic mapping histograms.

chaotic sequences, compared to 0.6926 for Tent chaotic sequences and 0.6930 for Logistic chaotic sequences[53]. Obviously Logistic-Tent chaotic mapping has better chaotic properties.

Quasi-reverse learning

Compared with independent randomly generated candidate solutions, the reverse solution is beneficial and more conducive to the propensity optimal solution[54]. Therefore, in this work, the chaotic initialized population obtained by mapping the Logistic-Tent chaotic sequence into the solution space is subjected to quasi-reverse learning to further optimize the initial population and enhance the likelihood of receiving an excellent initial solution.

Opposition-based learning (OBL) was established by Tizhoosh[55], generating opposite individuals in the current individual region, comparing and selecting the individuals with high adaptability for subsequent iterations, which can effectively enhance the population's quality and variety, and is conducive to optimizing its search effect. The following is the mathematical formula for reverse learning:

$$X_{OBL}(n+1) = LB + UB - X(n) \tag{15}$$

where LB and UB are the bottom and upper boundaries of the search space, respectively.

Quasi-opposition-based learning (QOBL) is one kind of OBL[56], and previous studies have demonstrated that using quasi-opposition-based solutions is more effective than inverse solutions in finding the global optimum. Quasi-opposition-based learning obtains the function of the population as follows:

$$X_{QOBL}(n+1) = \begin{cases} CS + rand(0,1) \times (MP - CS), & if\ MP > CS \\ MP + rand(0,1) \times (CS - MP), & otherwise \end{cases} \tag{16}$$

where $X(n)$ is the chaotic initialized population after the Logistic-Tent chaotic mapping of Eq. (13), and LB, UB represent the chaotic initialized population's top and lower boundaries, respectively. $X_{QOBL}(n)$ is the new population produced via quasi-reverse learning. MP is illustrated in Eq. (17).

$$MP = LB + UB - X(n) \tag{17}$$

CS as shown in equation Eq. (18).

$$CS = \frac{LB + UB}{2} \tag{18}$$

Flowchart of chaotic inverse exploration initialization strategy

As shown in Fig. 3, this strategy first generates a chaotic initialized population by initializing the population with Logistic-Tent chaotic mapping, and then generates its inverse solution using quasi-reverse learning, which can explore various areas of the issue space, avoiding the over-concentration of individuals in certain regions during the initialization, and reducing the likelihood of discovering the local optimum solution process at an early stage. Finally, the chaotic initialized population is merged with the inverse population, the fitness is calculated, and the optimal n individuals are chosen to become the new initial population. It can make the inverse solution complementary to the original solution and improve the coverage of the population, thus strengthening the optimization algorithm's early-stage exploration capability.

t-Distribution feeding strategy

An adaptive t-distributed feeding strategy was employed to define the connection between feeding behavior and temperature. This approach increased population variety and enhanced the algorithm's local search effectiveness.

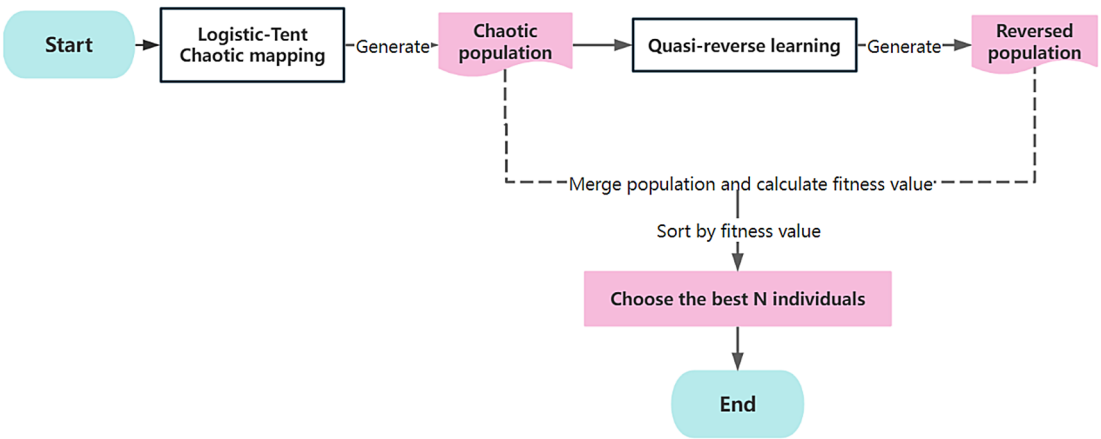


Fig. 3. Flow chart of chaotic reverse initialization strategy.

The t-distribution, sometimes referred to as the student distribution, is introduced in this work. Equation (19) shows its probability density function, which has the degree of freedom parameter n. This study uses Eq. (19) to illustrate the link between the temperature and the feeding quantity. The t-distribution combines the features of the Gaussian and Cauchy distributions. Figure 4 compares the pictures of the three distributions.

$$f(x) = \frac{\tau\left(\frac{n+1}{2}\right)}{\sqrt{n \times \pi} \times \tau\left(\frac{n}{2}\right)} \times \left(1 + \frac{x^2}{n}\right)^{-\frac{n+1}{2}}, -\infty < x < +\infty \tag{19}$$

where τ is the Gamma function.

From Fig. 4, It is evident that early in the t-distribution iteration, the properties of the Cauchy distribution will be displayed, enhancing the population's variety. At the middle and late stage of iteration, the value of the degree of freedom parameter is larger. Compared with the Gaussian distribution in the original text, the t-distribution approximates the Gaussian distribution, which increases the algorithm's ability to evolve locally. And the t-distribution has a longer tail relative to the Gaussian distribution, which means it can better capture extreme values or outliers. For modeling data such as crayfish intake, which comes with a high degree of volatility, the t-distribution can be a more realistic reflection of the actual situation. Therefore, using an adaptive t-distribution feeding strategy maintains the algorithm's stability and enhances the population's variety.

Curve growth guidance factor

Curve growth acceleration factor is used to control the influence of the global and individual optimal solutions, thus raising the likelihood of discovering the globally best solution and enhancing search efficiency.

X_{shade} is an important guide to the evolution of crayfish populations. When the quantity of iterations rises, many particles may become trapped in local optima. To preserve population variety and enhance the algorithm's capacity to escape these local optima, the global optimum's effect on the solution search process must be increased. This adjustment helps make the search more stable and comprehensive.

To achieve this, we introduce two parameters, C_1 and C_2, as outlined in Eqs. (20) and (21), which respectively control the influence of the global and individual optimal solutions. These parameters allow for varying weight distributions at different stages of the iteration, increasing the search's variety and unpredictability. The algorithm is able to avoid stagnating in local optimum solutions because to this flexibility.

$$C_1 = a \times \log\left((bx + c)/T\right) + d \times e^{(f*t+g)/T} \tag{20}$$

$$C_2 = 1 - C_1 \tag{21}$$

Updated formula for the location of X_{shade} updated to formula 23. By dynamically adjusting C_1 and C_2, the algorithm can strike a balance finding novel answers with taking use of already-found ones. Specifically, as C_1 is incremented, C_2 is decremented accordingly (Fig. 5 (a)), and this change ensures that information about the global and individual optimal solutions is combined to more fully utilize the information about the current population, thus improving the search efficiency and raising the likelihood of discovering the globally ideal answer. When the quantity of iterations rises, the values of C_1 and C_2 are automatically adjusted in the process, allowing the algorithm to focus on extensive exploration at the beginning and progressively shift to localized and refined development in the later stages. This dynamic adjustment mechanism makes a difference to boost the algorithm's convergence speed and global search capability.

In order to make C_1, C_2 meet the requirements, after experiments, the parameters of Eq. (20) are set as a = 0.11; b = 1; c = 1; d = 0.11; f = 1; g = 1. As shown in Fig. 5 (b), Sensitivity analysis shows that:

- when a = d = 0.11, the C_1 curve exhibits a smooth transition, balancing exploration and exploitation, which leads to stable and effective search performance.

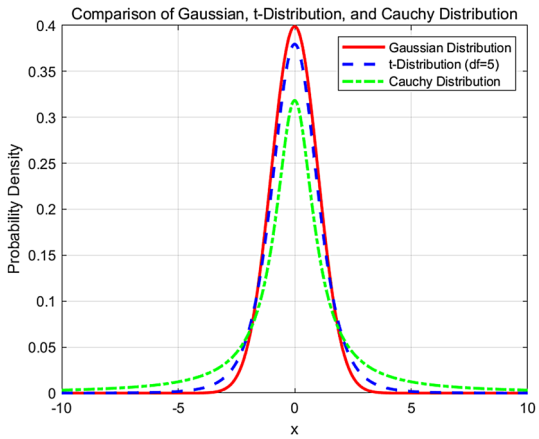


Fig. 4. Image comparison of T-distribution, Cauchy distribution, Gaussian distribution.

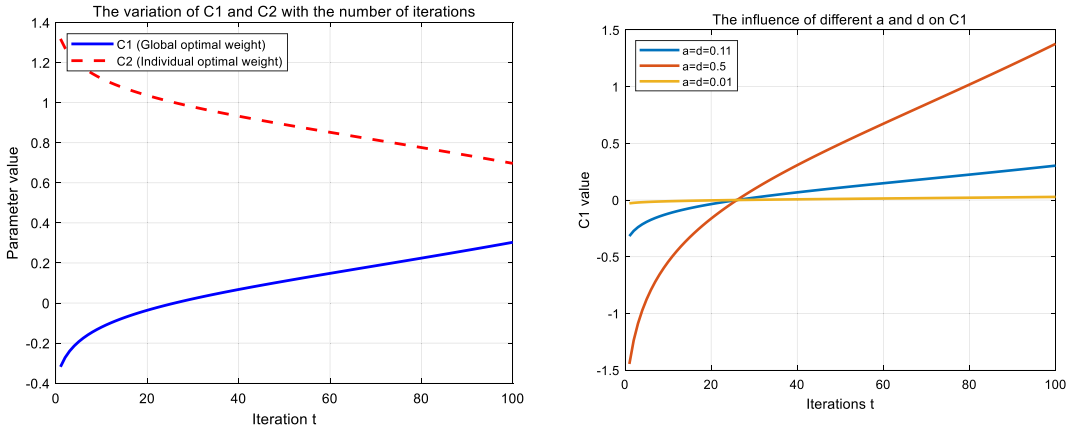


Fig. 5. Image comparison of T-distribution, Cauchy distribution, Gaussian distribution.

- When a = d = 0.5, rises too early, leading to premature convergence to local optima, reducing global exploration capacity.
- When a = d = 0.01, C_1 increases slowly, resulting in low global search efficiency, thereby lowering the convergence efficiency.
- Setting b = c = f = g = 1 simplifies the formula structure and reduces the complexity of parameter tuning, making the algorithm easier to implement while maintaining robust performance.

Experimental verification shows that setting a = d = 0.11 and b = c = f = g = 1 achieves the best balance between exploration and exploitation.

$$X_{shade} = C_1{*}X_G + C_2{*}X_L \tag{22}$$

The dynamically weighted three-step displacement updating strategy updates the equation for the summer avoidance phase, in which crayfish enter the burrow directly, to Eq. (23) through three improvements in the curve growth bootstrap factor, the hybrid adaptive cosine exponential weighting, and the inverse elite variant reinforcement strategy.

$$X_{i+1} = \omega * (C1{*}r_1{*}\cos(r_2) * (X_{worst} - X_i) + C2{*}r_2{*}\sin(r_2) * (X_{shade} - X_i)) \tag{23}$$

where ω is an innovative hybrid adaptive cosine exponential weight suggested in this paper as in Eq. (24). The figure below, Fig. 5, displays the amount of iterations.

$$\omega = \frac{cos\left(\frac{\pi}{2} \bullet \left(1 - \frac{t}{T}\right)\right)}{1 + exp\,(a - bt)} \tag{24}$$

where a = 5 and b = 0.1. As shown in Fig. 6 (a), if a is too small, the initial ω becomes too large, resulting in insufficient local search at the early stage, which negatively impacts optimization quality. Conversely, if a is too large, the initial ω is smaller, but since the denominator's decay rate remains the same, ω grows more slowly. While this improves early-stage accuracy, it weakens global search capability, making the algorithm prone to local optima. Additionally, Fig. 6 (b) shows that if b is too small, the denominator decays too slowly, causing ω to remain at a low value for an extended period, leading to weak global search ability and slow convergence. On the other hand, if b is too large, the denominator decreases rapidly, causing ω to reach its maximum too early. This results in premature convergence, preventing sufficient local optimization. Through the above comparisons, the combination of a = 5 and b = 0.1 provides adequate local search time in the early phase while ensuring a well-timed transition to global search. This prevents both premature convergence and prolonged stagnation. Therefore, this parameter selection is theoretically and experimentally justified.

As illustrated in Fig. 7, the weight factor exhibits a positive correlation with the number of iterations, t, indicating a nonlinear growth trend. Initially, there is a slow change both in the initial and final phases of the incremental procedure. when the algorithm first starts, the inertia weight is kept at a lower value for an extended period, which strengthens local optimization capability, thereby increasing the accuracy. In the intermediate and advanced phases, when the crayfish population exhibits higher convergence, the inertia weight increases with the number of iterations, stabilizing at a larger value for an extended duration. This adjustment fosters the global optimization potential of the crayfish population, improving the speed.

The design of the inertia weight integrates both cosine and exponential functions, which allows the algorithm to dynamically adjust its search intensity. This ensures a balanced approach, accommodating the global exploration needs as well as the local convergence requirements, effectively improving overall performance.

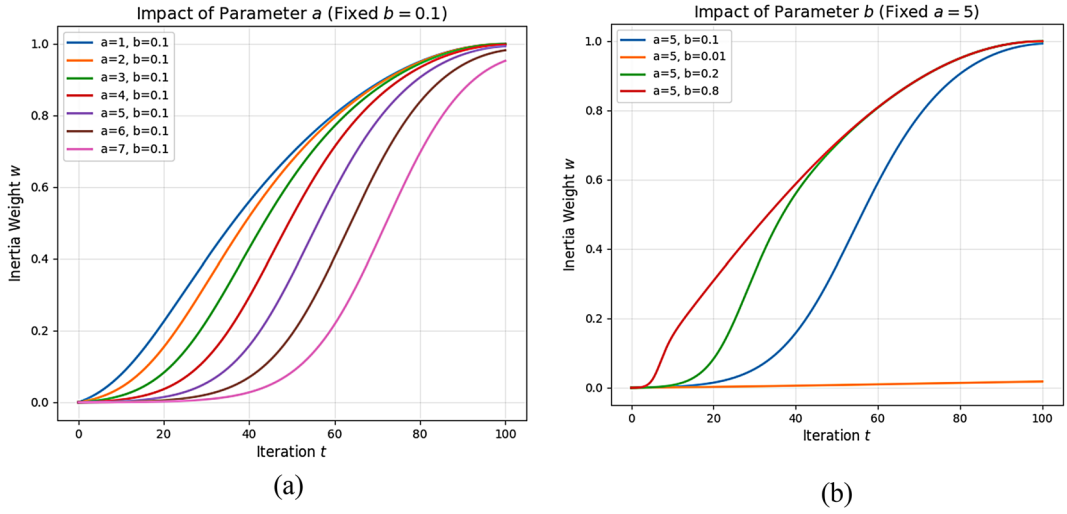


Fig. 6. Image of parameters a, b selected.

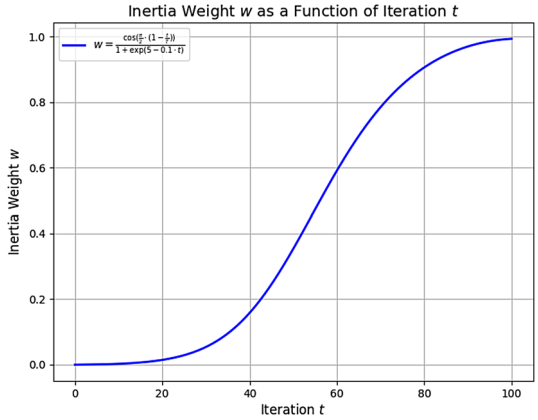


Fig. 7. Image of ω .

X_{worst} is the position after applying the Inverse Worst Elite Mutation Strategy update. In this paper, X_{shade} i.e., the global ideal solution as well as the local ideal solution, has been fully utilized, while the worst position is the neglected information. For the purpose of better utilizing the information provided by X_{worst}, this paper proposes the Reverse Elite Mutation Reinforcement Strategy. After the worst individual $\mathrm{X_{worst}}$ is selected, it is firstly subjected to quasi-reverse learning, through which the effective information in the individual at the worst position is maximized and the reverse elite solution is generated to guide the search process to approach best possible answer. Second, a nonlinear Cauchy-Gaussian variation strategy is applied to the worst position X_{worst}. This method introduces random perturbations to the globally optimal individuals, preventing local optima from being reached by the algorithm. In this paper, based on the Cauchy-Gaussian variation strategy, we introduce hybrid adaptive cosine exponential weights ω to perturb the generated inverse elite solution with nonlinear Cauchy-Gaussian variation as in Eq. (25). Pre-iteration population is more dispersed, given a larger weight of the Cauchy distribution, is conducive to generating a large step size jumping out of the current position, to facilitate the search for better solutions. Late iteration of population is more aggregated, given a larger weight of the Gaussian distribution, is conducive to generating a small step size in the current position. It is beneficial to generate small step length to carry out local perturbation at the current position to avoid local optima and improve the algorithm's convergence accuracy.

$$X_{oworst} = X_{oworst1} * ((1-\omega) * Cauchy + \omega * Gaussian) \tag{25}$$

where, X_{oworst} is the individual of the reverse elite solution perturbed by the nonlinear Cauchy-Gaussian variational strategy and $X_{oworst1}$ is the reverse elite solution after quasi-reverse learning. A random variable with a Gaussian distribution is referred to as Gaussian, while a random variable with a Cauchy distribution is referred to as Cauchy.

After solving the perturbation for the reverse elites, the greedy strategy is finally performed. Comparing the individuals before and after the mutation, the one with better adaptation is selected as the new individual. The selection of the optimal solution based on the greedy strategy is shown in Eq. (26).

$$X_{worst} = \begin{cases} X_{oworst} & if\ fobj\left(X_{oworst}\right) < fobj\left(X_{oworst1}\right) \\ X_{oworst1} & otherwise \end{cases} \tag{26}$$

where, $fobj\left(\right)$ is the fitness function.

The variables r_1 and r_2 represent random numbers within the range [0,1], which enhance individual randomness, disrupt directional tendencies, and introduce greater evolutionary possibilities. By incorporating sine and cosine functions, individual positions are updated through various methods, effectively preventing premature convergence and preserving population diversity. The resilience and stability of the algorithm are greatly increased by this approach, thereby enabling a more balanced process of prospecting and extraction.

In the first phases of the algorithm, X_{shade} representing the global and local optimal solutions, is used to guide the overall position towards a balanced state. At a later stage, the reverse elite solution of X_{worst}, there may be unexplored regions, and in order to make the value more comprehensive, the proportion of reverse elite solution of X_{worst} is gradually increased. Therefore, the formula introduces C_1, C_2, as in Eqs. (20) and (21). $C1$ is gradually increased and C_2 is taken to be gradually decreased.

MSCOA pseudo-code

Algorithm 1 Pseudo-code of the MSCOA.

Initialization using Eq. (14) and Eq. (16) to generate an initial population.
Calculate the fitness value of the population to obtain $\mathrm{X_G}$, $\mathrm{X_L}$, $\mathrm{X_{worst}}$
While t < T.
Defining temperature temp using Eq. (1)
Calculate the value of $\mathrm{C_1}, \mathrm{C_2}, \omega$,
Update $\mathrm{X_{worst}}$ using Eq. (17)
for i = 1 to N.
If temp > 30.
Define cave $\mathrm{X_{shade}}$ using Eq. (23)
If rand < 0.5.
Perform Summer Resort Stage.
Calculate $\mathrm{X_{oworst}}$ using Eq. (26)
Update $\mathrm{X_{worst}}$ according to Eq. (27)
Crayfish enters the summer resort stage based on Eq. (24)
Else

Perform Competition Stage.

Crayfish competes for caves using Eq. (7)

End.
Else.
Perform Foraging Stage.
Obtain food intake p and food size Q from Eq. (18) and Eq. (9)
If Q > 2

Crayfish shreds food using Eq. (10)

Crayfish forages based on Eq. (11)

Else

Crayfish forages based on Eq. (12)

End.
End.
Update fitness values, $\mathrm{X_G}$, $\mathrm{X_L}$, $\mathrm{X_{worst}}$
t = t + 1.
End.

MSCOA flowchart

In Fig. 8, the workflow of MSCOA is shown. First, population initialization is performed, using Eq. (14) and Eq. (16) to set the crayfish population, calculate the fitness value, and obtain the global optimum $\mathrm{X_G}$, local optimum $\mathrm{X_L}$, and global worst position $\mathrm{X_{worst}}$. Next, Eq. (16) is used for updating global worst position $\mathrm{X_{worst}}$, and Eq. (1) is used to set the temperature temp, and to determine whether to enter into a high temperature environment (i.e., Temp > 30). When the temperature exceeds 30 degrees, the algorithm either proceeds to the summer resort or the competition stage, after defining $\mathrm{X_{shade}}$ through Eq. (22). In the case where rand < 0.5, the algorithm simulates the crayfish's behavior in a high-temperature environment, entering the summer resort stage. Equation (25) is used to calculate $\mathrm{X_{oworst}}$ and Eq. (26) is used to update $\mathrm{X_{worst}}$ Eq. (23) is used to adjust the position of crayfish. On the contrary, if rand > 0.5, crayfish will move to the competition stage and compete for burrows for better survival. This process is carried out by Eq. (6) for competition for burrows.

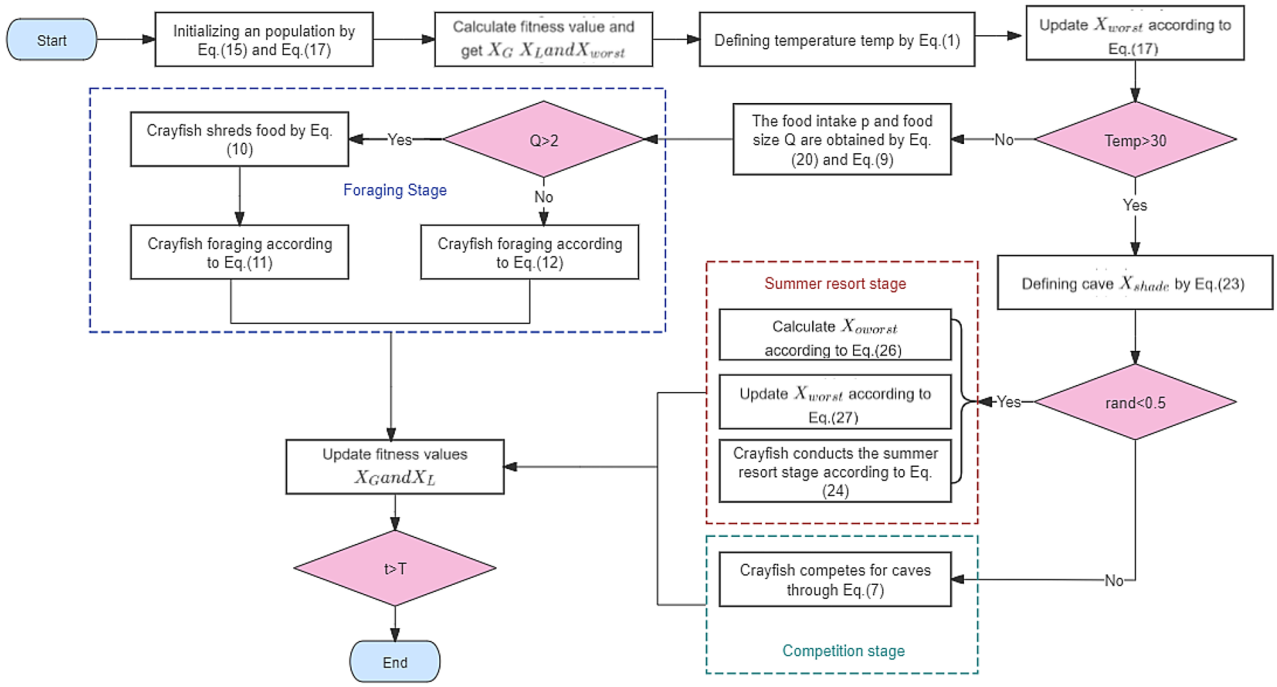


Fig. 8. Flow chart of MSCOA.

When not exceeding 30 degrees, the food size Q is computed using Eq. (8), and the food intake p is calculated by Eq. (19), and based on the size of Q, the decision of which foraging behavior to take is made. If Q > 2, the crayfish shredded the food according to Eq. (9), followed by foraging behavior according to Eq. (10). When Q < 2, Eq. (11) is applied to perform the foraging behavior. In each round of iteration, the algorithm updates the global and local optimal solution X_G and X_L, and gradually approximates the optimal solution based on the fitness value. The whole algorithm ends after the termination condition t > T is satisfied.

Compared with traditional COA, MSCOA introduces additional mechanisms to enhance the diversity of search and global exploration, avoiding trapping in local optimal solutions and improving performance in challenging optimization tasks.

Computational complexity analysis

The time complexity illustrates the algorithm's convergence rate. Let T represent the number of iterations, n the population size, and dim the search space's dimension. In the original COA, initialization complexity is $O(n * dim)$, with each iteration being $O(T * n * dim)$, leading to an overall complexity of $O(T * n * dim)$.

In MSCOA, after applying chaotic inverse exploration during initialization, the complexity becomes: $O(n * dim + n * log(n))$. The most significant part is $O(n * dim)$. In the iterative process, despite introducing a dynamically weighted three-step displacement updating strategy, the time complexity remains consistent with COA at $O(T * n * dim)$.

Thus, MSCOA shares the same asymptotic time complexity as COA, $O(T*n*dim)$, with added computations from chaotic reverse initialization slightly impacting constants without altering the overall order.

Experimental Results and Discussions.

Three primary experiments are conducted in this part. The first main experiment is to test the enhanced strategy's efficacy introduced in the algorithm, the second is evaluating MSCOA's performance against a number of similarly comparable algorithms. The third one compares the suggested algorithm against other algorithms from the second experiment using a rank-sum method. This test may be used to determine whether MSCOA and other algorithms differ noticeably from one another. Additionally, the fourth experiment utilizes the Friedman test to assess the overall performance of these comparative algorithms.

A. ***Experimental setup***.

To thoroughly examine the optimization accuracy and conversion rate of the proposed algorithm, 17 standard test functions have been selected from the CEC2005 and CEC2019 test sets for comparative experiments, aiming to comprehensively analyze the performance of the optimization algorithm. The function names, expressions, search ranges, and operational solutions are shown in Table 1. The dataset is appropriate for assessing an algorithm's performance in terms of local search, global search, resilience, and convergence ability since it contains a variety of optimization problem types, including unimodal, multimodal, hybrid, and high-dimensional issues.

Among them, F1 through F6 are unimodal functions, meaning they have a single global optimum. These functions are appropriate for evaluating the algorithm's capacity for local searches and rate of convergence. Multimodal functions, such as F7–F12, contain several local optima but only one global optimum. These

Name	Function	Dim	Search Range	Optimal Solution
Sphere	$f_1(x) = \sum_{i=1}^{n} x_i^2$	30	[−100,100]	0
Schwefel 2.22	$f_2(x) = \sum_{i=1}^{n} \lvert x_i \rvert + \prod_{i=1}^{n} \lvert x_i \rvert$ $f_3(x) = \sum_{i=1}^{n} \left(\sum_{j=1}^{i} x_j \right)$	30	[−10,10]	0
Schwefel 1.2		30	[−100,100]	0
Schwefel 2.21	$f_4(x) = max_i \{ \lvert x_i \rvert, 1 \le i \le n \}$	30	[−100,100]	0
Hilbert	$f_5(x) = \sum_{1=1}^{n} \sum_{j=1}^{n} \frac{1}{i+j-1} x_i x_j$	16	[−16384,1638]	1
Chebyshev	$f_6(x) = \sum_{i=1}^{n} Ti(x_i)$ $f_{12}(x) = \sum_{i=1}^{n} \left[x_i^2 - 10\cos(2\pi x_i) + 10 \right]$	9	[−8192,8192]	1
Quartic	$f_7(x) = \sum_{i=1}^{n} i x_i^4 + random[0, 1)$	30	[−1.28,1.28]	0
Rastrigin		30	[−5.12,5.12]	0
Ackley	$f_9(x) = -20exp\left(-0.2\sqrt{\frac{1}{n}\sum_{i=1}^{n} x_i^2} \right) - exp\left(\frac{1}{n}\sum_{i=1}^{n} cos 2\pi x_i \right)$	30	[−32,32]	0
Griewank	$f_{10}(x) = \frac{1}{4000}\sum_{i=1}^{n} x_i^2 - \prod_{i=1}^{n} \cos\left(\frac{x_i}{\sqrt{i}} \right) + 1$	30	[−600,600]	0
Lennard-Jones	$f_{11}(x) = \sum_{i<j} \left(\frac{1}{\gamma_{ij}^{12}} - \frac{2}{r_{ij}^{6}} \right)$	18	[−4,4]	1
Rastrigin	$f_{12}(x) = \sum_{l=1}^{n} \left[x_i^2 - 10\cos(2\pi x_1) + 10 \right]$	10	[−100,100]	1
Six-Hump Camel	$f_{13}(x_1, x_2) = 4x_1^2 - 2.1x_1^4 + \frac{x_1^6}{3} + x_1 x_2 - 4x_2^2 + 4x_2^4$	2	[−5,5]	−1.0316285
Goldstein-Price	$f_{14}(x) = \left[1 + (x_1 + x_2 + 1)^2 \left(19 - 14x_1 + 3x_1^2 - 14x_2 + 6x_1 x_2 + 3x_2^2 \right) \right] \times \left[30 + (2x_1 - 3x_2)^2 \left(18 - 32x_1 + 12x_1^2 + 48x_2 - 36x_1 x_2 + 27x_2^2 \right) \right]$	2	[−2,2]	3
Ackley	$f_{15}(x) = -20\exp\left(-0.2\sqrt{\frac{1}{n}\sum_{i=1}^{n} x_i^2} \right) - \exp\left(\frac{1}{n}\sum_{i=1}^{n} \cos(2\pi x_i) \right) + 20 + e$	10	[−100,100]	1
Hartmann 3D	$f_{16}(x) = -\sum_{i=1}^{4} c_i exp\left[-\sum_{j=1}^{4} a_{ij} (x_j - p_{ij})^2 \right]$	4	[0,1]	−3.86
Shekel	$f_{17}(x) = -\Sigma_{i=1}^{10} \left[(x - a_i)(x - a_1)^T + c_i \right]^{-1}$	4	[0,10]	−10

Table 1. Benchmark function.

functions are primarily used to assess the algorithm's capacity for both global search and escape from local optima. The hybrid composite functions F13–F15 have a very complicated search space and incorporate several distinct mathematical functions. The main purpose of these functions is to evaluate the optimization algorithm's resilience and capacity for global search. High-dimensional multimodal functions, such as F16 and F17, have increased dimensionality and several local optima. They are employed to evaluate the algorithm's capacity for exploration and exploitation abilities in high-dimensional complex environments.

Different categories of test functions are used to evaluate the optimization algorithm's performance in terms of convergence speed, global search ability, ability to avoid local optima, robustness, and high-dimensional optimization capability. To obtain unbiased results, in all comparisons, the overall size and maximum number of iterations for all functions are set to 30 and 1000, respectively.

All experiments were run on a PC equipped with an Intel (R) Core (TM) i5-12500 H CPU, 16.00 GB RAM, and Windows 11 operating system, and all algorithms were implemented using MATLAB R2022a.

Comparison with different improvement strategies

MSCOA is compared with the crayfish optimization algorithm (COA1) that only applies the chaotic inverse exploration initialization strategy, the crayfish optimization algorithm (COA2) that only employs the adaptive t-distribution feeding strategy, the crayfish optimization algorithm (COA3) that only employs the curve-growth bootstrap factor, and the crayfish optimization algorithm (COA4) that only implements the dynamically-weighted three-step displacement update strategy, and the.

parameter settings are as the same as in this paper, to verify the superiority of MSCOA. To better illustrate the impact of each single strategy on the COA more clearly, the convergence curves of various algorithms on the functions F1-F17 are given in Fig. 9.

In Fig. 9, F1–F6 demonstrate that each strategy enhances convergence speed to varying degrees, with MSCOA consistently achieving faster convergence to lower fitness values compared to the original COA and its variants. This indicates that MSCOA excels in local search capabilities, allowing for more efficient and effective optimization. From F7–F12, it is evident that each strategy contributes to the superior performance of MSCOA, enhancing its exploration ability and enabling it to tackle complex optimization problems more effectively. From F13–F15, it can be observed that every strategy, to some extent, strengthens MSCOA's global search capability, endowing it with better robustness and improved global optimization performance. Finally, from F16 and F17, it is clear that MSCOA also demonstrates strong exploration and exploitation abilities in high-dimensional complex environments.

Taken together, the algorithms applying the various improvement strategies show different degrees of performance improvement. The MSCOA algorithm incorporating the various improvement strategies has enhanced optimization search and solution capabilities. It provides better robustness in complex optimization problems, thus potentially avoiding premature convergence.

Comparison between different novel intelligent algorithms

MSCOA is compared against COA[22], GWO[15], Subtraction-Average-Based Optimizer (SABO)[57], Dung Beetle Optimizer (DBO)[13], Wind-Driven Optimization (WWPA)[58], Chernobyl Disaster Optimizer (CDO)[59], Lungs Performance-Based Optimization (LPO)[60] and Artificial Protozoa Optimizer (APO)[61] under the same parameter settings: a population size of 30, a spatial dimension of 20, and a maximum iteration count of 1000. (Many optimization algorithms require a sufficient number of iterations to gradually converge to the optimal or near-optimal solution. If the number of iterations is too low, the algorithm may not have enough time to explore the search space thoroughly, leading to suboptimal solutions. Conversely, an iteration count of 1000 generally provides most optimization algorithms with enough time to perform an adequate search and evolution process, increasing the likelihood of convergence to a satisfactory solution. Moreover, excessively increasing the number of iterations significantly raises computational time and resource consumption. For complex optimization problems with large population sizes, excessive iterations can make computations infeasible in practical applications due to prolonged execution times. A maximum iteration count of 1000 strikes a balance between convergence performance and computational cost. It ensures that the algorithm has sufficient iterations to achieve good convergence behavior without making the computational time excessively long. In the field of optimization algorithm research, many studies and related literature frequently use around 1000 iterations to evaluate algorithm performance. This standardization allows for a consistent and comparable assessment across different algorithms, enabling researchers to fairly evaluate the performance of new algorithms against existing ones). To objectively assess optimization performance, these parameter values were used for 30 different executions of each method. The optimal value, average value, and standard deviation for each algorithm were recorded. A lower average value indicates higher optimization accuracy, but a lower standard deviation signifies more stability. The test results are illustrated in Table 4, with the best outcomes are indicated by bold numerals. Figure 10 provides a comparative visualization of fitness curves across algorithms for functions f1 to f17, and Fig. 11 displays the corresponding box plots.

In Table 2, the results of running MSCOA and various basic algorithms on functions F1-F17 are shown. Table 2 indicates that MSCOA can determine the optimal value on the 5 tested functions F1, F2, F3, F4, F5, F8, F10, F14 and F16 for dimension dim = 30. For the functions F6, F7, F11 and F17, the optimization results are quite near to the ideal value even if MSCOA is unable to achieve the theoretical optimal value. Meanwhile, comparative analysis of the standard deviations (std) across functions F1-F17 reveals that MSCOA demonstrates significantly smaller deviations in the majority of cases. This shows that the MSCOA has strong convergence stability and good robustness.

Figure 10 shows that MSCOA exhibits strong local convergence speed in unimodal functions (f1-f6), quickly reaching the optimal value. In bimodal functions (f7-f12), MSCOA shows strong global search ability, successfully

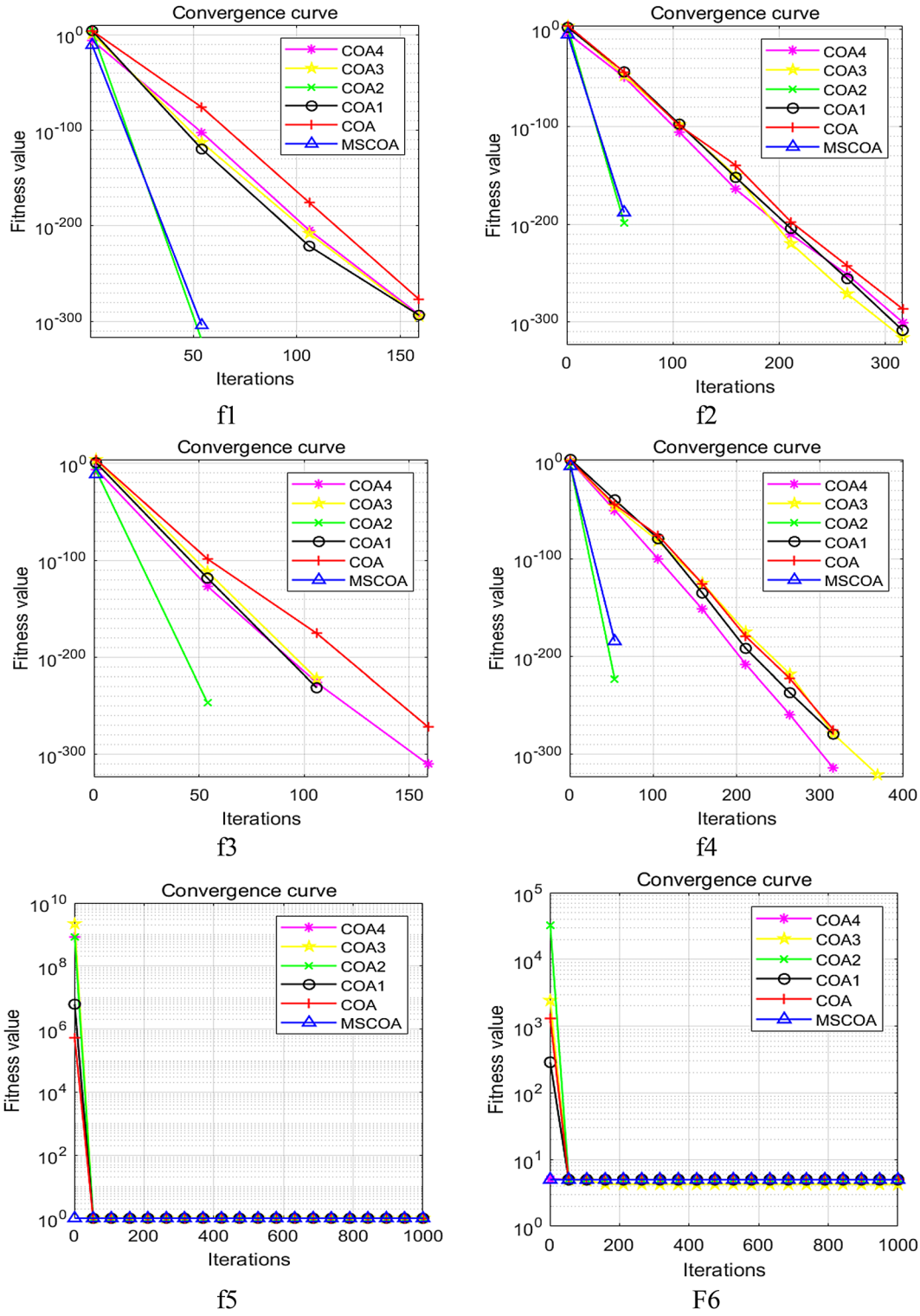


Fig. 9. Convergence curves of various algorithms.

avoiding local optimization. For functions f13-f15, MSCOA demonstrates good adaptability, robustness, and strong global search capability. In high-dimensional multimodal functions, MSCOA still performs excellently in terms of search and exploration ability, even when facing complex situations. Overall, MSCOA performs better in searching, but there is still room for improvement in complex functions and high-dimensional spaces.

It is evident from Fig. 11 that MSCOA has superior overall performance over multiple experiments compared to the other algorithms, usually finding better solutions, more focused results and more stable performance. It clearly outperforms other algorithms with higher median, high fluctuations and outliers.

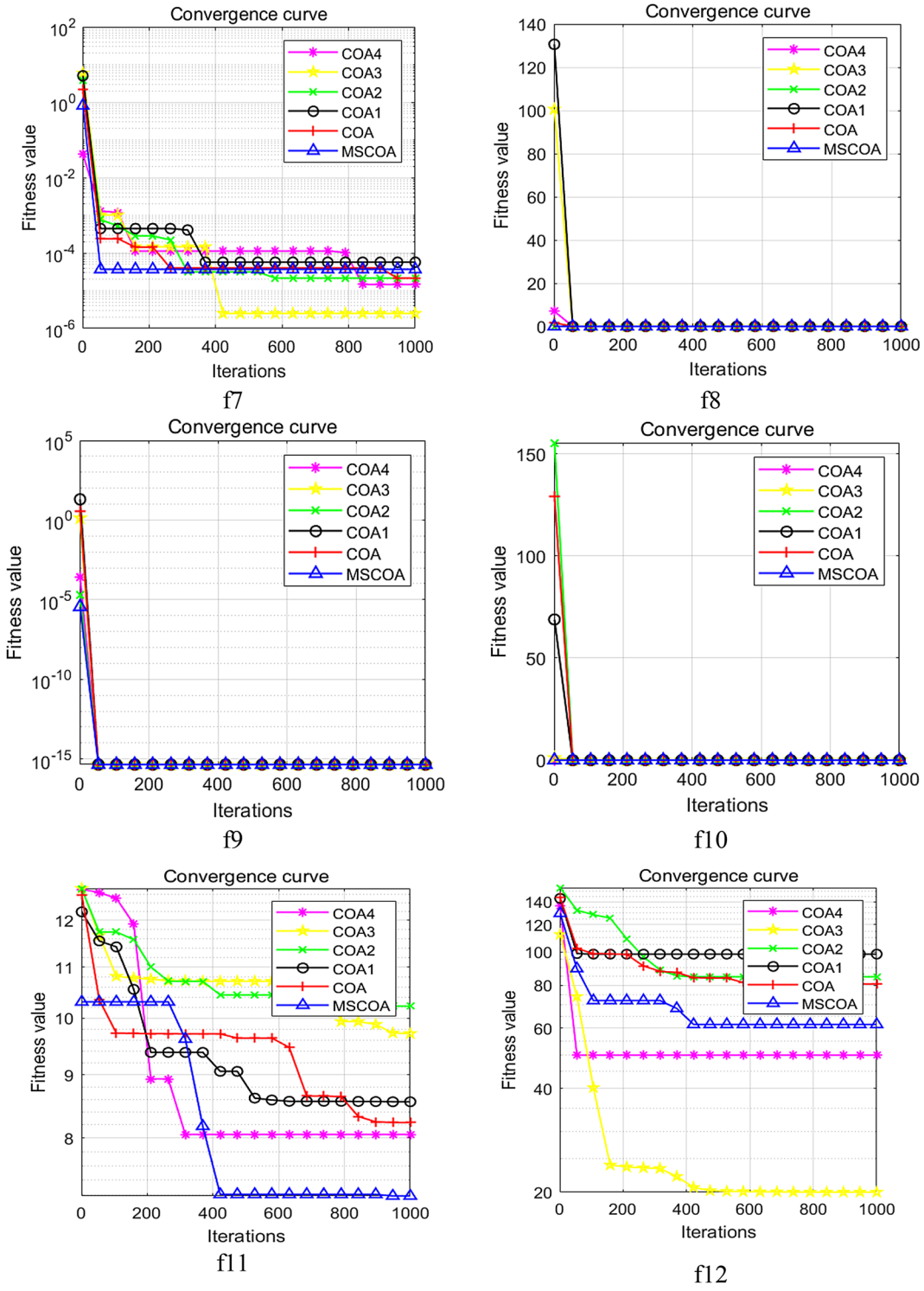


Fig. 9. (continued)

Comparison of Wilcoxon rank sum test

To assess the efficacy and optimization performance of MSCOA, this study used the Wilcoxon rank sum test[62] to confirm if the outcomes of MSCOA's running are statistically distinct from those of other algorithms at a significance threshold of a = 5%[63]. The hypothesis is disproved when p < a, suggesting that the two algorithms differ significantly from one another. In the event where p > a, the hypothesis is accepted, signifying that there is no discernible variation between the two methods. Table 4 displays the MSCOA, COA, SABO, GWO, WWPA, CDO, LPO and APO test findings at a significance level of a = 5%.

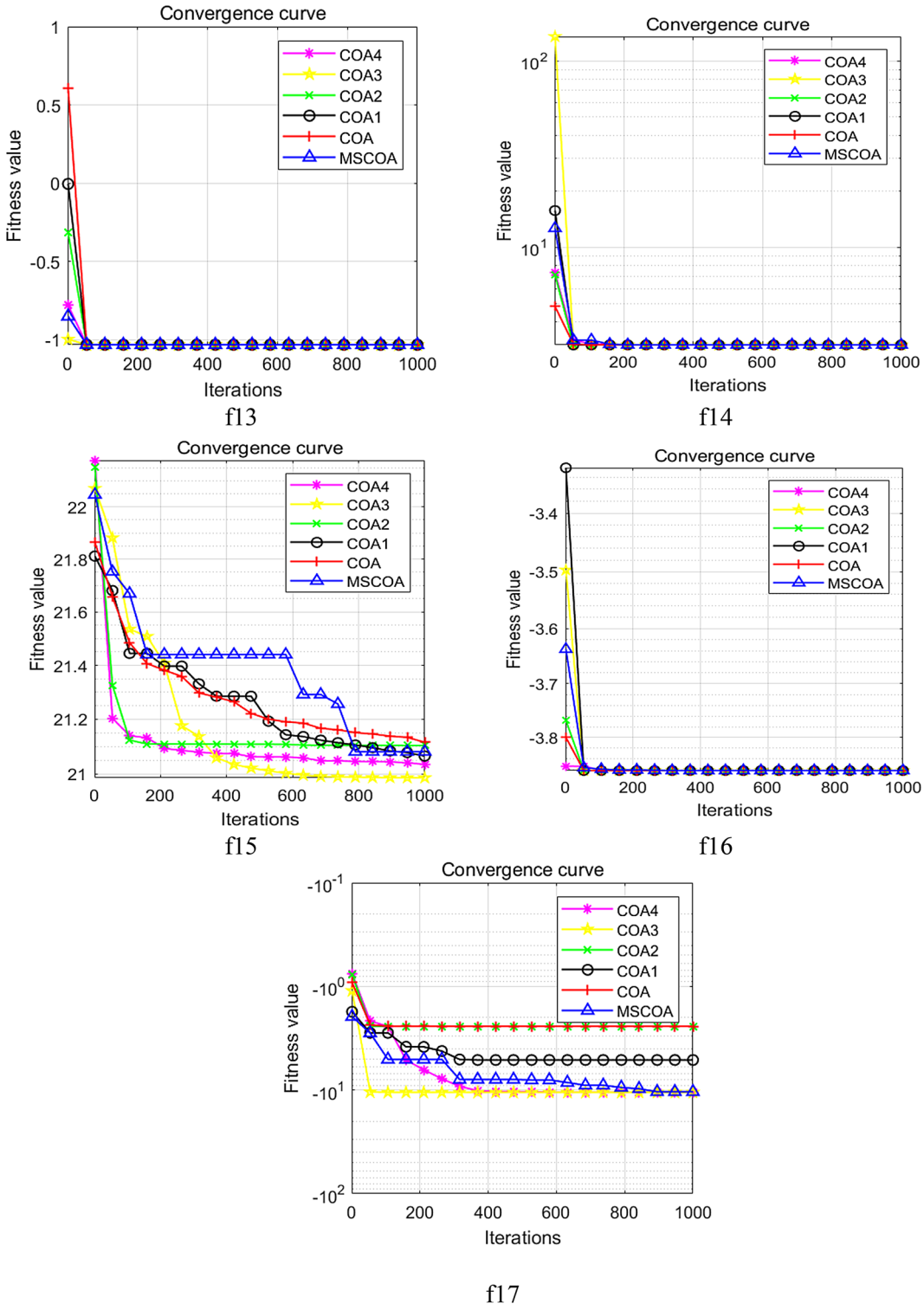
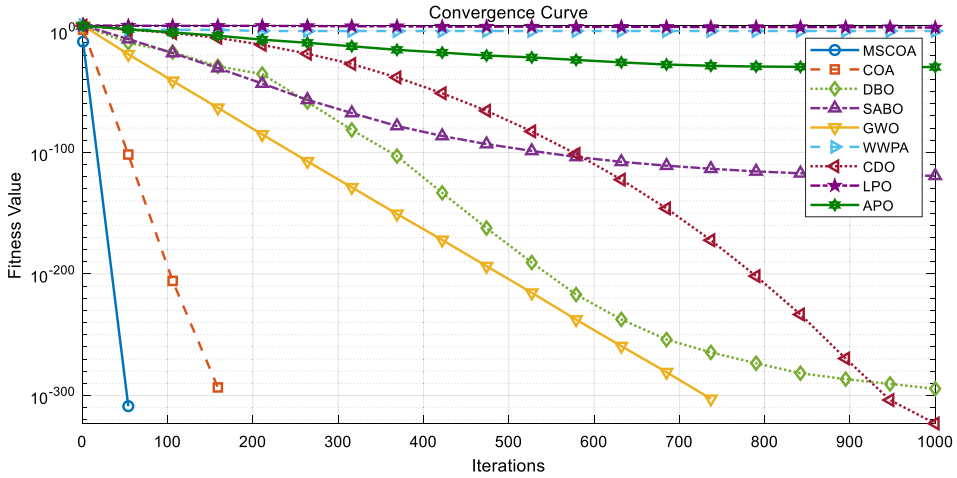


Fig. 9. (continued)

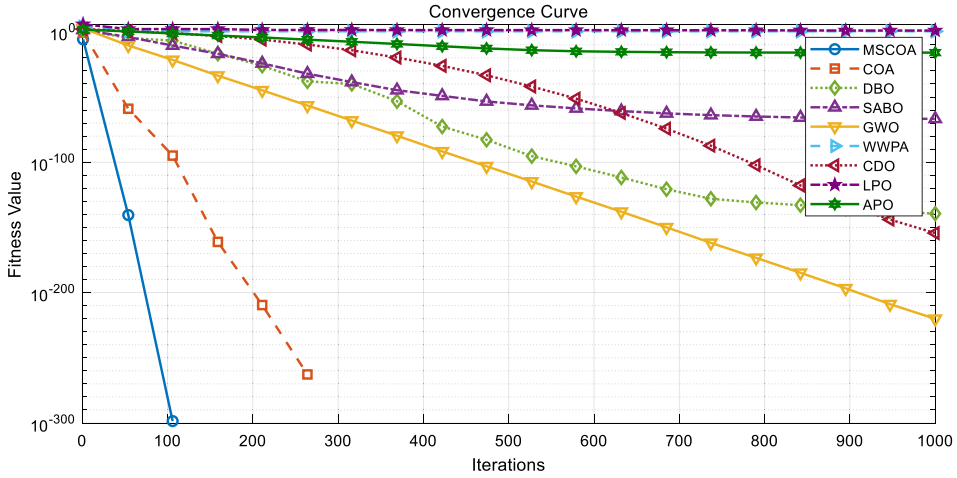
Table 4 shows that there are notable variations between the calculation results and the other seven methods, and that for the majority of test functions, the most of the p values of MSCOA are smaller than the significance level a.

Runtime analysis on different problem sizes

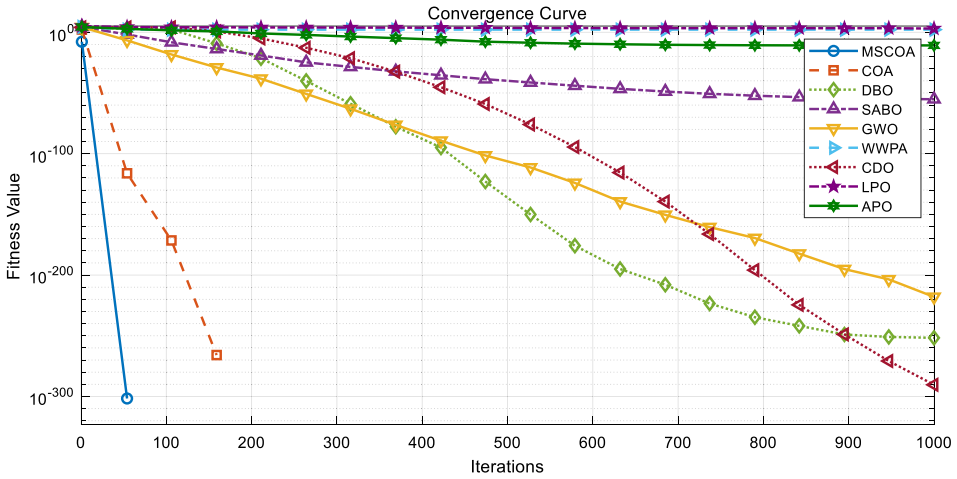
To supplement the theoretical complexity discussion and validate the scalability of MSCOA, this study conducts comprehensive experiments on benchmark functions F1–F4 across varying problem dimensions (10, 30, 50,



(a) f1

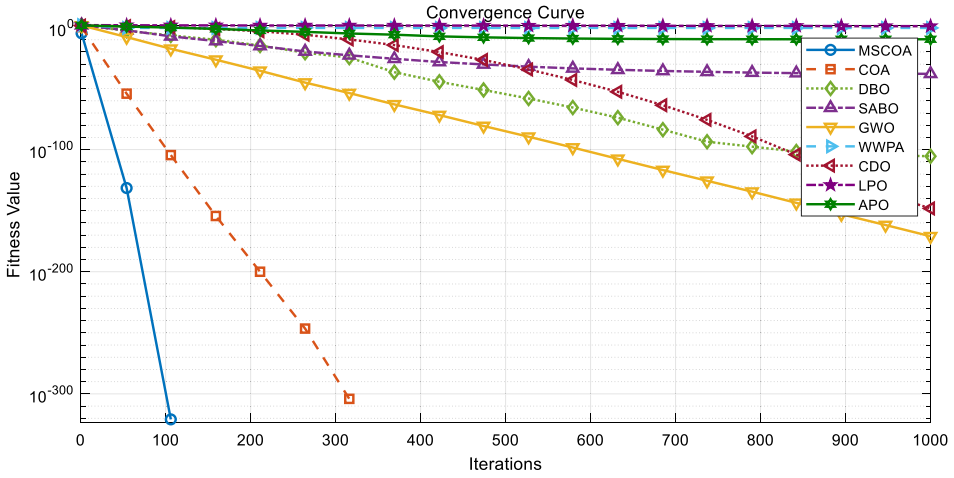


(b) f2

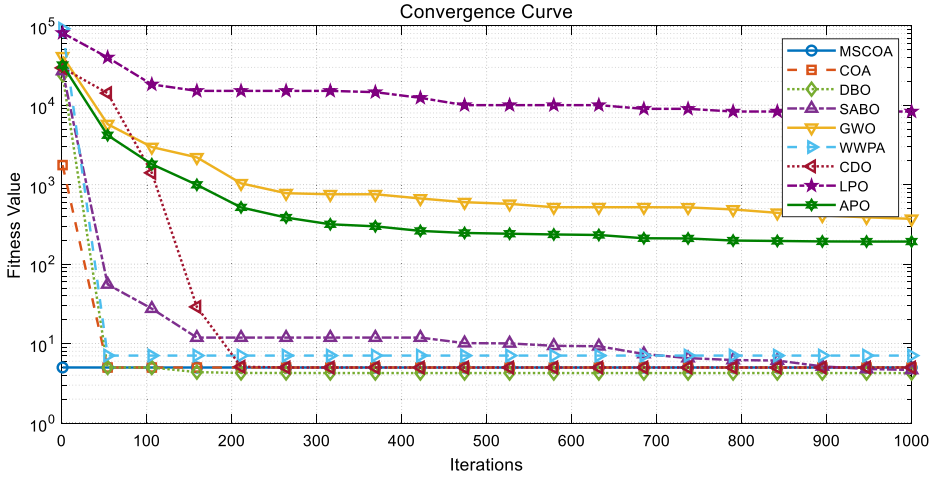


(c) f3

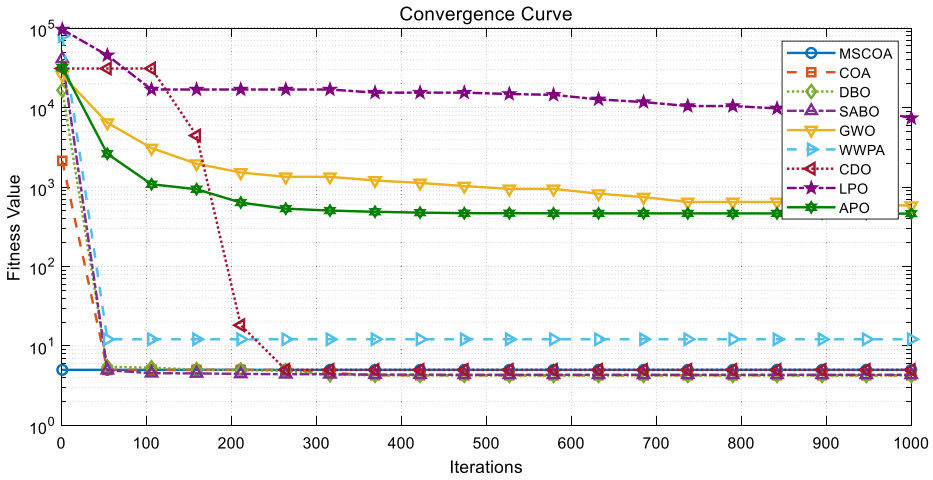
Fig. 10. Convergence diagram of various algorithms.



(d) f4



(e) f5

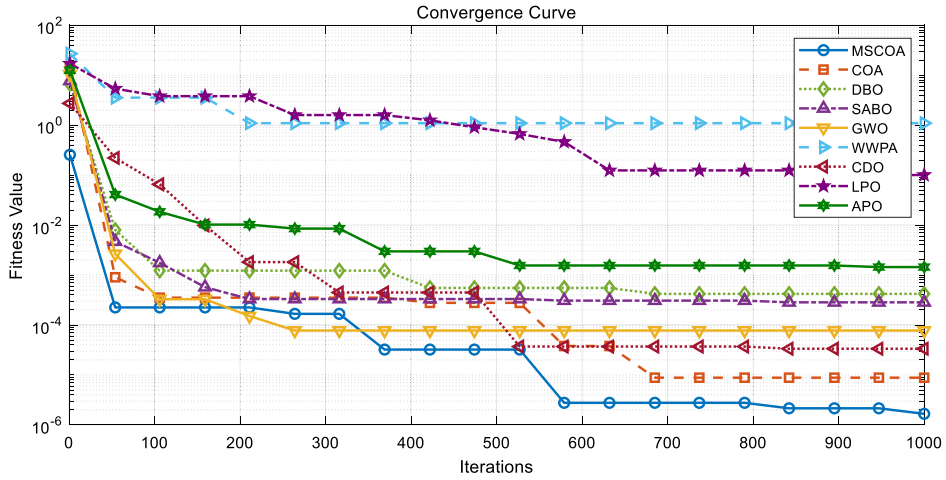


(f) f6

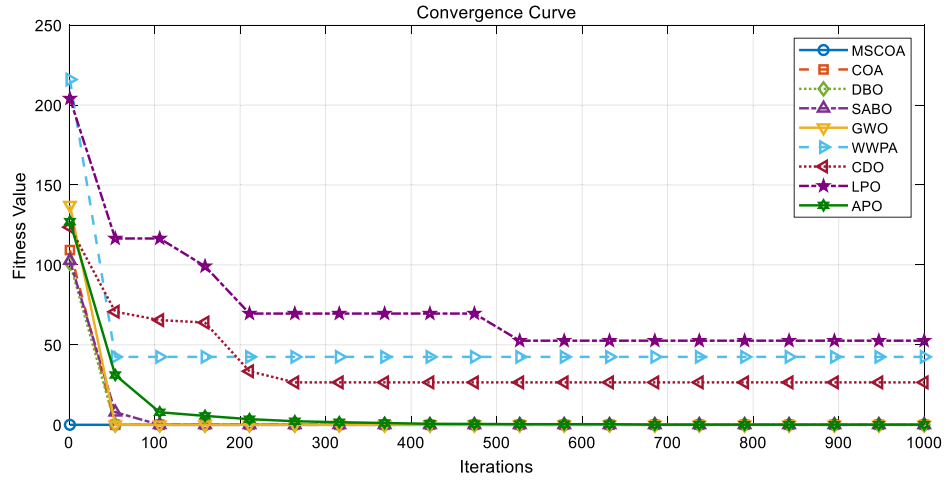
Fig. 10. (continued)

and 100). Additionally, a log-log fitting analysis is performed to derive the empirical time complexity. Table 5 presents the average computational time (in seconds) of MSCOA and the baseline algorithms.

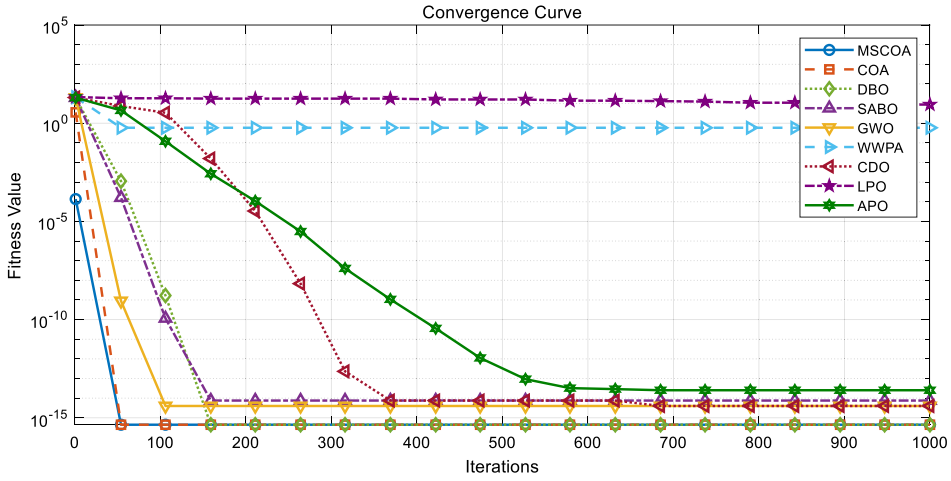
As shown in Table 3 the runtime of all algorithms increases with the problem dimensionality. MSCOA exhibits relatively higher computational time in high-dimensional scenarios (e.g., 100 dimensions), primarily due to the incorporation of enhanced strategies such as the Chaotic Reverse Initialization Strategy, which increases computational overhead.



(g) f7



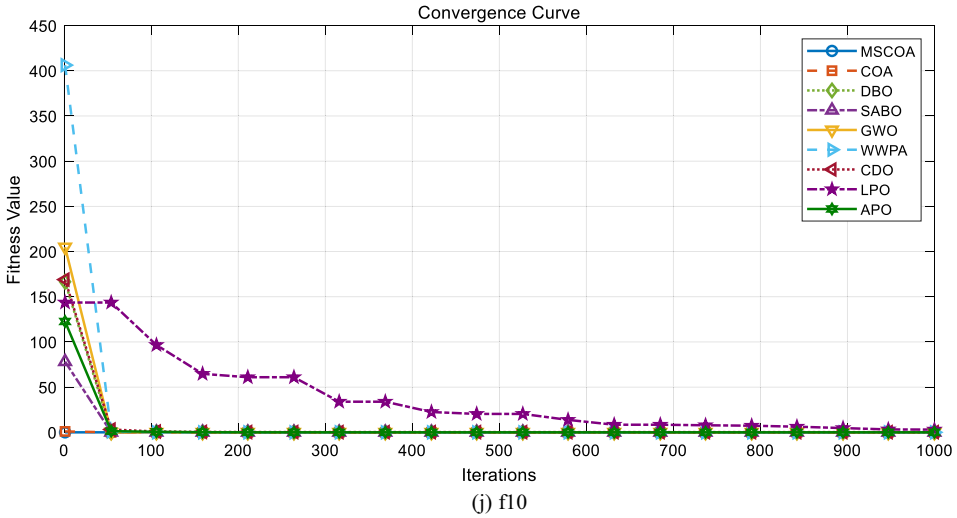
(h) f8



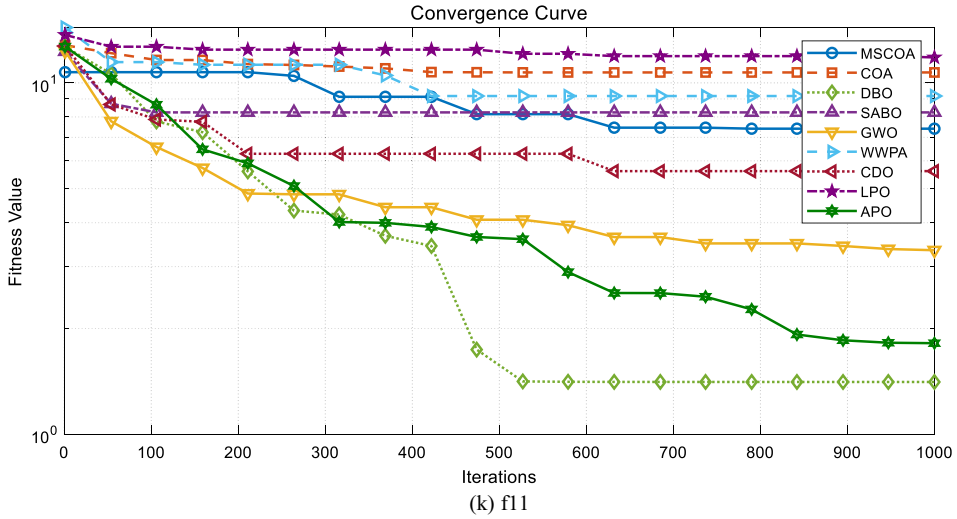
(i) f9

Fig. 10. (continued)

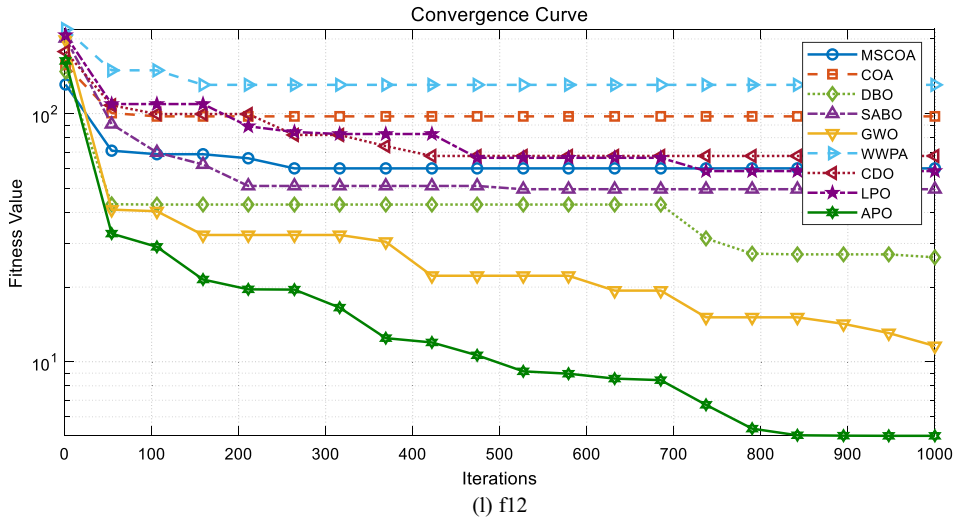
However, in conjunction with Fig. 12 and the empirical time complexity (where MSCOA approximates O(n), the observed linear increase in runtime aligns well with theoretical expectations. Compared to other algorithms, which demonstrate a slower runtime growth, MSCOA balances computational efficiency and optimization performance, ensuring effective search capability while maintaining a reasonable trade-off between runtime and solution quality.



(j) f10



(k) f11

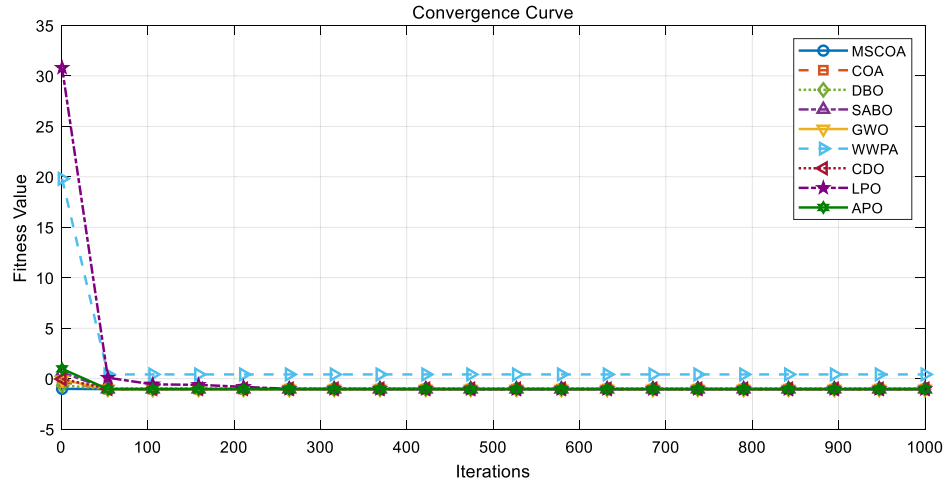


(l) f12

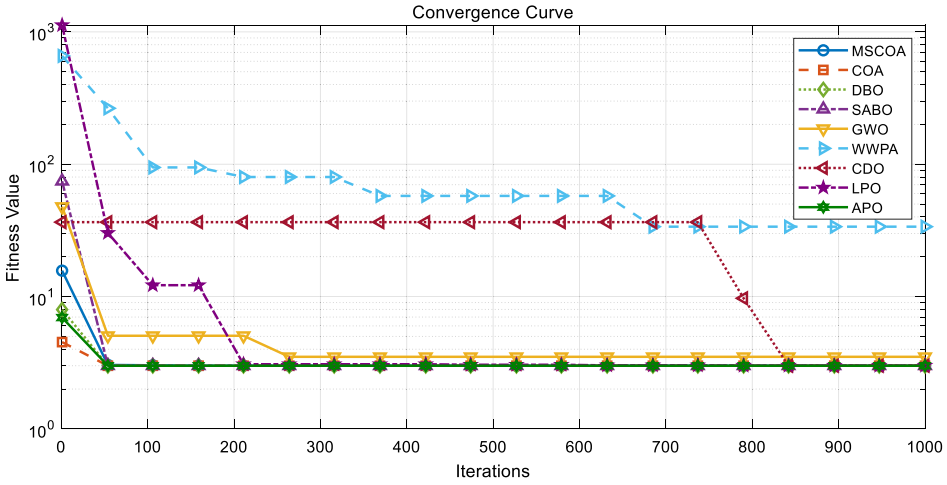
Fig. 10. (continued)

Scalability test of MSCOA for Large-Scale problems

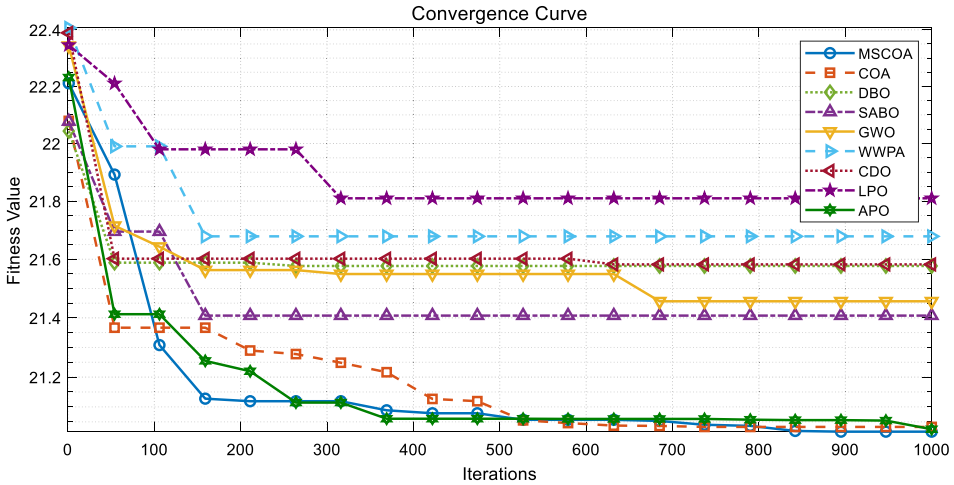
In order to further verify the scalability of MSCOA in high-dimensional optimization tasks, we conducted additional experiments on complex optimization problems with dimensions of D = 200 and D = 500. Table 3 summarizes the mean, standard deviation, and minimum value metrics obtained by MSCOA on selected variable-



(m) f13



(n) f14

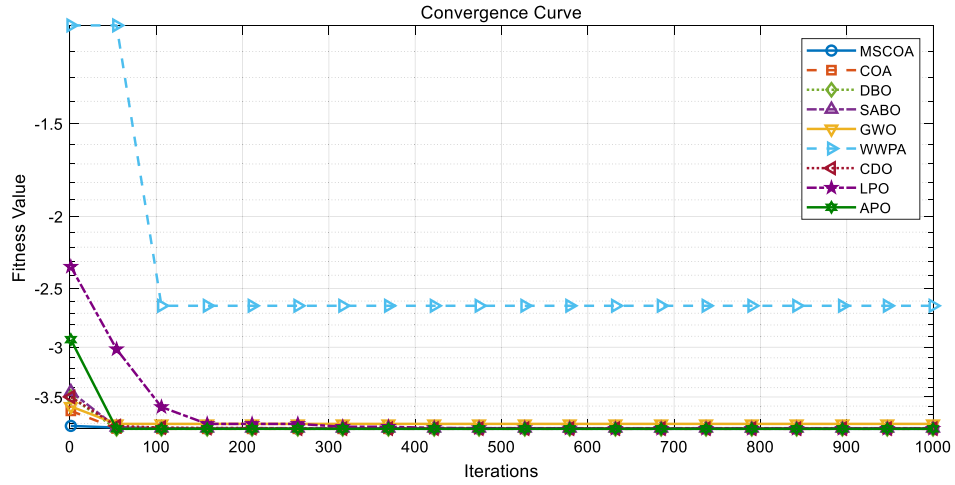


(o) f15

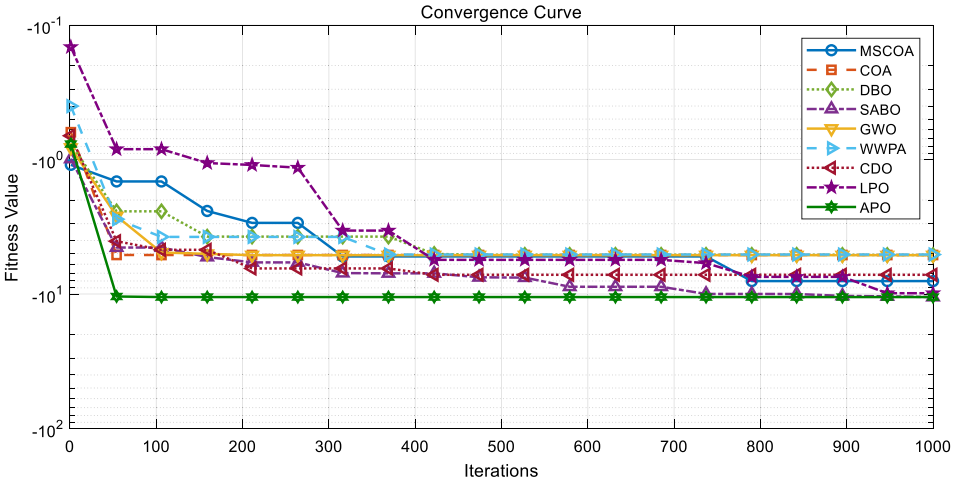
Fig. 10. (continued)

dimensional benchmark functions F1–F4, F7–F10, F13, and F14. The results indicate that even with a substantial increase in dimensionality, MSCOA continues to exhibit relatively stable and competitive performance.

From Table 3, it can be observed that for most functions, the fitness values achieved by MSCOA remain close to the global optimum, accompanied by small standard deviations. This outcome implies that, despite the significant increase in dimensionality, MSCOA still maintains commendable convergence stability and accuracy.



(p) f16



(q) f17

Fig. 10. (continued)

These findings are consistent with the time-complexity analysis presented in the previous section, which shows that the algorithm's runtime grows linearly with the problem dimension. Hence, MSCOA demonstrates strong scalability in large-scale optimization tasks.

Application of MSCOA in predicting breast cancer

Extreme learning machine

Huang et al.[64] introduced the Extreme Learning Machine (ELM) learning technique for a single hidden layer feedforward neural network. The structure of the ELM is illustrated in Fig. 13.

ELM involves solving a system where the weights between the input and hidden layer are randomly assigned, while the output weights β are determined by minimizing the difference between the predicted and target outputs. This is formulated as Eq. (27).

$$H \bullet \beta = T \tag{27}$$

where H is the output matrix of the hidden layer, β represents the vector of output weights, and T is the matrix of target outputs.

The solution for the output weights is found using the Moore-Penrose inverse of H[65], denoted as H^{+}. The solution is expressed as:

$$\beta = H^{+}T \tag{28}$$

This yields a unique optimal solution by merely defining the quantity of hidden nodes, without impacting the input weights or biases of the hidden neurons[66]. These characteristics have drawn a lot of academics to research the theory, applications, and extensions of ELM[67,68]. At the moment, supervised learning has been the primary focus, with promising results[69,70].

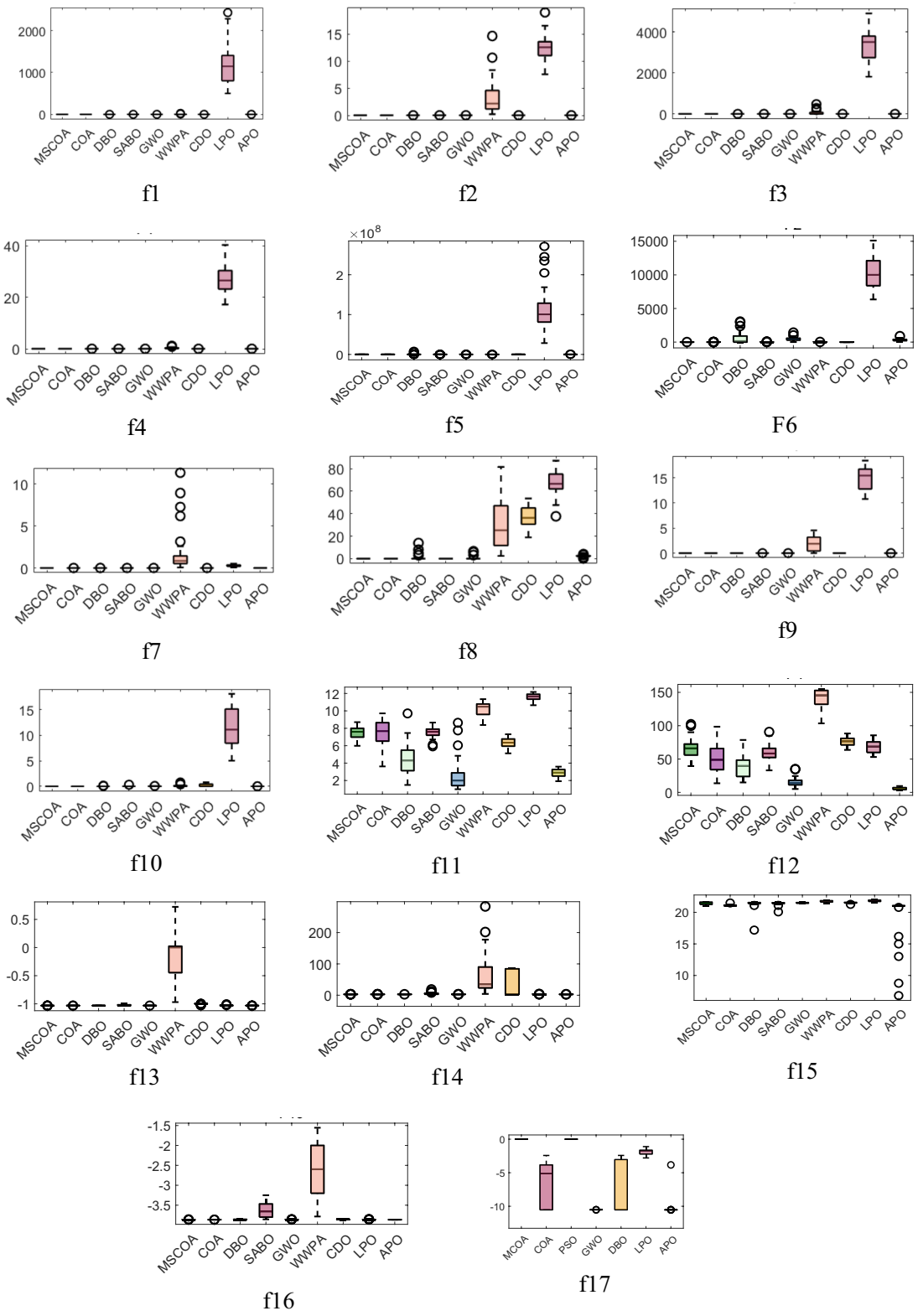


Fig. 11. The box-plot of various algorithms.

Function	Result	MSCOA	COA	DBO	SABO	GWO	WWPA	CDO	LPO	APO
F1	min	**0**	**0**	1.29E-106	6.66E-123	6.42E-37	1.26E-04	1.11E-103	3.58E+02	2.32E-15
	std	**0**	**0**	6.97E-66	6.88E-116	9.43E-33	2.32E+00	5.83E-91	3.81E+02	2.09E-12
	avg	**0**	**0**	1.38E-66	1.29E-116	3.20E-33	1.65E+00	1.64E-91	1.07E+03	8.94E-13
F2	min	**0**	**0**	7.96E-53	5.30E-67	4.02E-21	1.34E-02	6.35E-51	7.14E+00	1.66E-08
	std	**0**	**0**	7.45E-28	7.82E-65	3.17E-19	6.25E+00	7.25E-47	3.31E+00	4.60E-08
	avg	**0**	1.17E-267	1.36E-28	3.51E-65	1.43E-19	4.27E+00	2.84E-47	1.26E+01	6.80E-08
F3	min	**0**	**0**	5.94E-94	6.56E-68	1.32E-17	2.09E-01	1.53E-95	1.43E+03	1.31E-06
	std	**0**	**0**	6.00E-51	5.33E-47	2.88E-13	1.08E+02	1.91E-82	1.33E+03	1.10E-04
	avg	**0**	**0**	1.18E-51	9.73E-48	8.19E-14	6.49E+01	3.50E-83	3.45E+03	1.13E-04
F4	min	**0**	**0**	1.69E-53	1.34E-51	1.91E-12	5.10E-03	2.38E-47	1.81E+01	5.00E-05
	std	**0**	**0**	1.53E-22	4.35E-50	1.25E-10	3.06E-01	9.27E-43	4.98E+00	1.59E-04
	avg	**0**	1.36E-239	2.80E-23	3.14E-50	1.05E-10	3.80E-01	3.80E-43	2.53E+01	2.14E-04
F5	min	**1**	**1**	**1**	1	**1**	1.02E+00	**1**	2.84E+07	3.48E+02
	std	**0**	**0**	1.28E+06	5.40E-03	1.32E+05	2.03E+02	**0**	5.88E+07	6.32E+04
	avg	**1**	**1**	4.51E+05	1.00E+00	6.23E+04	7.89E+01	**1**	1.13E+08	5.82E+04
F6	min	4.24E+00	**4.22E+00**	4.27E+00	4.25E+00	3.47E+01	5.05E+00	5	6.35E+03	1.03E+02
	std	**0**	1.85E-01	1.43E+03	1.12E+01	2.99E+02	3.31E+00	**0**	2.46E+03	1.51E+02
	avg	4.98E+00	**4.95E+00**	1.01E+03	9.76E+00	4.83E+02	8.23E+00	5	1.05E+04	3.23E+02
F7	min	**4.26E-07**	6.86E-06	4.99E-05	2.45E-05	2.82E-04	3.40E-03	1.00E-05	1.04E-01	7.18E-04
	std	**7.98E-05**	1.14E-04	0.0014	1.01E-04	7.01E-04	4.20E+00	1.72E-04	1.35E-01	1.01E-03
	avg	**1.01E-04**	1.16E-04	1.60E-03	1.51E-04	1.30E-03	2.42E+00	2.28E-04	2.62E-01	2.29E-03
F8	min	**0**	**0**	**0**	**0**	**0**	4.10E-03	2.37E+01	3.10E+01	2.46E-01
	std	**0**	**0**	5.05E+00	**0**	2.28E+00	2.76E+01	7.64E+00	1.11E+01	1.03E+00
	avg	**0**	**0**	1.36E+00	**0**	1.62E+00	2.61E+01	4.30E+01	6.53E+01	2.32E+00
F9	min	**4.44E-16**	**4.44E-16**	**4.44E-16**	4.44E-16	1.82E-14	4.12E-02	4.00E-15	1.22E+01	2.56E-08
	std	**0**	**0**	**0**	1.08E-15	5.65E-15	1.76E+00	**0**	1.83E+00	1.30E-07
	avg	**4.44E-16**	**4.44E-16**	**4.44E-16**	3.64E-15	2.72E-14	2.69E+00	4.00E-15	1.55E+01	1.60E-07
F10	min	**0**	**0**	**0**	**0**	**0**	2.60E-03	**0**	3.25E+00	1.17E-09
	std	**0**	**0**	1.94E-02	9.70E-02	3.48E-02	2.44E-01	2.46E-01	4.52E+00	7.31E-03
	avg	**0**	**0**	6.90E-03	1.77E-02	3.44E-02	2.48E-01	1.31E-01	1.29E+01	7.31E-03
F11	min	6.45E+00	4.70E+00	1.41E+00	5.56E+00	**1.00E+00**	8.36E+00	4.42E+00	1.06E+01	1.92E+00
	std	7.21E-01	1.75E+00	1.87E+00	**6.70E-01**	1.2581	8.34E-01	7.73E-01	7.89E-01	4.27E-01
	avg	7.66E+00	7.74E+00	4.38E+00	7.49E+00	**2.63E+00**	1.01E+01	6.14E+00	1.16E+01	2.86E+00
F12	min	3.55E+01	1.19E+01	1.09E+01	3.58E+01	7.03E+00	1.26E+02	5.90E+01	5.31E+01	**3.14E+00**
	std	7.95E+00	3.14E+01	1.49E+01	1.20E+01	1.02E+01	8.47E+00	6.33E+00	9.28E+00	**1.58E+00**
	avg	6.86E+01	4.84E+01	3.54E+01	5.29E+01	2.06E+01	1.46E+02	7.54E+01	6.82E+01	**5.79E+00**
F13	min	**−1.03E+00**	**−1.03E+00**	**−1.03E+00**	**−1.03E+00**	**−1.03E+00**	−9.53E-01	−1.02E+00	**−1.03E+00**	**−1.03E+00**
	std	2.90E-06	1.37E-13	**5.83E-16**	2.61E-02	7.50E-08	1.37E+00	5.60E-03	4.47E-03	4.42E-15
	avg	**−1.03E+00**	**−1.03E+00**	**−1.03E+00**	−1.02E+00	**−1.03E+00**	1.90E-01	−1.00E+00	−1.03E+00	**−1.03E+00**
F14	min	**3.00E+00**	**3.00E+00**	**3.00E+00**	3.0039	**3.00E+00**	3.13E+00	3.00E+00	3.00E+00	**3.00E+00**
	std	3.90E-03	4.55E-10	3.22E-15	3.27E+00	1.48E+01	9.07E+01	3.37E+01	6.21E-02	**1.25E-15**
	avg	3.00E+00	**3.00E+00**	**3.00E+00**	4.42E+00	5.70E+00	6.09E+01	4.05E+01	3.05E+00	**3.00E+00**
F15	min	**2.04E+01**	2.10E+01	2.12E+01	2.10E+01	2.07E+01	2.14E+01	2.12E+01	2.15E+01	21.0122
	std	**7.41E-02**	1.98E+00	1.77E-01	8.74E-02	3.21E+00	1.44E-01	1.00E-01	1.19E-01	3.54E+00
	avg	2.15E+01	2.11E+01	2.14E+01	2.15E+01	2.10E+01	2.17E+01	2.15E+01	2.18E+01	**1.99E+01**
F16	min	**−3.86E+00**	**−3.86E+00**	**−3.86E+00**	−3.86E+00	**−3.86E+00**	−3.73E+00	−3.86E+00	−3.86E+00	**−3.86E+00**
	std	1.40E-03	1.41E-01	3.00E-03	2.01E-01	2.40E-03	6.29E-01	4.20E-03	6.19E-04	**2.71E-15**
	avg	**−3.86E+00**	−3.84E+00	−3.86E+00	−3.67E+00	−3.86E+00	−2.38E+00	−3.86E+00	−3.86E+00	**−3.86E+00**
F17	min	−4.8386	**−1.05E+01**	**−1.05E+01**	−9.3738	−1.05E+01	−1.05E+01	−9.10E+00	−8.35E+00	**−1.05E+01**
	std	**7.15E-01**	3.08E+00	2.92E+00	1.26E+00	1.79E+00	2.04E+00	1.45E+00	1.69E+00	1.70E+00
	avg	−3.46E+00	−6.79E+00	−8.71E+00	−4.90E+00	−9.95E+00	−5.23E+00	−5.73E+00	−3.90E+00	**−1.01E+01**

Table 2. Optimization results of different algorithms.

Function	MSCOA vs. COA	MSCOA vs. DBO	MSCOA vs. SABO	MSCOA vs. GWO	MSCOA vs. WWPA	MSCOA vs. CDO	MSCOA vs. LPO	MSCOA vs. APO
F1	1	1.21E-12	1.21E-12	1.21E-12	1.21E-12	1.21E-12	1.21E-12	1.21E-12
F2	1.66E-11	1.21E-12	1.21E-12	1.21E-12	1.21E-12	1.21E-12	1.21E-12	1.21E-12
F3	1	1.21E-12	1.21E-12	1.21E-12	1.21E-12	1.21E-12	1.21E-12	1.21E-12
F4	4.57E-12	1.21E-12	1.21E-12	1.21E-12	1.21E-12	1.21E-12	1.21E-12	1.21E-12
F5	**1**	5.85E-09	1.21E-12	1.21E-12	1.21E-12	**1**	1.21E-12	1.21E-12
F6	0.022783	0.013894	1.64E-05	1.72E-12	1.72E-12	**0.1608**	1.72E-12	1.72E-12
F7	**0.652**	2.03E-09	0.0406	4.08E-11	3.02E-11	0.0024	3.02E-11	3.02E-11
F8	**1**	0.0216	**1**	1.90E-09	1.21E-12	1.21E-12	1.21E-12	1.21E-12
F9	**1**	**1**	3.94E-12	8.30E-13	1.21E-12	1.69E-14	1.21E-12	1.21E-12
F10	**1**	0.0216	**0.3337**	1.70E-08	1.21E-12	1.95E-09	1.21E-12	1.21E-12
F11	**0.1453**	6.01E-08	0.0963	6.72E-10	1.61E-10	8.48E-09	3.02E-11	3.02E-11
F12	0.0251	3.35E-08	5.61E-05	3.02E-11	3.02E-11	8.84E-07	**0.25805**	3.02E-11
F13	2.99E-11	1.45E-11	1.96E-10	0.015	3.02E-11	3.02E-11	3.02E-11	2.60E-11
F14	3.02E-11	2.89E-11	2.15E-10	1.17E-09	3.02E-11	1.41E-04	3.02E-11	5.18E-12
F15	8.89E-10	0.0026	**0.0679**	**0.0724**	2.03E-07	**0.0877**	2.67E-09	1.61E-10
F16	3.02E-11	5.51E-06	3.34E-11	**0.1335**	3.02E-11	1.55E-09	3.02E-11	1.21E-12
F17	3.59E-05	9.23E-07	2.25E-04	5.49E-11	0.025101	3.01E-07	0.0069724	1.14E-09

Table 3. Wilcoxon signed-rank test result.

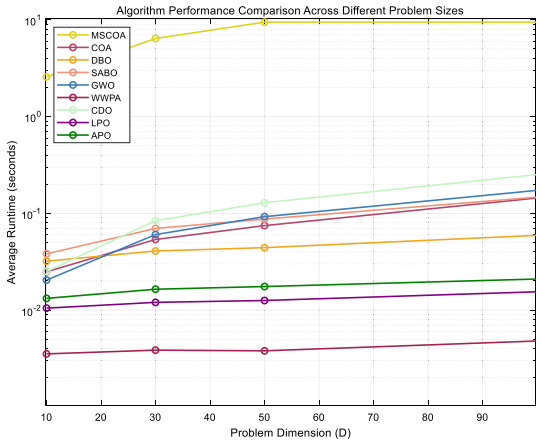


Fig. 12. Algorithms performance comparison across different problem sizes.

MSCOA-ELM algorithm modeling

The MSCOA-ELM algorithm model is constructed based on the 2 algorithm models described in this paper. Using the MSCOA algorithm, the optimal parameters are obtained by carrying out the phase of heat avoidance, the competition phase, and the foraging phase, which are the 3 processes, and then input into the ELM algorithm to seek the optimal prediction effect. Figure 14 depicts the MSCOA-ELM algorithm model's execution procedure.

In Fig. 14, ELM model using MSCOA are shown. First, after the algorithm starts, the parameters of MSCOA and ELM are initialized to ensure that all variables and parameters are set to their initial state. Next, the fitness value of each population individual is computed, and the individual's merit is measured by evaluating the fitness of the population. Then, it enters the behavioral updating phase, executing refuge, competition and foraging behaviors sequentially, and constantly updating the positions of individuals in the population through Eq. (21), Eq. (8), Eq. (10), or Eq. (11), so as to seek the globally optimal solution. After each update, it checks whether the stopping condition is satisfied, and if it is not satisfied, the position update and fitness evaluation are repeated until the halting condition is fulfilled. When the halting condition is reached, the algorithm outputs the optimized MSCOA parameters and the corresponding ELM model to ensure that the model parameters are optimal. Finally, the algorithm ends and outputs the best optimization results for further model training and prediction.

Performance evaluation metrics

A confusion matrix is a matrix that categorizes the classification problem according to the two dimensions of the true situation and the discriminative situation, and the binary classification confusion matrix is displayed in Fig. 15.

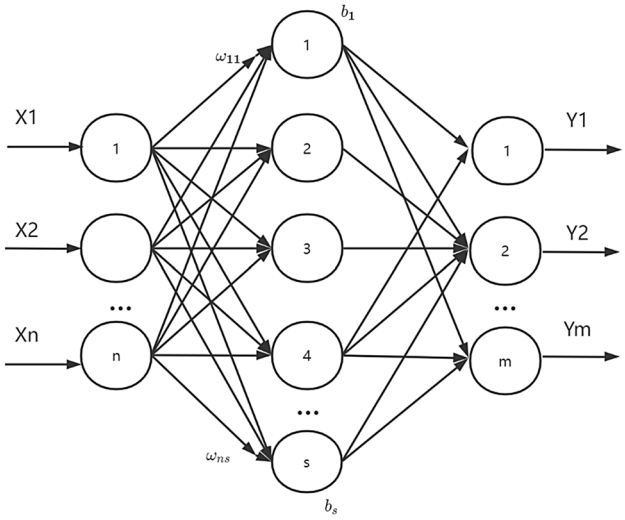


Fig. 13. ELM model structure.

The matrix has four components: TP (True Positive): Correctly classified positive examples.TN (True Negative): Correctly classified negative examples. FP (False Positive): Incorrectly classified as positive when they are negative (Type I error). FN (False Negative): Incorrectly classified as negative when they are positive (Type II error).

In this paper, 20% of the data was utilized for testing all models, while the remaining 80% was used for training. A number of performance metrics, including as each model's accuracy, precision, recall, and F1 score, are assessed, based on the four variables of the confusion matrix. The following is an explanation of the performance measures:

The percentage of correctly anticipated events to all occurrence is known as accuracy.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{29}$$

The following ratio represents precision: the amount of correctly forecast positive cases divided by the entire amount of expected positive cases. The following is the formula 30.

$$Precision = \frac{TP}{TP + FP} \tag{30}$$

Recall is described as the proportion of correctly anticipated positive events to each instance within actual class. The following formula is displayed.

$$Recall = \frac{TP}{TP + FN} \tag{31}$$

F1score: the precision and recall weighted average, which has the following definition:

$$F1score = 2 \times \frac{Precision \times Recall}{Precision + Recall} \tag{32}$$

AUC-ROC curve: the model's capacity to differentiate between categories is shown by the AUC, while the ROC is a probability curve. It indicates the model's capacity for category discrimination. AUC indicates how well the model distinguishes between various groups. Plotting TPR on the y-axis against FPR on the x-axis is known as the ROC curve. FPR is defined as follows, and TRP is a synonym for recall:

$$\mathrm{FPR} = \frac{\mathrm{FP}}{\mathrm{FP} + \mathrm{TN}} \tag{33}$$

Optimal hidden neurons in ELM

To determine the optimal number of hidden neurons in the ELM and assess its impact on performance, we conducted comparative experiments using the MSCOA-ELM model with configurations comprising 2, 5, 8, and 20 hidden neurons. The results indicated that when the number of hidden neurons was set to 5 or 20, the accuracy reached 0.9825. As shown in Fig. 16, the average optimal number of hidden neurons was 7.50, with an average optimal accuracy of 0.9912. A smaller number of neurons resulted in underfitting due to insufficient feature learning capacity, while an excessive number increased model complexity without significantly enhancing

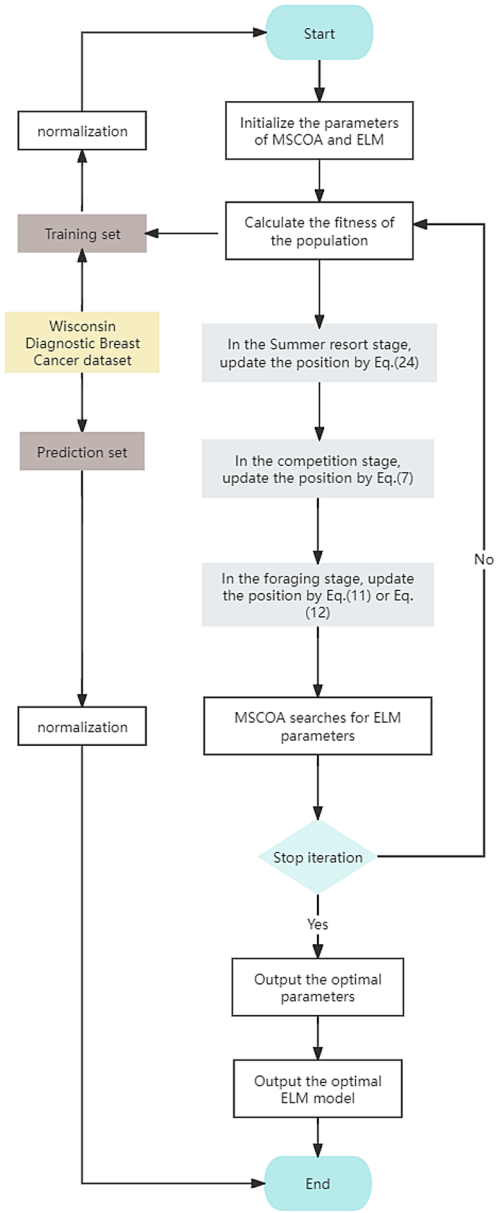


Fig. 14. Breast cancer prediction process based on MSCOA-ELM.

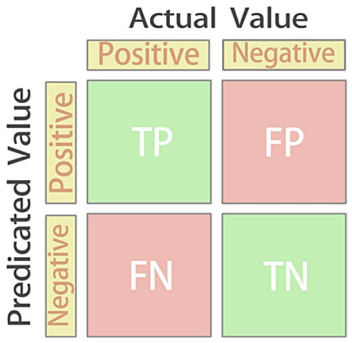


Fig. 15. Binary confusion matrix.

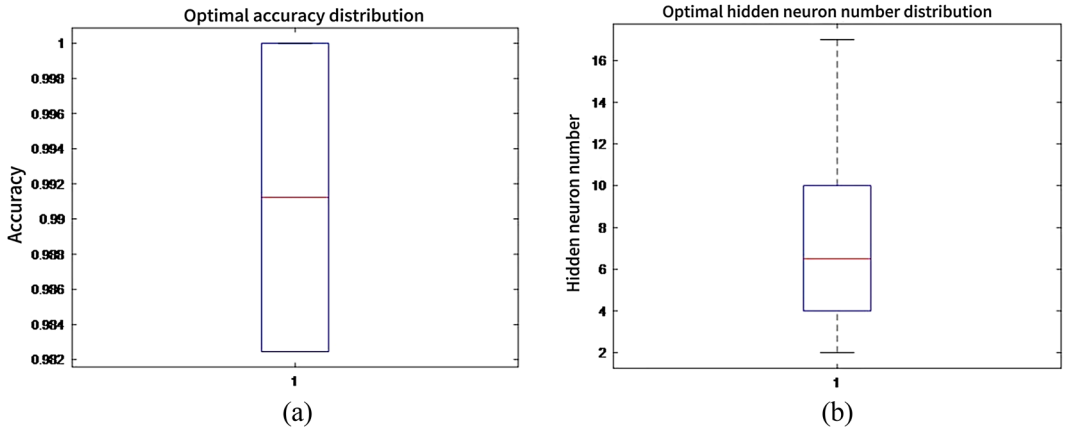


Fig. 16. Box Plot of Optimal Accuracy Distribution and Optimal Hidden Neuron Number Distribution.

Dimension	MSCOA	COA	DBO	SABO	GWO	WWPA	CDO	LPO	APO
10	1.2524	0.0289	0.0355	0.0418	0.0235	0.0039	0.0281	0.0115	0.0139
30	3.7156	0.0529	0.0411	0.0700	0.0589	0.0038	0.0851	0.0119	0.0155
50	5.1788	0.0757	0.0447	0.0870	0.0946	0.0039	0.1344	0.0131	0.0168
100	9.4740	0.1417	0.0612	0.1454	0.1772	0.0051	0.2574	0.0170	0.0223

Table 4. Comparison of performance measures of each model.

accuracy. Considering the trade-offs among model complexity, generalization ability, and result stability, we ultimately selected 8 hidden neurons. This configuration enables the ELM to effectively extract features while avoiding both underfitting and overfitting, thereby ensuring reliable performance in subsequent applications.

Results and discussion

1) generalization ability evaluation of MSCOA-ELM on diabetes dataset Firstly, to evaluate the generalization capability of MSCOA-ELM, experiments were conducted on the diabetes dataset from the UCI Machine Learning Repository. The performance of each model index is shown in Table 4, and the ROC curve is shown in Fig. 17. As shown in Table 3, MSCOA-ELM exhibits marked improvements in accuracy, precision, recall, F1 score, and AUC compared to the original ELM model, indicating that the introduction of MSCOA can more effectively optimize ELM parameters and enhance the model's feature learning and classification performance. However, from an overall perspective, certain indicators of SABO-ELM and APO-ELM remain slightly superior to those of MSCOA-ELM, suggesting that while MSCOA-ELM has already achieved a commendable performance on this dataset, there is still room for further improvement. In general, the experimental results of MSCOA-ELM on the diabetes dataset confirm its promising generalization ability, but also highlight the need for deeper research and exploration in algorithm refinement and parameter tuning.

2) ability evaluation of MSCOA-ELM on breast Cancer dataset Subsequently, MSCOA is formally applied to the Wisconsin Diagnostic Breast Cancer dataset from the UCI dataset. From UCI machine learning repository website https://archive.ics.uci.edu/ml/index.php search Wisconsin Diagnostic Breast Cancer to download.

This section aims to use machine learning techniques to establish an adjunctive diagnostic for breast cancer prediction. Numerous models, including the recommended MSCOA-ELM, were assessed, tuned, and trained. Fig. 18 illustrates the outcomes of the confusion matrix experiments for all predictive models applied to the Wisconsin Diagnostic Breast Cancer dataset, alongside a visualization of the classification results. Based on metrics, each model's performance was assessed, and Table 7 presents the findings.

As presented in Table 5, MSCOA-ELM achieved 100% in accuracy, precision, recall, and F1-score, demonstrating that MSCOA can robustly converge to the optimal set of weights and biases of the ELM model. GWO-ELM (accuracy = 0.976) and COA-ELM (accuracy = 0.967) approached the performance of MSCOA-ELM but exhibited relatively lower recall. DBO-ELM showed the highest recall (0.959) with a lower precision (0.946). SABO-ELM maintained a perfect precision of 1.0, yet its recall was merely 0.893. WWPA-ELM presented the lowest recall (0.852) among the models. CDO-ELM attained an F1-score of 0.927, surpassing WWPA-ELM's 0.915, thereby highlighting its superior overall performance. The original ELM model exhibited notably lower accuracy (0.710) and F1-score (0.778), underscoring the necessity of metaheuristic algorithms for parameter optimization. The AUC values and ROC curve shown in Fig. 19 further validated these results. Meanwhile, the ROC curves and AUC metrics of other models reflected their trade-offs between true positive rate and false positive rate, aligning with the analyzed precision-recall dynamics.

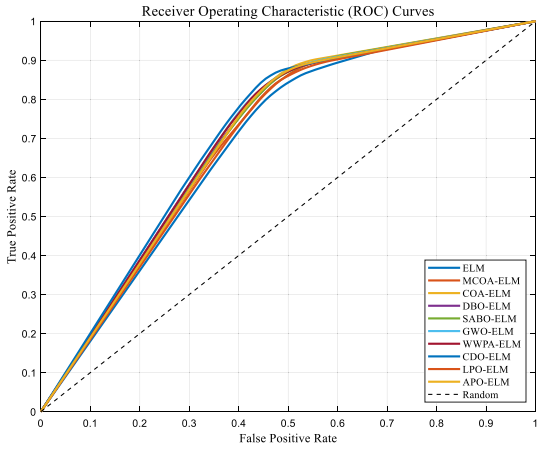


Fig. 17. Comparison of ROC for Different Models on Diabetes Dataset.

3) robustness verification on noisy data To validate the robustness of the MSCOA-ELM in real medical scenarios, this study introduces Gaussian noise and label flipping into the Wisconsin Diagnostic Breast Cancer dataset (WDBC) to simulate the measurement errors or label noise that may occur in actual clinical data. Specifically, Gaussian noise with a mean of 0 and a standard deviation of σ is added to the input features, with the noise level set at σ = 5%. In addition, 5% of the sample labels are randomly flipped to simulate mislabeling. After generating the noisy data, the dataset is partitioned into training and testing sets using the original 80%–20% split.

Experimental results, as shown in Table 6, reveal that on the original, noise-free data, the standard ELM achieves an accuracy of only 64.0351%, whereas the MSCOA-ELM, by optimizing the ELM parameters, attains an accuracy of 92.9825%, thereby demonstrating a stronger feature learning capability. After introducing noise, the standard ELM's accuracy increased to 78.0702%, which may be attributed to overfitting to noise patterns; however, MSCOA-ELM maintained a high accuracy of 90.3509%. This indicates that MSCOA-ELM effectively filters out noise impacts and sustains model stability, making it more suitable for real-world medical scenarios where data imperfections are prevalent.

Conclusion and future work

This study proposes an Enhanced Crayfish Optimization Algorithm (MSCOA) by integrating chaotic reverse exploration initialization, an adaptive t-distributed feeding strategy, and a ternary optimization mechanism, significantly improving the algorithm's global search capability, local exploitation efficiency, and dynamic balance performance.

Experimental results demonstrate that MSCOA generally outperforms or matches traditional COA and other metaheuristic algorithms on 17 benchmark functions from CEC2019 and CEC2005. For unimodal functions, MSCOA exhibits superior local exploitation efficiency and consistently identifies global optima. On multimodal, composite, and high-dimensional functions, MSCOA remains competitive, effectively avoiding local optima while balancing global exploration and local optimization. Wilcoxon rank-sum tests (95% confidence level) confirm significant performance differences between MSCOA and comparison algorithms.

Furthermore, MSCOA is integrated with Extreme Learning Machine (ELM) for breast cancer prediction. Evaluated on the Wisconsin Diagnostic Breast Cancer dataset (UCI), MSCOA-ELM achieves 100% accuracy, demonstrating its potential to assist clinicians in earlier and more accurate diagnoses.

However, there are still some limitations in the current study:

Theoretical limitations

- While MSCOA improves the local-global balance, its performance on highly complex problems remains improvable, and the algorithm's complexity may lead to increased computational overhead for large-scale applications.
- Although the adaptive t-distribution feeding strategy and hybrid weights have been empirically validated, theoretical proofs of convergence remain incomplete.
- Performance in spaces with over 1000 dimensions requires further validation, as the inherent randomness of the initial chaotic mapping may impact stability.
- In noisy environments, the performance of MSCOA may deteriorate due to the influence of measurement errors and imperfect data on fitness evaluation, which can result in suboptimal solutions.
- The current version of MSCOA is primarily designed for single-objective optimization, and its effectiveness in multi-objective optimization problems has not been sufficiently validated.

Practical limitations

- Model performance is heavily dependent on the quality of medical data and is sensitive to noise and class imbalance.

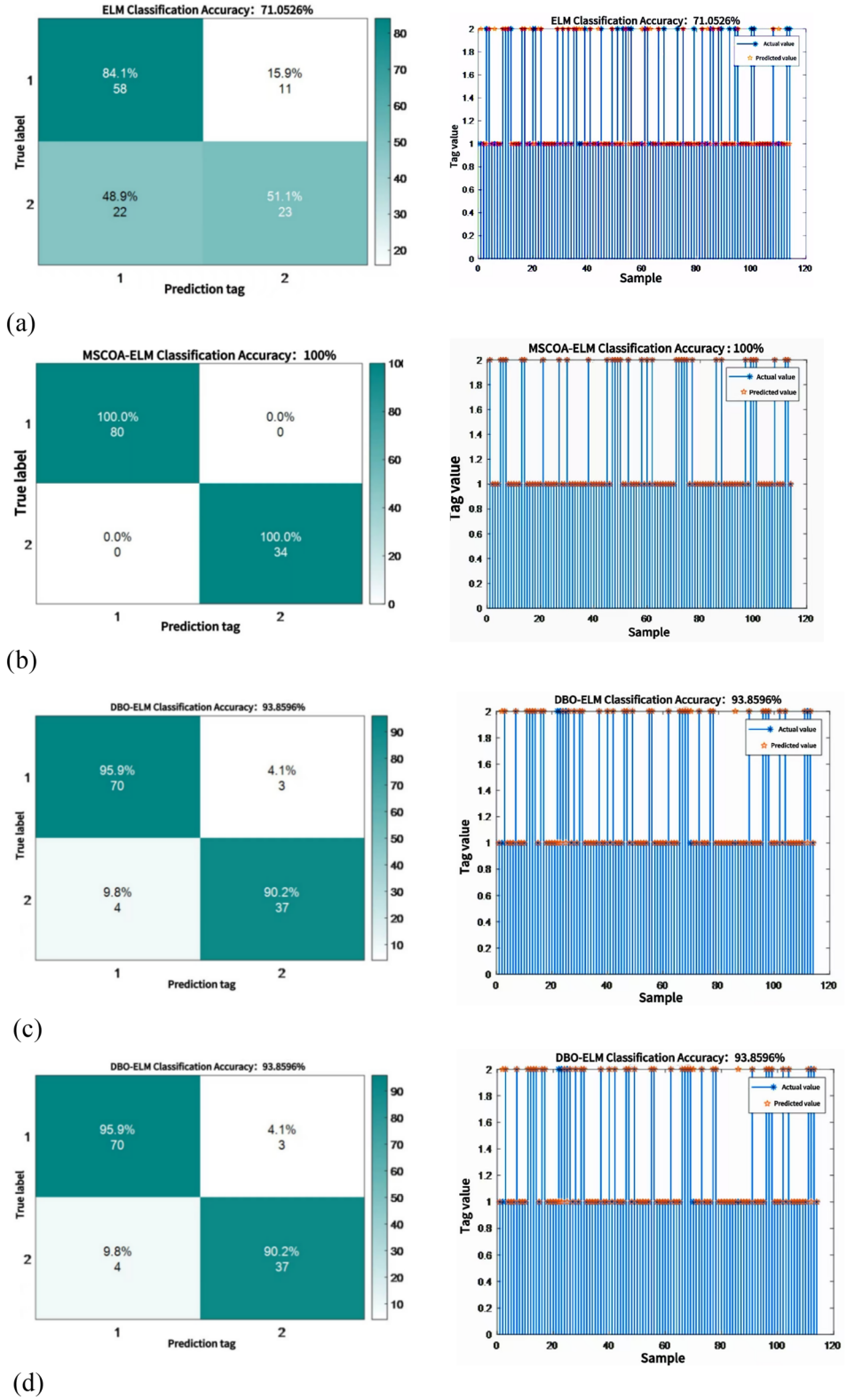


Fig. 18. Comparison of ROC Curves for Different Models on Breast Cancer Dataset.

- Training MSCOA-ELM on embedded systems may face resource constraints compared to lighter-weight algorithms.
- The training time of MSCOA-ELM may be limited in embedded devices when compared to more efficient models.

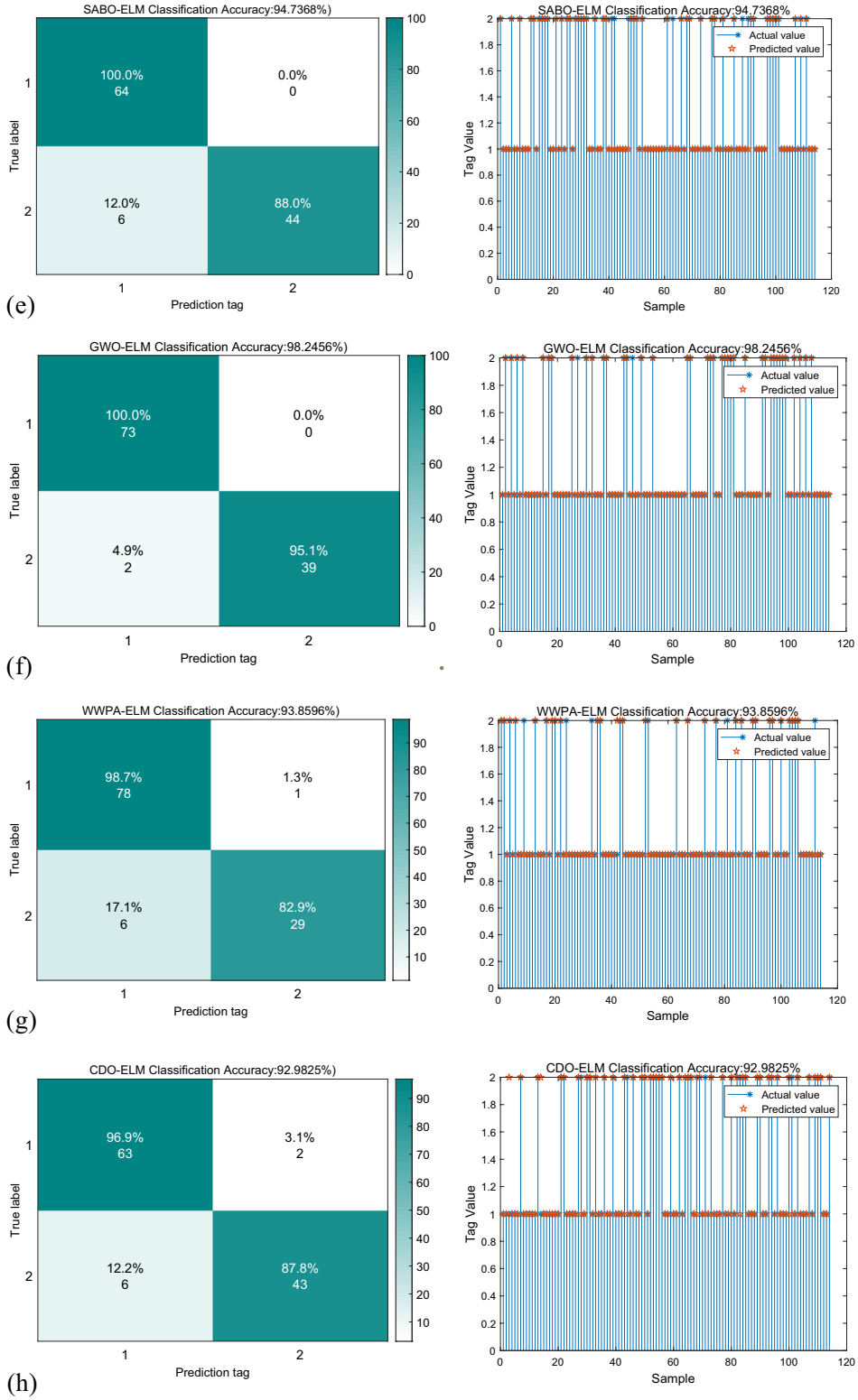


Fig. 18. (continued)

Future work can be improved in the following areas:

- Enhance MSCOA via novel mechanisms (e.g., multi-strategy fusion) to broaden its applicability across various fields such as finance, energy, and industrial optimization.
- Integrate machine learning with metaheuristics to improve adaptability.

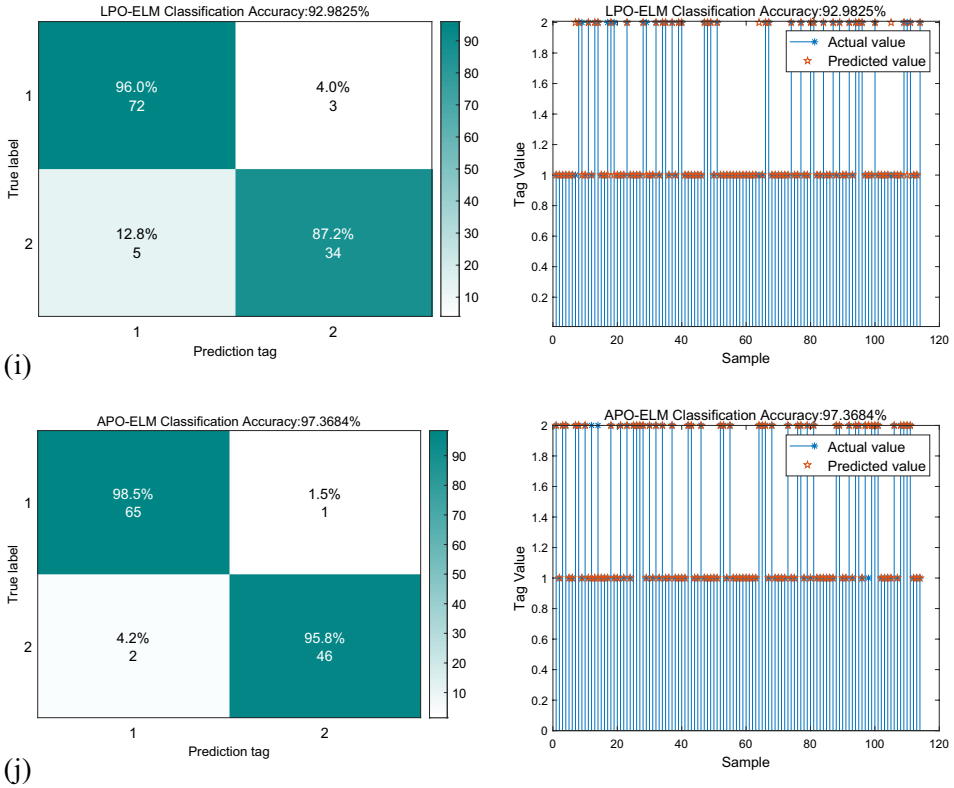


Fig. 18. (continued)

Model	Accuracy	Precision	Recall	F1-Score	AUC
ELM	0.710	0.725	0.840	0.778	0.9222
MSCOA-ELM	**1**	**1**	**1**	**1**	**0.9580**
COA-ELM	0.967	1	0.937	0.967	0.9533
DBO-ELM	0.939	0.946	0.959	0.951	0.9579
SABO-ELM	0.940	1	0.893	0.943	0.9575
GWO-ELM	0.976	1	0.953	0.976	0.9573
WWPA-ELM	0.908	0.987	0.852	0.915	0.9363
CDO-ELM	0.924	0.969	0.888	0.927	0.9555
LPO-ELM	0.929	0.935	0.960	0.947	0.9576
APO-ELM	0.974	0.970	0.985	0.977	0.9559

Table 7. Comparison of performance measures of each model.

- Apply model pruning or quantization techniques to reduce resource consumption in edge computing environments.
- Investigate strategies to enhance MSCOA's robustness against noise to improve performance in noisy environments.
- Develop multi-objective extensions of MSCOA to enhance its applicability and reliability in complex real-world scenarios.

Finally, at the policy level, MSCOA-ELM provides a cost-effective AI tool for early breast cancer screening, supporting standardized diagnostic workflows.

- Governments and healthcare institutions should prioritize funding for "optimization algorithm + medical AI" research.
- establish data-sharing platforms to bridge regional healthcare disparities.
- Open-sourcing the MSCOA-ELM framework could democratize access to advanced diagnostic tools in developing regions.

Test Function	D = 200			D = 500		
	avg	std	min	avg	std	min
F1	0	0	0	0	0	0
F2	0	0	0	0	0	**0**
F3	0	0	0	0	0	0
F4	0	0	0	0	0	0
F7	1.27E-04	1.97E-04	6.72E-06	1.2996E-04	1.28E-04	1.43E-06
F8	0	0	0	0	0	0
F9	8.88E-16	0	8.88E-16	8.88E-16	0	8.88E-16
F10	0	0	0	0	0	0
F13	−1.03E + 00	3.12E-07	−1.32E + 00	−1.03E + 00	2.10E-06	−1.03E + 00
F14	3.00E + 00	1.06E-02	3.00E + 00	3.00E + 00	5.28E-03	3.00E + 00

Table 5. The test results of MSCOA for higher dimensional problems (D = 200 and D = 500).

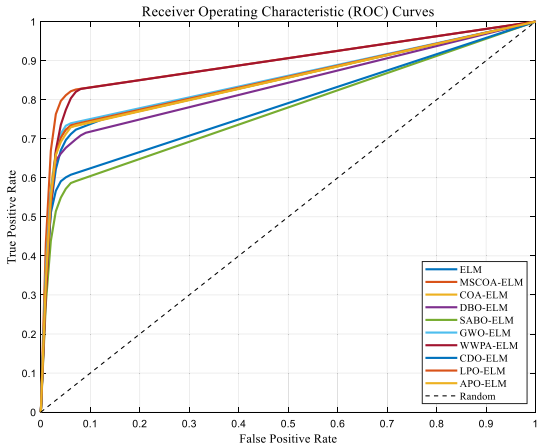


Fig. 19. Comparison of ROC Curves for Different Models on Breast Cancer Dataset.

Data Type	Algorithm	Accuracy
Original	Standard ELM	64.0351%
Original	MSCOA-ELM	92.9825%
Noisy	Standard ELM	78.0702%
Noisy	MSCOA-ELM	90.3509%

Table 8. Accuracy comparison between MSCOA - ELM and standard ELM on original and noisy Data.

Model	Accuracy	Precision	Recall	F1-Score	AUC
ELM	0.6948	0.588	0.200	0.298	0.6910
MSCOA-ELM	0.8377	0.833	0.660	0.737	0.7017
COA-ELM	0.7987	0.794	0.529	0.635	0.7064
DBO-ELM	0.7987	0.868	0.559	0.681	0.7067
SABO-ELM	0.8312	**0.892**	0.600	0.718	0.7055
GWO-ELM	0.7792	0.735	0.632	0.679	0.7087
WWPA-ELM	0.7597	0.757	0.500	0.602	0.7080
CDO-ELM	0.7597	0.757	0.500	0.602	**0.7141**
LPO-ELM	0.8312	0.780	0.653	0.713	0.6977
APO-ELM	**0.8442**	0.761	**0.729**	**0.743**	0.7049

Table 6. Comparison of performance measures of each model.

Data availability

The source code and datasets used in this study are available upon request by contacting the corresponding author at victoria@stu.sxufe.edu.c.

Received: 11 November 2024; Accepted: 8 July 2025
Published online: 09 August 2025

References

1. Hu, G., Guo, Y., Wei, G. & Abualigah, L. Genghis Khan shark optimizer: a novel nature-inspired algorithm for engineering optimization. *Adv. Eng. Inf.* **58**, 102210 (2023).
2. Abdollahzadeh, B. et al. Puma optimizer (PO): A novel metaheuristic optimization algorithm and its application in machine learning. *Clust Comput.* **27** 5235–5283 (2024).
3. Oyelade, O. N., Ezugwu, A. E. S., Mohamed, T. I. & Abualigah, L. Ebola optimization search algorithm: A new nature-inspired metaheuristic optimization algorithm. *IEEE Access***10**, 16150–16177 (2022).
4. Shishavan, S. T. & Gharehchopogh, F. S. An improved cuckoo search optimization algorithm with genetic algorithm for community detection in complex networks. *Multimed. Tools Appl.***81**, 25205–25231 (2022).
5. Li, J. et al. Application of XGBoost algorithm in the optimization of pollutant concentration. *Atmos. Res.* **276**, 106238 (2022).
6. Mohammadi, F. G., Amini, M. H. & Arabnia, H. R. Evolutionary computation, optimization, and learning algorithms for data science. Optim., Learn., Control Interdep. Complex Networks 37–65 (2020).
7. Yang, X. S. & Karamanoglu, M. Nature-inspired computation and swarm intelligence: a state-of-the-art overview. eds Yang, X. S. & Karamanoglu, M. Nature-Inspired Comput. Swarm Intell. 3–18 (Academic Press, London, 2020).
8. Pelusi, D. et al. Improving exploration and exploitation via a hyperbolic gravitational search algorithm. *Knowledge-Based Systems***193**, 105404 (2020).
9. Liu, H., Zhang, X., Liang, H. & Tu, L. Stability analysis of the human behavior-based particle swarm optimization without stagnation assumption. *Expert Syst. Appl.***159**, 113638 (2020).
10. Abdel-Basset, M., Mohamed, R., Jameel, M. & Abouhawwash, M. Spider wasp optimizer: A novel meta-heuristic optimization algorithm. *Artif. Intell. Rev.***56**, 11675–11738 (2023).
11. Akbari, M. A. et al. The cheetah optimizer: A nature-inspired metaheuristic algorithm for large-scale optimization problems. *Sci. Rep.***12**, 10953 (2022).
12. Agushaka, J. O., Ezugwu, A. E. & Abualigah, L. Dwarf mongoose optimization algorithm. *Comput. Methods Appl. Mech. Eng.***391**, 114570 (2022).
13. Xue, J. & Shen, B. Dung beetle optimizer: A new meta-heuristic algorithm for global optimization. *J. Supercomput.***79**, 7305–7336 (2023).
14. Agushaka, J. O., Ezugwu, A. E. & Abualigah, L. Gazelle optimization algorithm: A novel nature-inspired metaheuristic optimizer. *Neural Comput. Appl.***35**, 4099–4131 (2023).
15. Mirjalili, S., Mirjalili, S. M. & Lewis, A. Grey wolf optimizer. *Adv. Eng. Softw.***69**, 46–61 (2014).
16. Falahah, I. A. et al. Frilled Lizard optimization: A novelbio-inspired optimizer for solving engineering applications. *J. Artif. Intell. Rev.* **79**, 3631–3678 (2024).
17. Makhadmeh, S. N. et al. Recent advances in grey Wolf optimizer, its versions and applications. *IEEE Access.* **11**, 12345–12380 (2023).
18. Chhabra, A., Hussien, A. G. & Hashim, F. A. Improved bald eagle search algorithm for global optimization and feature selection. *Alexandria Eng. J.***68**, 141–180 (2023).
19. Pan, J. S. et al. A parallel compact gannet optimization algorithm for solving engineering optimization problems. *Mathematics***11**, 439 (2023).
20. Ikram, R. M. A. et al. A novel swarm intelligence: cuckoo optimization algorithm (COA) and sailfish optimizer (SFO) in landslide susceptibility assessment. *Stoch. Environ. Res. Risk Assess.* **37**, 1717–1743 (2023).
21. Houssein, E. H. & Sayed, A. Dynamic candidate solution boosted beluga whale optimization algorithm for biomedical classification. *Mathematics***11**, 707 (2023).
22. Jovanovic, D. et al. Tuning machine learning models using a group search firefly algorithm for credit card fraud detection. *Mathematics***10**, 2272 (2022).
23. Han, E. & Ghadimi, N. Model identification of proton-exchange membrane fuel cells based on a hybrid convolutional neural network and extreme learning machine optimized by improved honey Badger algorithm. *Sustain. Energy Technol. Assess.* **52**, 102005 (2022).
24. Ewees, A. A., Al-qaness, M. A., Abualigah, L. & Abd elaziz, M. HBO-LSTM: Optimized long short term memory with heap-based optimizer for wind power forecasting. *Energy Convers. Manag.***268**, 116022 (2022).
25. Ma, J. et al. A comprehensive comparison among metaheuristics (MHs) for geohazard modeling using machine learning: insights from a case study of landslide displacement prediction. *Eng. Appl. Artif. Intell.* **114**, 105150 (2022).
26. Jovanovic, A. et al. Particle swarm optimization tuned multi-headed long short-term memory networks approach for fuel prices forecasting. *J. Netw. Comput. Appl.* **233**, 104048 (2025).
27. Jia, H., Rao, H., Wen, C. & Mirjalili, S. Crayfish optimization algorithm. *Artif. Intell. Rev.* **56**, 1919–1979 (2023).
28. Fakhouri, H. N. et al. Novel hybrid crayfish optimization algorithm and self-adaptive differential evolution for solving complex optimization problems. *Symmetry***16**, 927 (2024).
29. Sorour, S. E. et al. An improved binary crayfish optimization algorithm for handling feature selection task in supervised classification. *Mathematics* **12**, 2364 (2024).
30. Sait, S. M., Mehta, P., Yıldız, A. R. & Yıldız, B. S. Optimal design of structural engineering components using artificial neural network-assisted crayfish algorithm. *Mater. Test.***66**, 1–10 (2024).
31. Shikoun, N. H., Al-Eraqi, A. S. & Fathi, I. S. BinCOA: An efficient binary crayfish optimization algorithm for feature selection. *IEEE Access***12**, 28621–28635 (2024).
32. AbdElminaam, D. S. et al. Parameters extraction of the Three-Diode photovoltaic model using crayfish optimization algorithm. *IEEE Access.* **12**, 1–15 (2024).
33. Sugitha, G. & Chaluvaraj, P. B. Self-Attention conditional generative adversarial network optimised with crayfish optimization algorithm for improving cyber security in cloud computing. *Comput. Secur.* **140**, 103773 (2024).
34. Chaib, L. et al. Improved crayfish optimization algorithm for parameters Estimation of photovoltaic models. *Energy Convers. Manag.* **313**, 118627 (2024).
35. Patel, V. V. et al. A systematic kidney tumor segmentation and classification framework using adaptive and Attentive-based deep learning networks with improved crayfish optimization algorithm. *IEEE Access.* **12**, 1–15 (2024).
36. Chauhan, S. et al. Parallel structure of crayfish optimization with arithmetic optimization for classifying the friction behaviour of Ti-6Al-4V alloy for complex machinery applications. *Knowl. -Based Syst.* **286**, 111389 (2024).
37. Yakubu, I. Z. & Murali, M. An efficient IoT-Fog-Cloud resource allocation framework based on Two-Stage approach. *IEEE Access.* **12**, 1–15 (2024).

38. Jia, W. et al. Research on bearing remaining useful life anti-noise prediction based on fusion of color-grayscale time-frequency features. *Meas. Sci. Technol.***35**, 123456 (2024).
39. Jiang, M. et al. Research on time-series based and similarity search based methods for PV power prediction. *Energy Convers. Manag*. **308**, 118391 (2024).
40. Huang, H., Gao, W. & Yang, G. Distribution transformer mechanical faults diagnosis method incorporating cross-domain feature extraction and recognition of unknown-type faults. *Measurement* **238**, 115270 (2024).
41. Almutairi, S. Z. et al. A hierarchical optimization approach to maximize hosting capacity for electric vehicles and renewable energy sources through demand response and transmission expansion planning. *Sci. Rep.***14**, 15765 (2024).
42. Zhong, R., Fan, Q., Zhang, C. & Yu, J. Hybrid remora crayfish optimization for engineering and wireless sensor network coverage optimization. *Clust Comput* **27** 10141–10168 (2024).
43. Zhang, Y., Liu, P. & Li, Y. Implementation of an enhanced crayfish optimization algorithm. *Biomimetics* **9**, 341 (2024).
44. Jia, H. et al. Modified crayfish optimization algorithm for solving multiple engineering application problems. *Artif. Intell. Rev.***57**, 127 (2024).
45. Wang, R., Zhang, S. & Zou, G. An improved multi-strategy crayfish optimization algorithm for solving numerical optimization problems. *Biomimetics***9**, 361 (2024).
46. Kouba, A., Petrusek, A. & Kozák, P. Continental-wide distribution of crayfish species in europe: update and maps. *Knowl. Manag Aquat. Ecosyst.* **413**, 05 (2014).
47. Allan, E. L., Froneman, P. W. & Hodgson, A. N. Effects of temperature and salinity on the standard metabolic rate (SMR) of the caridean shrimp *Palaemon peringueyi*. *J. Exp. Mar. Biol. Ecol.***337**, 103–108 (2006).
48. Larson, E. R. & Olden, J. D. The state of crayfish in the Pacific Northwest. *Fisheries* **36**, 60–73 (2011).
49. Bellman, K. L. & Krasne, F. B. Adaptive complexity of interactions between feeding and escape in crayfish. *Science* **221**, 779–781 (1983).
50. Gürses, D. et al. A novel hybrid water wave optimization algorithm for solving complex constrained engineering problems. *Mater. Test.* **63**, 560–564 (2021).
51. Cui, F. et al. Soil water content Estimation using ground penetrating radar data via group intelligence optimization algorithms: an application in the Northern Shaanxi coal mining area. *Energy Explor. Exploit.* **39**, 318–335 (2021).
52. Zhu, S. et al. A new one-dimensional compound chaotic system and its application in high-speed image encryption. *Appl. Sci.* **11**, 11206 (2021).
53. Guang-Yi, W. & Fang, Y. Cascade chaos and its dynamic characteristics. *Chin. Phys. B*. **22**, 050504 (2013).
54. Rahnamayan, S., Tizhoosh, H. R. & Salama, M. M. Opposition versus randomness in soft computing techniques. *Appl. Soft Comput.***8**, 906–918 (2008).
55. Tizhoosh, H. R. Opposition-based learning: a new scheme for machine intelligence. *Proc. CIMCA-IAWTIC'06*. **1**, 695–701 (2005).
56. Rahnamayan, S., Tizhoosh, H. R. & Salama, M. M. Quasi-oppositional differential evolution. *Proc. IEEE CEC* 2229–2236 (2007).
57. Trojovský, P. & Dehghani, M. Subtraction-average-based optimizer: A new swarm-inspired metaheuristic algorithm for solving optimization problems. *Biomimetics***8**, 149 (2023).
58. Bayraktar, Z., Komurcu, M. & Werner, D. H. Wind Driven Optimization (WDO): A novel nature-inspired optimization algorithm and its application to electromagnetics. *Proc. IEEE APS* 1–4 (2010).
59. Shehadeh, H. A. Chernobyl disaster optimizer (CDO): a novel meta-heuristic method for global optimization. *Neural Comput. Appl.* **35**, 10733–10749 (2023).
60. Ghasemi, M. et al. Optimization based on performance of lungs in body: lungs performance-based optimization (LPO). *Comput. Methods Appl. Mech. Eng.* **419**, 116582 (2024).
61. Wang, X. et al. Artificial protozoa optimizer (APO): A novel bio-inspired metaheuristic algorithm for engineering optimization. *Knowledge-Based Systems***295**, 111737 (2024).
62. Derrac, J. et al. A practical tutorial on the use of nonparametric statistical tests as a methodology for comparing evolutionary and swarm intelligence algorithms. *Swarm Evol. Comput.* **1**, 3–18 (2011).
63. Hashim, F. A. et al. Henry gas solubility optimization: A novel physics-based algorithm. *Future Gener Comput. Syst.* **101**, 646–667 (2019).
64. Huang, G. B., Ding, X. & Zhou, H. Optimization method based extreme learning machine for classification. *Neurocomputing* **74**, 155–163 (2010).
65. Banerjee, K. S. *Generalized Inverse of Matrices and its Applications* (Wiley, 1973).
66. Huang, G. B., Zhou, H., Ding, X. & Zhang, R. Extreme learning machine for regression and multiclass classification. *IEEE Trans. Syst. Man. Cybern. B***42**, 513–529 (2011).
67. Wang, Z., Qu, Q., Yu, G. & Kang, Y. Breast tumor detection in double views mammography based on extreme learning machine. *Neural Comput. Appl.* **27**, 227–240 (2016).
68. Wang, Z. et al. Breast tumor detection in digital mammography based on extreme learning machine. *Neurocomputing* **128**, 175–184 (2014).
69. Huang, G. B., Wang, D. H. & Lan, Y. Extreme learning machines: A survey. *Int. J. Mach. Learn. Cybern.***2**, 107–122 (2011).
70. Huang, G. B., Zhu, Q. Y. & Siew, C. K. Extreme learning machine: Theory and applications. *Neurocomputing***70**, 489–501 (2006).

Author contributions

Y.L. conceived the research, developed the Hybrid Cumulative Strategy-Enhanced Crayfish Optimization Algorithm (MSCOA), conducted the experiments, analyzed the data, and wrote the manuscript.

Declarations

Competing interests

The authors declare no competing interests.

Additional information

Correspondence and requests for materials should be addressed to Y.-J.L.

Reprints and permissions information is available at www.nature.com/reprints.

Publisher's note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.