Artificial Intelligence Review (2024) 57:127
https://doi.org/10.1007/s10462-024-10738-x

Modified crayfish optimization algorithm for solving multiple engineering application problems

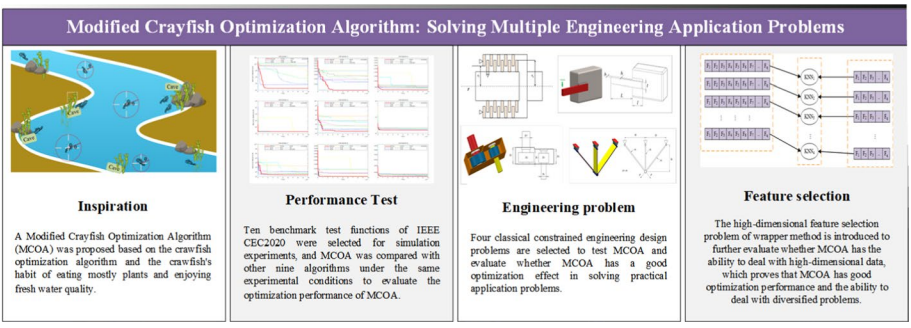
Heming Jia[1] · Xuelian Zhou[1] · Jinrui Zhang[1] · Laith Abualigah[2,3] · Ali Riza Yildiz[4] · Abdelazim G. Hussien[5]

Accepted: 24 February 2024

Abstract

Crayfish Optimization Algorithm (COA) is innovative and easy to implement, but the crayfish search efficiency decreases in the later stage of the algorithm, and the algorithm is easy to fall into local optimum. To solve these problems, this paper proposes an modified crayfish optimization algorithm (MCOA). Based on the survival habits of crayfish, MCOA proposes an environmental renewal mechanism that uses water quality factors to guide crayfish to seek a better environment. In addition, integrating a learning strategy based on ghost antagonism into MCOA enhances its ability to evade local optimality. To evaluate the performance of MCOA, tests were performed using the IEEE CEC2020 benchmark function and experiments were conducted using four constraint engineering problems and feature selection problems. For constrained engineering problems, MCOA is improved by 11.16%, 1.46%, 0.08% and 0.24%, respectively, compared with COA. For feature selection problems, the average fitness value and accuracy are improved by 55.23% and 10.85%, respectively. MCOA shows better optimization performance in solving complex spatial and practical application problems. The combination of the environment updating mechanism and the learning strategy based on ghost antagonism significantly improves the performance of MCOA. This discovery has important implications for the development of the field of optimization.

Graphical Abstract



Extended author information available on the last page of the article

Published online: 24 April 2024

Keywords Crayfish Optimization Algorithm · Environmental updating mechanism · Ghost opposition-based learning strategy · Global optimization problem · Constrained engineering design problems · High dimensional feature selection

1 Introduction

For a considerable period, engineering application problems have been widely discussed by people. At present, improving the modern scientific level of engineering construction has become the goal of human continuous struggle, including constrained engineering design problems (Zhang et al. 2022a; Mortazavi 2019) affected by a series of external factors and feature selection problems (Kira and Rendell 1992), and so on. Constrained engineering design problems refers to the problem of achieving optimization objectives and reducing calculation costs under many external constraints, which is widely used in mechanical engineering (Abualigah et al. 2022), electrical engineering (Razmjooy et al. 2021), civil engineering (Kaveh 2017), chemical engineering (Talatahari et al. 2021) and other engineering fields, such as workshop scheduling (Meloni et al. 2004), wind power generation (Lu et al. 2021), and UAV path planning (Belge et al. 2022), parameter extraction of photovoltaic models(Zhang et al. 2022b; Zhao et al. 2022), Optimization of seismic foundation isolation system (Kandemir and Mortazavi 2022), optimal design of RC support foundation system of industrial buildings (Kamal et al. 2023), synchronous optimization of fuel type and external wall insulation performance of intelligent residential buildings (Moloodpoor and Mortazavi 2022), economic optimization of double-tube heaters (Moloodpoor et al. 2021).

Feature selection is the process of choosing specific subsets of features from a larger set based on defined criteria. In this approach, each original feature within the subset is individually evaluated using an assessment function. The aim is to select pertinent features that carry distinctive characteristics. This selection process reduces the dimensionality of the feature space, enhancing the model's generalization ability and accuracy. The ultimate goal is to create the best possible combination of features for the model. By employing feature selection, the influence of irrelevant factors is minimized. This reduction in irrelevant features not only streamlines the computational complexity but also reduces the time costs associated with processing the data. Through this method, redundant and irrelevant features are systematically removed from the model. This refinement improves the model's accuracy and results in a higher degree of fit, ensuring that the model aligns more closely with the underlying data patterns.

In practical applications of feature selections, models are primarily refined using two main methods: the filter (Cherrington et al. 2019) and wrapper (Jović et al. 2015) techniques. The filter method employs a scoring mechanism to assess and rank the model's features. It selects the subset of features with the highest scores, considering it as the optimal feature combination. On the other hand, the wrapper method integrates the selection process directly into the learning algorithm. It embeds the feature subset evaluation within the learning process, assessing the correlation between the chosen features and the model. In recent years, applications inspired by heuristic algorithms can be seen everywhere in our lives and are closely related to the rapid development of today's society. These algorithms play an indispensable role in solving a myriad of complex engineering problems and feature selection challenges. They have proven particularly effective in addressing

spatial, dynamic, and random problems, showcasing significant practical impact and tangible outcomes.

With the rapid development of society and science and technology, through continuous exploitation and exploration in the field of science, more and more complex and difficult to describe multi-dimensional engineering problems also appear in our research process. Navigating these complexities demands profound contemplation and exploration. While traditional heuristic algorithms have proven effective in simpler, foundational problems, they fall short when addressing the novel and intricate multi-dimensional challenges posed by our current scientific landscape and societal needs. Thus, researchers have embarked on a journey of continuous contemplation and experimentation. By cross-combining and validating existing heuristic algorithms, they have ingeniously devised a groundbreaking solution: Metaheuristic Algorithms (MAs) (Yang 2011). This innovative approach aims to tackle the complexities of our evolving problems, ensuring alignment with the rapid pace of social and technological development. MAs is a heuristic function based algorithm. It works by evaluating the current state of the problem and possible solutions to guide the algorithm in making choices in the search space. MAs improves the efficiency and accuracy of the problem solving process by combining multiple heuristic functions and updating the search direction at each step based on their weights. The diversity of MAs makes it a universal problem solver, adapting to the unique challenges presented by different problem domains. Essentially represents a powerful paradigm shift in computational problem solving, providing a powerful approach to address the complexity of modern engineering and scientific challenges. Compared with traditional algorithms, MAs has made great progress in finding optimal solutions, jumping out of local optima, and overcoming convergence difficulties in the later stage of solution through the synergy of different algorithms. These enhancements mark a significant progress, which not only demonstrates the adaptability of the scientific method, but also emphasizes the importance of continuous research and cooperation. It also has the potential to radically solve problems in domains of complex engineering challenges, enabling researchers to navigate complex problem landscapes with greater accuracy and efficiency.

Research shows that MAs are broadly classified into four different research directions: swarm-based, natural evolution-based, human-based, and physics-based. These categories include a wide range of innovative problem-solving approaches, each drawing inspiration from a different aspect of nature, human behavior, or physical principles. Researchers exploration these different pathways to solve complex challenges and optimize the solutions efficiently. First of all, the swarm-based optimization algorithm is the optimization algorithm that uses the wisdom of population survival to solve the problem. For example, Particle Swarm Optimization Algorithm (PSO) (Wang et al. 2018a) is an optimization algorithm based on the group behavior of birds. PSO has a fast search speed and is only used for real-valued processing. However, it is not good at handling discrete optimization problems and has fallen into local optimization. Artificial Bee Colony Optimization Algorithm (ABC) (Jacob and Darney 2021) realizes the sharing and communication of information among individuals when bees collect honey according to their respective division of labor. In the Salp Swarm Algorithm (SSA) (Mirjalili et al. 2017), individual sea squirts are connected end to end and move and prey in a chain, and follow the leader with followers according to a strict "hierarchical" system. Ant Colony Optimization Algorithm (ACO) (Dorigo et al. 2006), ant foraging relies on the accumulation of pheromone on the path, and spontaneously finds the optimal path in an organized manner.

Secondly, a natural evolutionary algorithm inspired by the law of group survival of the fittest, an optimization algorithm that finds the best solution by preserving the

characteristics of easy survival and strong individuals, such as: Genetic Programming Algorithm (GP) (Espejo et al. 2009), because biological survival and reproduction have certain natural laws, according to the structure of the tree to deduce certain laws of biological genetic and evolutionary process. Evolutionary Strategy Algorithm (ES) (Beyer and Schwefel 2002), the ability of a species to evolve itself to adapt to the environment, and produce similar but different offspring after mutation and recombination from the parent. Differential Evolution (DE) (Storn and Price 1997) eliminates the poor individuals and retains the good ones in the process of evolution, so that the good ones are constantly approaching the optimal solution. It has a strong global search ability in the initial iteration, but when there are fewer individuals in the population, individuals are difficult to update, and it is easy to fall into the local optimal. The Biogeography-based Optimization Algorithm (BBO) (Simon 2008), influenced by biogeography, filters out the global optimal value through the iteration of the migration and mutation of species information.

Then, Human-based optimization algorithms are optimization algorithms that take advantage of the diverse and complex human social relationships and activities in a specific environment to solve problems, such as: The teaching–learning-based Optimization (TLBO) (Rao and Rao 2016) obtained the optimal solution by simulating the Teaching relationship between students and teachers. It simplifies the information sharing mechanism within each round, and all evolved individuals can converge to the global optimal solution faster, but the algorithm often loses its advantage when solving some optimization problems far from the origin. Coronavirus Mask Protection Algorithm (CMPA) (Yuan et al. 2023), which is mainly inspired by the self-protection process of human against coronavirus, establishes a mathematical model of self-protection behavior and solves the optimization problem. Cultural Evolution Algorithm (CEA) (Kuo and Lin 2013), using the cultural model of system thinking framework for exploitation to achieve the purpose of cultural transformation, get the optimal solution. Volleyball Premier League Algorithm (VPL) (Moghdani and Salimifard 2018) simulates the process of training, competition and interaction of each team in the volleyball game to solve the global optimization problem.

Finally, Physics-based optimization algorithm is an optimization algorithm that uses the basic principles of physics to simulate the physical characteristics of particles in space to solve problems. For example, Snow Ablation Algorithm (SAO) (Deng and Liu 2023), inspired by the physical reaction of snow in nature, realizes the transformation among snow, water and steam by simulating the sublation and ablation of snow. RIME Algorithm (RIME) (Su et al. 2023) is a exploration and exploitation of mathematical model balance algorithm based on the growth process of soft rime and hard rime in nature. Central Force Optimization Algorithm (CFO) (Formato 2007), aiming at the problem of complex calculation of the initial detector, a mathematical model of uniform design is proposed to reduce the calculation time. Sine and cosine algorithm (SCA) (Mirjalili 2016) establishes mathematical models and seeks optimal solutions based on the volatility and periodicity characteristics of sine and cosine functions. Compared with the candidate solution set of a certain scale, the algorithm has a strong search ability and the ability to jump out of the local optimal, but the results of some test functions fluctuate around the optimal solution, and there is a certain precocious situation, and the convergence needs to be improved.

While the original algorithm is proposed, many improved MAs algorithms are also proposed to further improve the optimization performance of the algorithm in practical application problems, such as: Yujun-Zhang et al. combined the arithmetic optimization algorithm (AOA) with the Aquila Optimizer(AO) algorithm to propose a new meta-heuristic algorithm (AOAAO) (Zhang et al. 2022c). CSCAHHO algorithm (Zhang et al. 2022d) is a new algorithm obtained by chaotic mixing of sine and cosine algorithm (SCA) and Harris

Hqwk optimization algorithm (HHO). Based on LMRAOA algorithm proposed to solve numerical and engineering problems (Zhang et al. 2022e). Yunpeng Ma et al. proposed an improved teaching-based optimization algorithm to artificially reduce NOx emission concentration in circulating fluidized bed boilers (Ma et al. 2021). The improved algorithm SOS(MSOS) (Kumar et al. 2019), based on the natural Symbiotic search (SOS) algorithm, improves the search efficiency of the algorithm by introducing adaptive return factors and modified parasitic vectors. Modified beluga whale optimization with multi-strategies for solving engineering problems (MBWO) (Jia et al. 2023a) by gathering Beluga populations for feeding and finding new habitats during long-distance migration. Betul Sultan Yh-ld-z et al. proposed a novel hybrid optimizer named AO-NM, which aims to optimize engineering design and manufacturing problems (Yıldız et al. 2023).

The Crayfish Optimization Algorithm (COA) (Jia et al. 2023b) is a novel metaheuristic algorithm rooted in the concept of population survival wisdom, introduced by Heming Jia et al. in 2023. Drawing inspiration from crayfish behavior, including heat avoidance, competition for caves, and foraging, COA employs a dual-stage strategy. During the exploration stage, it replicates crayfish searching for caves in space for shelter, while the exploitation stage mimics their competition for caves and search for food. Crayfish, naturally averse to dry heat, thrive in freshwater habitats. To simulate their behavior and address challenges related to high temperatures and food scarcity, COA incorporates temperature variations into its simulation. By replicating crayfish habits, the algorithm dynamically adapts to environmental factors, ensuring robust problem-solving capabilities. Based on temperature fluctuations, crayfish autonomously select activities such as seeking shelter, competing for caves, and foraging. When the temperature exceeds 30°C, crayfish instinctively seek refuge in cool, damp caves to escape the heat. If another crayfish is already present in the cave, a competition ensues for occupancy. Conversely, when the temperature drops below 30°C, crayfish enter the foraging stage. During this phase, they make decisions about food consumption based on the size of the available food items. COA achieves algorithmic transformation between exploration and exploitation stages by leveraging temperature variations, aiming to balance the exploration and exploitation capabilities of the algorithm. However, COA solely emulates the impact of temperature on crayfish behavior, overlooking other significant crayfish habits, leading to inherent limitations. In the latter stages of global search, crayfish might cluster around local optimum positions, restricting movement. This hampers the crayfish's search behavior, slowing down convergence speed, and increasing the risk of falling into local optima, thereby making it challenging to find the optimal solution.

In response to the aforementioned challenges, this paper proposes a Modified Crayfish Optimization Algorithm (MCOA). MCOA introduces an environmental update mechanism inspired by crayfish's preference for living in fresh flowing water. MCOA incorporates crayfish's innate perception abilities to assess the quality of the surrounding aquatic environment, determining whether the current habitat is suitable for survival. The simulation of crayfish crawling upstream to find a more suitable aquatic environment is achieved by utilizing adaptive flow factors and leveraging the crayfish's second, third foot perceptions to determine the direction of water flow.This method partially replicates the survival and reproduction behavior of crayfish, ensuring the continual movement of the population. It heightens the randomness within the group, widens the search scope for crayfish, enhances the algorithm's exploration efficiency, and effectively strengthens the algorithm's global optimization capabilities. Additionally, the ghost opposition-based learning strategy (Jia et al. 2023c) is implemented to introduce random population initialization when the algorithm becomes trapped in local optima.

This enhancement significantly improves the algorithm's capability to escape local optima, promoting better exploration of the solution space. After the careful integration of the aforementioned two strategies, the search efficiency and predation speed of the crayfish algorithm experience a substantial improvement. Moreover, the algorithm's convergence rate and global optimization ability are significantly enhanced, leading to more effective and efficient problem-solving capabilities.

In the experimental section, we conducted a comprehensive comparison between MCOA and nine other metaheuristic algorithms. We utilized the IEEE CEC2020 benchmark function to evaluate the performance of the algorithm. The evaluation involved statistical methods such as the Wilcoxon rank sum test and Friedman test to rank the averages, validating the efficiency of the MCOA algorithm and the effectiveness of the proposed improvements. Furthermore, MCOA was applied to address four constrained engineering design problems as well as the high-dimensional feature selection problem using the wrapper method. These practical applications demonstrated the practicality and effectiveness of MCOA in solving real-world engineering problems.

The main contributions of this paper are as follows:

- In the environmental renewal mechanism, the water quality factor and roulette wheel selection method are introduced to simulate the process of crayfish searching for a more suitable water environment for survival.
- The introduction of the ghost opposition-based learning strategy enhances the randomness of crayfish update locations, effectively preventing the algorithm from getting trapped in local optima, and improving the overall global optimization performance of the algorithm.
- The fixed value of food intake is adaptively adjusted based on the number of evaluations, enhancing the algorithm's capacity to escape local optima. This adaptive change ensures a more dynamic exploration of the solution space, improving the algorithm's overall optimization effectiveness.
- The MCOA's performance is compared with nine metaheuristics, including COA, using the IEEE CEC2020 benchmark function. The comparison employs the Wilcoxon rank sum test and Friedman test to rank the averages, providing evidence for the efficiency of MCOA and the effectiveness of the proposed improvements.
- The application of MCOA to address four constrained engineering design problems and the high-dimensional feature selection problem using the wrapper method demonstrates the practicality and effectiveness of MCOA in real-world applications.

The main structure of this paper is as follows, the first part of the paper serves as a brief introduction to the entire document, providing an overview of the topics and themes that will be covered. In the second part, the paper provides a comprehensive summary of the Crayfish Optimization Algorithm (COA). In the third part, a modified crawfish optimization algorithm (MCOA) is proposed. By adding environment updating mechanism and ghost opposition-based learning strategy, MCOA can enhance the global search ability and convergence speed to some extent. Section four shows the experimental results and analysis of MCOA in IEEE CEC2020 benchmark functions. The fifth part applies MCOA to four kinds of constrained engineering design problems. In Section six, MCOA is applied to the high-dimensional feature selection problem of wrapper methods to demonstrate the effectiveness of MCOA in practical application problems. Finally, Section seven concludes the paper.

2 Crayfish optimization algorithm (COA)

Crayfish is a kind of crustaceans living in fresh water, its scientific name is crayfish, also called red crayfish or freshwater crayfish, because of its food, fast growth rate, rapid migration, strong adaptability and the formation of absolute advantages in the ecological environment. Changes in temperature often cause changes in crayfish behavior. When the temperature is too high, crayfish choose to enter the cave to avoid the damage of high temperature, and when the temperature is suitable, they will choose to climb out of the cave to forage. According to the living habits of crayfish, it is proposed that the three stages of summer, competition for caves and going out to forage correspond to the three living habits of crayfish, respectively.

Crayfish belong to ectotherms and are affected by temperature to produce behavioral differences, which range from 20 °C to 35 °C. The temperature is calculated as follows:

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

where *temp* represents the temperature of the crayfish's environment.

2.1 Initializing the population

In the *d*-dimensional optimization problem of COA, each crayfish is a $1 \times d$ matrix representing the solution of the problem. In a set of variables (X_1, X_2, X_3…… X_{d}), the position (X) of each crayfish is between the upper boundary (ub) and lower boundary (lb) of the search space. In each evaluation of the algorithm, an optimal solution is calculated, and the solutions calculated in each evaluation are compared, and the optimal solution is found and stored as the optimal solution of the whole problem. The position to initialize the crayfish population is calculated using the following formula.

$$X_{i,j} = lb_j + (ub_j - lb_j) \times rand \tag{2}$$

where $X_{i,j}$ denotes the position of the *i-th* crayfish in the *j-th* dimension, ub_{j} denotes the upper bound of the *j-th* dimension, lb_{j} denotes the lower bound of the *j-th* dimension, and *rand* is a random number from 0 to 1.

2.2 Summer escape stage (exploration stage)

In this paper, the temperature of 30 °C is assumed to be the dividing line to judge whether the current living environment is in a high temperature environment. When the temperature is greater than 30 ℃ and it is in the summer, in order to avoid the harm caused by the high temperature environment, crayfish will look for a cool and moist cave and enter the summer to avoid the influence of high temperature. The caverns are calculated as follows.

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

where X_G represents the optimal position obtained so far for this evaluation number, and X_L represents the optimal position of the current population.

The behavior of crayfish competing for the cave is a random event. To simulate the random event of crayfish competing for the cave, a random number rand is defined, when

rand < 0.5 means that there are no other crayfish currently competing for the cave, and the crayfish will go straight into the cave for the summer. At this point, the crayfish position update calculation formula is as follows.

$$Xnew = X_{i,j} + C_2 \times rand \times (X_{shade} - X_{i,j}) \tag{4}$$

Here, X_{new} is the next generation position after location update, and C_2 is a decreasing curve. C_2 is calculated as follows.

$$C_2 = 2 - (FEs/MaxFEs) \tag{5}$$

Here, *FEs* represents the number of evaluations and *MaxFEs* represents the maximum number of evaluations.

2.3 Competition stage (exploitation stage)

When the temperature is greater than 30 °C and rand ≥ 0.5, it indicates that the crayfish have other crayfish competing with them for the cave when they search for the cave for summer. At this point, the two crayfish will struggle against the cave, and crayfish X_i adjusts its position according to the position of the other crayfish X_z. The adjustment position is calculated as follows.

$$Xnew = X_{i,j} - X_{z,j} + X_{shade} \tag{6}$$

Here, z represents the random individual of the crayfish, and the random individual calculation formula is as follows.

$$z = round(rand \times (N - 1)) + 1 \tag{7}$$

where, N is the population size.

2.4 Foraging stage (exploitation stage)

The foraging behavior of crayfish is affected by temperature, and temperature less than or equal to 30 ℃ is an important condition for crayfish to climb out of the cave to find food. When the temperature is less than or equal to 30 °C, the crayfish will drill out of the cave and judge the location of the food according to the optimal location obtained in this evaluation, so as to find the food to complete the foraging. The position of the food is calculated as follows.

$$X_{food} = X_G \tag{8}$$

The amount of food crayfish eat depends on the temperature. When the temperature is between 20 °C and 30°C, crayfish have strong foraging behavior, and the most food is found and the maximum food intake is also obtained at 25 °C. Thus, the food intake pattern of crayfish resembles a normal distribution. Food intake was calculated as follows.

$$p = C_1 \times \frac{1}{\sqrt{2 \times \pi} \times \sigma} \times \exp(-\frac{(temp - \mu)^2}{2\sigma^2}) \tag{9}$$

Here, μ is the most suitable temperature for crayfish feeding, and σ and C_1 are the parameters used to control the variation of crayfish intake at different temperatures.

The food crayfish get depends not only on the amount of food they eat, but also on the size of the food. If the food is too large, the crayfish can't eat the food directly. They need to tear it up with their claws before eating the food. The size of the food is calculated as follows.

$$Q = C_3 \times rand \times (fitness_i / fitness_{food}) \tag{10}$$

Here, C_3 is the food factor, which represents the largest food, and its value is 3, $fitness_i$ represents the fitness value of the *i-th* crayfish, and $fitness_{food}$ represents the fitness value of the location of the food.

Crayfish use the value of the maximum food Q to judge the size of the food obtained and thus decide the feeding method. When $Q > (C_3 + 1)/2$, it means that the food is too large for the crayfish to eat directly, and it needs to tear the food with its claws and eat alternately with the second and third legs. The formula for shredding food is as follows.

$$X_{food} = \exp\left(-\frac{1}{Q}\right) \times X_{food} \tag{11}$$

After the food is shredded into a size that is easy to eat, the second and third claws are used to pick up the food and put it into the mouth alternately. In order to simulate the process of bipedal eating, the mathematical models of sine function and cosine function are used to simulate the crayfish eating alternately. The formula for crayfish alternating feeding is as follows.

$$X_{new} = X_{i,j} + X_{food} \times p \times (\cos(2 \times \pi \times rand) - \sin(2 \times \pi \times rand)) \tag{12}$$

When $Q \leq (C_3 + 1)/2$, it indicates that the food size is suitable for the crayfish to eat directly at this time, and the crayfish will directly move towards the food location and eat directly. The formula for direct crayfish feeding is as follows.

$$X_{new} = (X_{i,j} - X_{food}) \times p + p \times rand \times X_{i,j} \tag{13}$$

2.5 Pseudo-code for COA

Algorithm 1 Crayfish optimization algorithm pseudo-code

Initialization iterations T, population N, dimension dim

Randomly generate an initial population

Calculate the fitness value of the population to get X_G, X_L

While $t<T$

  Defining temperature temp by Eq. (1).

  If $temp>30$

    Define cave Xshade according to Eq. (3).

    If $rand<0.5$

      Crayfish conducts the summer resort stage according toEq. (4).

    Else

      Crayfish compete for caves through Eq. (6).

  End

Else

  The food intake p and food size Q are obtained by Eq. (9) and Eq. (10).

    If $Q>2$

     Crayfish shreds food by Eq. (11).

     Crayfish foraging according to Eq. (12).

    Else

     Crayfish foraging according to Eq. (13).

     End

  End

  Update fitness values, X_G, X_L

  $t=t+1$

End

3 Modified crayfish optimization algorithm (MCOA)

Based on crayfish optimization algorithm, we propose a high-dimensional feature selection problem solving algorithm (MCOA) based on improved crayfish optimization algorithm. In MCOA, we know that the quality of the aquatic environment has a great impact on the survival of crayfish, according to the living habits of crayfish, which mostly feed on plants and like fresh water. Oxygen is an indispensable energy for all plants and animals to maintain life, the higher the content of dissolved oxygen in the water body, the more vigorous the feeding of crayfish, the faster the growth, the less disease, and the faster the water flow in the place of better oxygen permeability, more aquatic plants, suitable for survival, so crayfish has a strong hydrotaxis. When crayfish perceive that the current environment is too dry and hot or lack of food, they crawl backward according to their second, third and foot perception (r) to judge the direction of water flow, and find an aquatic environment with sufficient oxygen and food to sustain life. Good aquatic environment has sufficient oxygen and abundant aquatic plants, to a certain extent, to ensure the survival and reproduction of crayfish.

In addition, we introduce ghost opposition-based learning to help MCOA escape the local optimal trap. The ghost opposition-based learning strategy combines the candidate individual, the current individual and the optimal individual to randomly generate a new candidate position to replace the previous poor candidate position, and then takes the best point or the candidate solution as the central point, and then carries out more specific and extensive exploration of other positions. Traditional opposition-based learning (Mahdavi et al. 2018) is based on the central point and carries out opposition-based learning in a fixed format. Most of the points gather near the central point and their positions will not exceed the distance between the current point and the central point, and most solutions will be close to the optimal individual. However, if the optimal individual is not near the current exploration point, the algorithm will fall into local optimal and it is difficult to find the optimal solution. Compared with traditional opposition-based learning, ghost opposition-based learning is a opposition-based learning solution

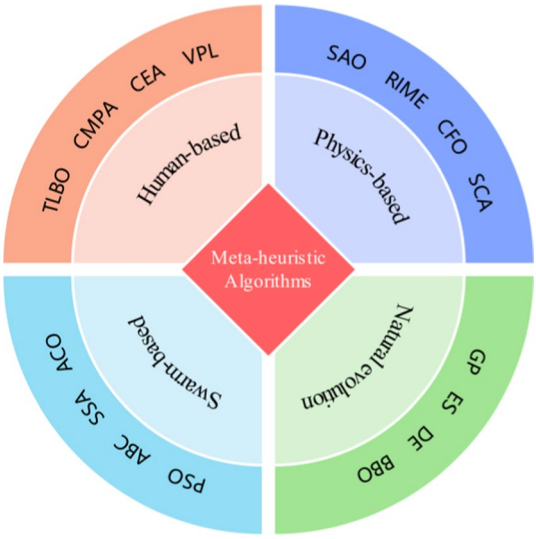


Fig. 1 Classification of MAs

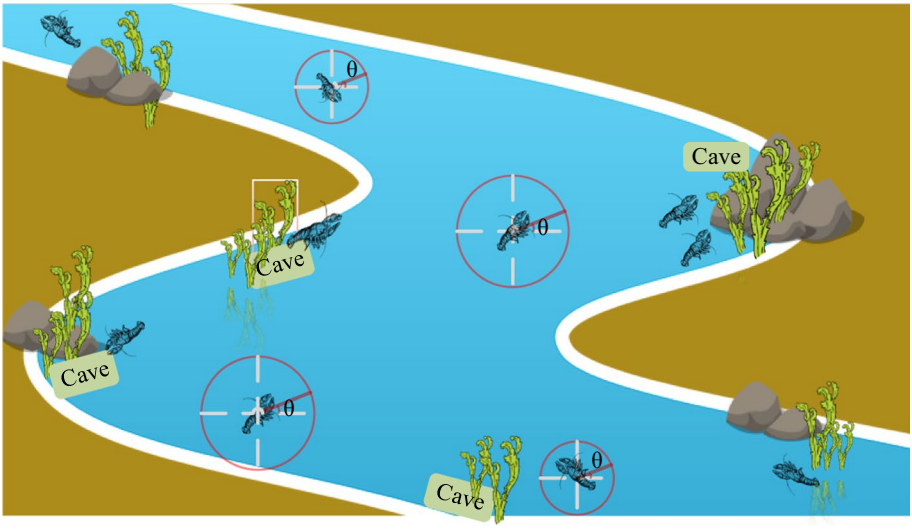
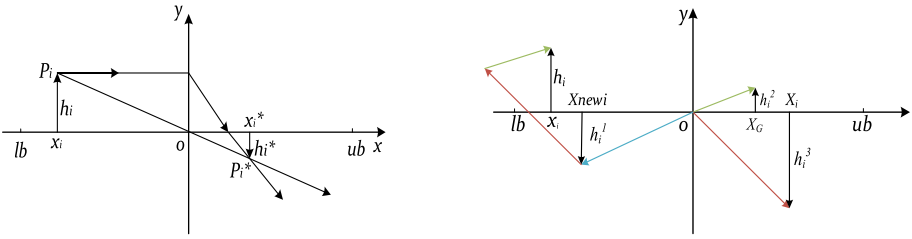


Fig. 2 Schematic diagram of the environment update mechanism



(a)Opposition-based Learning (b)Ghost Opposition-based Learning

Fig. 3 Schematic diagram of ghost opposition-based learning strategy

that can be dynamically changed by adjusting the size of parameter k, thereby expanding the algorithm's exploration range of space, effectively solving the problem that the optimal solution is not within the search range based on the center point, and making the algorithm easy to jump out of the local optimal.

According to the life habits of crayfish, this paper proposes a Modified Crayfish Optimization Algorithm (MCOA), which uses environment update mechanism and ghost opposition-based learning strategy to improve COA, and shows the implementation steps, pseudo-code and flow chart of MCOA algorithm as follows.

3.1 Environment update mechanism

In the environmental renewal mechanism, a water quality factor V is introduced to represent the quality of the aquatic environment at the current location. In order to simplify

the design and computational complexity of the system, the water quality factor V of the MCOA is represented by a hierarchical discretization, and its value range is set to 0 to 5. Crayfish perceive the quality of the current aquatic environment through the perception (r) of the second and third legs, judge whether the current living environment can continue to survive through the perception, and independently choose whether to update the current location. The location update is calculated as follows.

$$X_{new} = X_2 + (X_1 - X_{i,j}) \times \cos(\theta) \times B \times V \times rand + X_1 \times \sin(\theta) \times B \times rand \tag{14}$$

Among them, each crayfish has a certain difference in its own perception of water environment r, X_2 is a random position between the candidate optimal position and the current position, which is calculated by Eq. (15), X_1 is a random position in the population, and B is an adaptive water flow factor, which is calculated by Eq. (16).

$$X2 = (Xbest - X_{i,j}) \times r \tag{15}$$

$$B = c \times \cos\left(\frac{\pi}{2} \times \left(1 - \frac{FEs}{MaxFEs}\right)\right) \tag{16}$$

Among them, the sensing force r of the crayfish's second and third legs is a random number [0,1]. c is a constant that represents the water flow velocity factor with a value of 2. When $V \leq 3$, it indicates that the crayfish perceives the quality of the current living environment to be good and is suitable for continued survival. When $V > 3$, it indicates that the crayfish perceives that the current living environment quality is poor, and it needs to crawl in the opposite direction according to the direction of water flow that crayfish perceives, so as to find an aquatic environment with sufficient oxygen and abundant food Fig. 1.

In the environmental updating mechanism, in order to describe the behavior of crayfish upstream in more detail, the perception area of crayfish itself is abstractly defined as a circle in MCOA, and crayfish is in the center of the circle. In each evaluation calculation, a random Angle θ is first calculated by the roulette wheel selection algorithm to determine the moving direction of the crayfish in the circular area, and then the moving path of the crayfish is determined according to the current moving direction. In the whole circle, random angles can be chosen from 0 to 360 degrees, from which the value of θ can be determined to be of magnitude [− 1,1]. The difference of random Angle indicates that each crayfish moves its position in a random direction, which broadens the search range of crayfish, enhances the randomness of position and the ability to escape from local optimum, and avoids local convergence Fig. 2.

3.2 Ghost opposition-based learning strategy

The ghost opposition-based learning strategy takes a two-dimensional space as an example. It is assumed that there is a two-dimensional space, as shown in Fig. 3. On the X-axis, [*ub*, *lb*] represents the search range of the solution, and the ghost generation method is shown in Fig. 3. Assuming that the position of a new candidate solution is *Xnew* and the height of the solution is *h1 i*, the position of the best solution on the X-axis is the projected position of the candidate solution, and the position and height are *XG*, *h2 i*, respectively. In addition, on the X-axis there is a projection position X_i of the candidate solution with a height of *h3 i*.Thus, the position of the ghost is obtained. The projection position of the ghost on

the X-axis is x_i by vector calculation, and its height is h_i. The ghost position is calculated using the following formula.

$$x_i = X_{newi} - X_i + XG \tag{17}$$

In Fig. 3, the Y-axis represents the convex lens. Suppose there is a ghost position P_i, where x_i is its projection on the X-axis and h_i is its height. $P^* i$ is the real image obtained by convex lens imaging. $P^* i$ is projected on the X-axis as $x^* i$ and has height $h^* i$. Therefore, the opposite individual $x^* i$ of individual x_i can be obtained. $x^* i$ is the corresponding point corresponding to the ghost individual x_i obtained from O as the base point. According to the lens imaging principle, we can obtain Eq. (18), and the calculation formula is as follows.

$$k_i = \frac{h_i}{h_i^*} = \frac{(ub + lb)/2 - x_i}{x_i^* - (ub + lb)/2}, \tag{18}$$

The strategy formula of ghost opposition-based learning is evolved from Eq. (18). The strategy formula of ghost opposition-based learning is calculated as follows.

$$X_i^* = \frac{ub + lb}{2} + \frac{ub + lb}{2k} - \frac{X_i}{k} \tag{19}$$

$$k = (1 + (t/T)^{0.5})^{10} \tag{20}$$

3.3 Implementation of MCOA algorithm

3.3.1 Initialization phase

Initialize the population size N, the population dimension d, and the number of evaluations FEs. The initialized population is shown in Eq. (2).

3.3.2 Environment update mechanism

Crayfish judge the quality of the current aquatic environment according to the water quality factor V, and speculate whether the current aquatic environment can continue to survive. When $V > 3$ indicates that the crawfish perceives the quality of the current aquatic environment as poor and is not suitable for survival. According to the sensory information of the second and third legs and the adaptive flow factor, the crawfish judges the direction of the current flow, and then moves upstream to find a better aquatic environment to update the current position. The position update formula is shown in Eq. (14). When $V < 3$, it means that the crayfish has a good perception of the current living environment and is suitable for survival, and does not need to update its position.

3.3.3 Exploration phase

When the temperature is greater than 30 ℃ and $V > 3$, it indicates that crayfish perceive the current aquatic environment quality is poor, and the cave is dry and without moisture,

Table 1 Complexity analysis results of the eight algorithms

CEC	MCOA	COA	ROA	STOA	AOA	HHO	PDO	GA
F1	50.25427	42.06643	117.50879	78.69803	61.34279	190.05023	558.40814	46.75556
F2	52.47411	42.15295	132.87335	83.67219	63.75562	228.31452	545.78267	46.87563
F3	52.20520	34.79025	121.86170	77.28263	63.30686	209.25259	550.62068	42.79461
F4	41.56454	32.77783	127.68574	75.06781	58.81640	204.59109	531.71144	42.96858
F5	44.18214	40.45754	116.32387	73.13517	54.93385	201.21176	561.77099	42.35488
F6	44.31313	35.36359	118.61792	77.25077	57.87205	201.93613	563.40438	45.41542
F7	48.32919	44.74684	124.51911	85.37758	63.19400	210.17520	558.04406	48.82333
F8	58.45609	37.98938	155.53463	92.00123	59.56738	284.79159	548.82226	43.74182
F9	60.93452	34.90186	158.42571	71.52983	55.55031	288.45261	554.60660	40.14132
F10	40.67619	27.66778	138.68413	65.99252	50.03648	267.35518	548.90780	32.67195

which cannot achieve the effect of summer vacation. It is necessary to first update the position by crawling in the reverse direction according to the flow direction, and find a cool and moist cave in a better quality aquatic environment for summer.

3.3.4 Exploitation stage

When the temperature is less than 30 ℃ and $V > 3$, it indicates that crayfish perceive the current aquatic environment is poor, and there is not enough food to maintain the survival of crayfish. It is necessary to escape from the current food shortage living environment by crawling in the reverse direction according to the current direction, and find a better aquatic environment to maintain the survival and reproduction of crayfish.

3.3.5 Ghost opposition-based learning strategy

Through the combination of the candidate individual, the current individual and the optimal individual, a candidate solution is randomly generated and compared with the current solution, the better individual solution is retained, the opposite individual is obtained, and the location of the ghost is obtained. The combination of multiple positions effectively prevents the algorithm from falling into local optimum, and the specific implementation formula is shown in Eq. (19).

3.3.6 Update the location

The position of the update is determined by comparing the fitness values. If the fitness of the current individual update is better, the current individual replaces the original individual. If the fitness of the original individual is better, the original individual is retained to exist as the optimal solution.

The pseudocode for MCOA is as follows (Algorithm 2).

Algorithm 2 Modified Crayfish optimization algorithm pseudo-code

Initialize the parameters and population

Calculate the fitness of the initialized population and find the best individual (*FEs*=*FEs*+1)

WHILE(*FEs*<=*MaxFEs*)

Update temp, C_2, and other parameters

FOR *i*=1 to *N*

FOR *j*=1 to *d*

IF *V*>3 (Poor water quality, enter the environment update mechanism.)

The better survival position is found according to Eq. (14)

ELSE

IF *temp*>30

IF *rand*<0.5

According to Eq. (4), find a suitable summer location

ELSE

The summer vacation position is obtained by competition according to Eq. (6)

END IF

ELSE

IF *Q*>(C3+1)/2

The new position is obtained by feeding according to Eq. (12)

ELSE

The new position is obtained by feeding according to Eq. (13)

END IF

END IF

END IF

END FOR

END FOR

Calculate the k value according to Eq. (17)

FOR *i*=1 to *N*

The ghost opposition-based learning strategy is performed according to Eq. (20)

END FOR

END WHILE

Calculate the fitness of the updated population and find the best individual (*FEs*=*FEs*+1)

END

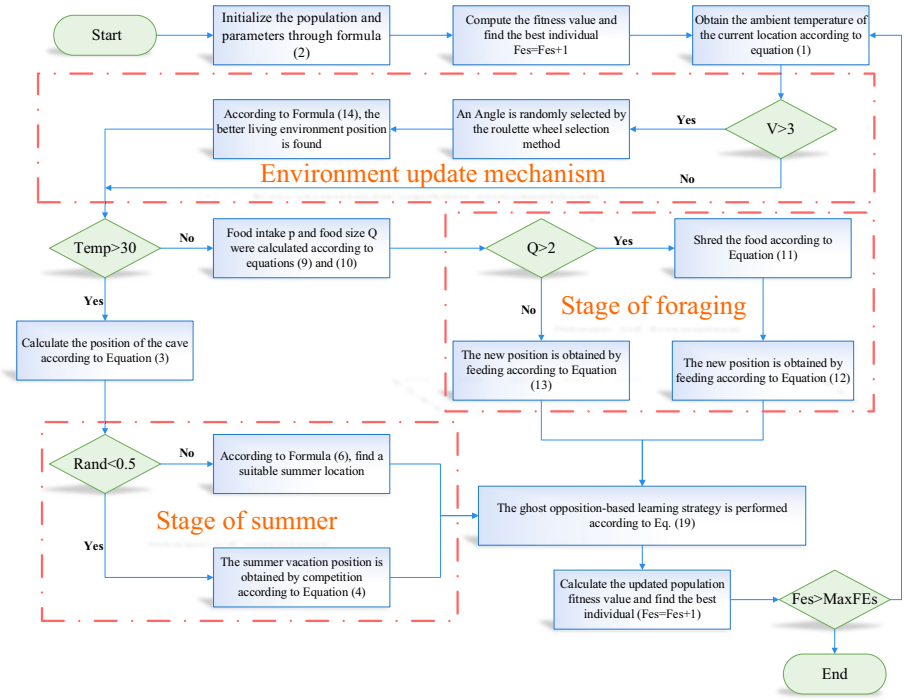


Fig. 4 Flow chart of the MCOA algorithm

The flow chart of the MCOA algorithm is as follows.

3.4 Computational complexity analysis

The complexity analysis of algorithms is an essential step to evaluate the performance of algorithms. In the experiment of complexity analysis of the algorithm, we choose the IEEE CEC2020 Special Session and Competition as the complexity evaluation standard of the single objective optimization algorithm. The complexity of MCOA algorithm mainly depends on several important parameters, such as the population size ($N=30$), the number of dimensions of the problem ($d=10$), the maximum number of evaluations of the algorithm ($MaxFES=100{,}000$) and the solution function (C). Firstly, the running time of the test program is calculated and the running time (T_0) of the test program is recorded, and the test program is shown in Algorithm 3. Secondly, under the same dimension of calculating the running time of the test program, the 10 test functions in the IEEE CEC2020 function set were evaluated 100,000 times, and their running time (T_1) was recorded. Finally, the running time of 100,000 evaluations of 10 test functions performed by MCOA for 5 times under the same dimension was recorded, and the average value was taken as the running time of the algorithm (T_2). Therefore, the formula for calculating the time complexity of MCOA algorithm is given in Eq. (21).

$$O(MCOA) = \frac{T2 - T1}{T0} \tag{21}$$

Table 2 Parameter Settings of the algorithm

Algorithm	Parameter setting	Value
MCOA	C_3	3
	μ	25
	σ	3
	c	2
COA	C_3	3
	μ	25
	σ	3
ROA	C	0.1
STOA	$b=1$	1
AOA	$a=5$	5
	μ	0.499
HHO	β	1.5
	u	(0,1)
	v	(0,1)
	$E0$	(− 1,1)
PDO	*rho*	0.005
	epsPD	0.1
GA	*Crossing-Over Rate*	0.7
	Mutation Rate	0.01
	α	1.5
MSCSO	*SM*	*2*
	Roulette Wheel selection	*[0,360]*
	β	*[0,2π]*
LSHADE	NP_{max}	$18\times d$
	NP_{min}	*4*
	p	*0.11*
	ms	*5*

Table 3 Specific function Settings of IEEE CEC2020 benchmark functions

No	Functions	Equation	F_{min}
1	Shifted and rotated Bent Cigar function	$F1 = x_1^2 + 10^6 \sum_{i=2}^{D} x_i^2$	100
		$F1(M(x - o_1)) + F1^*$	
2	Shifted and rotated Schwefel's function	$f(x) = 418.9829 \times D - \sum_{i=1}^{D} g(z_i)$	1100
		$z_i = x_i + 4.209687462275036e + 02$	
		$g(z_i) = \begin{cases} z_i sin\left(\lvert z_i \rvert^{\frac{1}{2}}\right), & if \lvert z_i \rvert \le 500 \\ 500 - mod(z_i, 500)\sin(\sqrt{\lvert 500 - mod(z_i, 500) \rvert}) - \frac{(z_i - 500)^2}{10000d}, & if\ z_i > 500 \\ (mod(\lvert z_i \rvert, 500) - 500)\sin(\sqrt{\lvert mod(\lvert z_i \rvert, 500) - 500 \rvert}) - \frac{(z_i + 500)^2}{10000d}, & if\ z_i < -500 \end{cases}$	
		$F2(x) = f\left(M\left(\frac{1000(x - o_2)}{100}\right)\right) + F2^*$	
3	Shifted and rotated Lunacek bi-Rastrigin function	$f(x) = min\left(\sum_{i=1}^{D}(\widehat{x_i} - \mu_0)^2, dD + s\sum_{i=1}^{D}(\widehat{x_i} - \mu_0)^2\right) + 10\left(D - \sum_{i=1}^{D}\cos(2\pi\widehat{z_i})\right)$	700
		$\mu_0 = 2.5, \mu_1 = \sqrt{\frac{\mu_0^2 - d}{s}}, s = 1 - \frac{1}{2\sqrt{D+20} - 8.2}, d = 1$	
		$y = \frac{10(x-o)}{100}, \frac{x_i}{x_i} = 2sign(x_i^*)y_i + \mu_0, fori = 1, 2, ..., D$	
		$z = \overset{100}{\wedge}(\hat{x} - \mu_0)$	
		$F3(x) = f\left(M\left(\frac{600(x-o_3)}{100}\right)\right) + F3^*$	
4	Expanded Rosenbrock's plus Griewangk's function	$f1(x) = \sum_{i=1}^{D-1}(100(x_i^2 - x_{i+1})^2 + (x_i - 1)^2)$	1900
		$f2(x) = \sum \frac{x_i^2}{4000} - \prod_{i=1}^{D}\cos\left(\frac{x_i}{\sqrt{i}}\right) + 1$	
		$f4 = f2(f1(x_1, x_2)) + f2(f1(x_2, x_3)) + \cdots + f2(f1(x_{D-1}, x_D)) + f2(f1(x_D, x_1)) + f4^*$	

Table 3 (continued)

No	Functions	Equation	F_{min}
5	Hybrid function 2 (N = 4)	$N = 3, p = [0.3, 0.3, 0.4]$ $g1$: *Modified Schvvefel′s Function*, $g2$: *Rastrigin′s Function* $g3$: *High Conditioned Elliptic Function f*3	1700
6	Hybrid function 2 (N = 4)	$N = 4, p = [0.2, 0.2, 0.3, 0.3]$ $g1$:*Expanded Schaffer Function*, $g2$: *HGBat Function* $g3$:*Rosenbrock′s Function*, $g4$:*Modified Schwefel′sFunction*	1600
7	Hybrid function 3 (N = 5)	$N = 5, p = [0.1, 0.2, 0.2, 0.2, 0.3]$ $g1$:*Expanded Schaffer Function*, $g2$:*HGBat Function* $g3$:*Rosenbrock′s Function*, $g4$:*Modified Schwefel′s Function* $g5$: *High Conditioned Elliptic Function*	2100
8	Composition function 1 (N = 3)	$N = 3, \sigma = [10, 20, 30], \lambda = [1, 10, 1], bias = [0, 100, 200]$ $g1$:*Rastrigin′s Function*, $g2$:*Griewank′s Function*, $g3$:*Modified Schwefel′s Function*	2200
9	Composition function 2(N = 4)	$V = 4, \sigma = [10, 20, 30, 40], \lambda = [10, 1e-6, 10, 1], bias = [0, 100, 200, 300]$ $g1$:*Ackley′s Function*, $g2$:*High Conditioned Elliptic Function* $g3$:*Griewank′s Function*, $g4$:*Rastrigin′s Function*	2400
10	Composition function 1 (N = 5)	$N = 5, \sigma = [10, 20, 30, 40, 50], \lambda = [10, 1, 10, 1e-6, 1], bias = [0, 100, 200, 300, 400]$ $g1$: *Rastrigin′s Function*, $g2$: *Happycat Function*, $g3$: *Ackley′s Function* $g4$: *Discus Function*, $g5$: *Rosenbrock′s Function*	2500

Search Range: [− 100,100]dim

Table 4 Experimental results of each algorithm in IEEE CEC2020 (optimal data are marked in bold)

Functions	Statistics	MCOA	COA	ROA	STOA	AOA	HHO	PDO	GA	MSCSO	LSHADE
F1	mean	2.660E+03	4.703E+03	3.759E+09	9.313E+07	3.397E+09	2.464E+05	5.653E+09	5.451E+09	3.478E+07	**100**
	std	2.763E+03	4.065E+03	2.198E+09	1.808E+08	2.387E+09	1.052E+05	2.075E+09	2.481E+09	1.122E+08	**0**
	rank	2	3	8	6	7	5	10	9	4	**1**
F2	mean	1.511E+03	1.522E+03	2.291E+03	1.829E+03	2.054E+03	1.978E+03	2.490E+03	2.872E+03	1.675E+03	**1.179E+03**
	std	2.039E+02	2.314E+02	2.899E+02	2.359E+02	2.603E+02	2.580E+02	2.097E+02	3.227E+02	2.127E+02	**8.838E+01**
	rank	2	3	8	5	7	6	9	10	4	**1**
F3	mean	7.362E+02	7.678E+02	8.151E+02	7.482E+02	7.940E+02	7.818E+02	8.130E+02	8.245E+02	7.586E+02	**7.149E+02**
	std	**6.247E+00**	3.204E+01	3.373E+01	1.187E+01	1.626E+01	2.148E+01	2.424E+01	1.708E+01	1.575E+01	6.399E+00
	rank	2	5	9	3	7	6	8	10	4	**1**
F4	mean	**1900**	1900	1900	1900	1900	1900	1900	1900	1900	1900.168687
	std	**0**	0	0	0	0	0	0	1.03585E-07	0	0.063107628
	rank	**1**	**1**	**1**	**1**	**1**	**1**	**1**	9	**1**	10
F5	mean	3.576E+03	3.831E+03	3.839E+05	3.249E+04	2.791E+04	1.804E+04	2.750E+05	4.364E+06	3.586E+04	**1.806E+03**
	std	2.351E+03	2.376E+03	2.911E+05	8.627E+04	1.685E+04	2.727E+04	2.089E+05	3.151E+06	1.156E+05	**1.054E+02**
	rank	3	2	9	6	7	4	8	10	5	**1**
F6	mean	1.630E+03	1.721E+03	1.881E+03	1.717E+03	1.933E+03	1.814E+03	1.947E+03	2.261E+03	1.721E+03	**1.610E+03**
	std	**3.509E+01**	1.227E+02	1.150E+02	6.197E+01	1.183E+02	1.150E+02	8.777E+01	2.136E+02	8.107E+01	4.016E+01
	rank	2	3	7	5	8	6	9	10	4	**1**
F7	mean	2.320E+03	2.327E+03	1.532E+05	9.141E+03	7.513E+03	1.203E+04	1.596E+05	4.646E+06	7.307E+03	**2.154E+03**
	std	**1.455E+02**	1.656E+02	7.703E+05	6.370E+03	3.120E+03	9.611E+03	1.685E+05	5.049E+06	4.641E+03	1.668E+02
	rank	2	3	7	6	5	8	9	10	4	**1**
F8	mean	**2.299E+03**	2.301E+03	2.597E+03	2.836E+03	2.647E+03	2.352E+03	2.766E+03	2.802E+03	2.300E+03	2.300E+03
	std	1.192E+01	8.238E-01	2.209E+02	5.754E+02	2.229E+02	2.044E+02	2.574E+02	2.447E+02	2.207E+01	**2.511E-01**
	rank	**1**	4	6	8	7	5	9	10	2	3
F9	mean	**2.566E+03**	2.746E+03	2.796E+03	2.746E+03	2.819E+03	2.809E+03	2.813E+03	2.852E+03	2.706E+03	2.720E+03
	std	**6.369E+00**	6.569E+00	6.808E+01	4.479E+01	8.394E+01	7.364E+01	3.474E+01	2.852E+01	1.144E+02	5.999E+01
	rank	**1**	4	6	5	9	7	8	10	2	3

Table 4 (continued)

Functions	Statistics	MCOA	COA	ROA	STOA	AOA	HHO	PDO	GA	MSCSO	LSHADE
F10	mean	**2.921E+03**	2.928E+03	3.112E+03	2.930E+03	3.072E+03	2.936E+03	3.209E+03	3.202E+03	2.925E+03	2.932E+03
	std	**1.869E+01**	2.934E+01	1.162E+02	2.077E+01	1.000E+02	2.020E+01	7.910E+01	9.954E+01	6.220E+01	2.090E+01
	rank	**1**	5	8	3	7	6	9	10	2	4
Friedman average rank		**1.7**	3.3	6.9	4.8	6.5	5.4	8	9.8	3.2	2.6
Friedman rank		**1**	4	8	5	7	6	9	10	3	2

best/worst data and are bold to facilitate clear observation by the reader

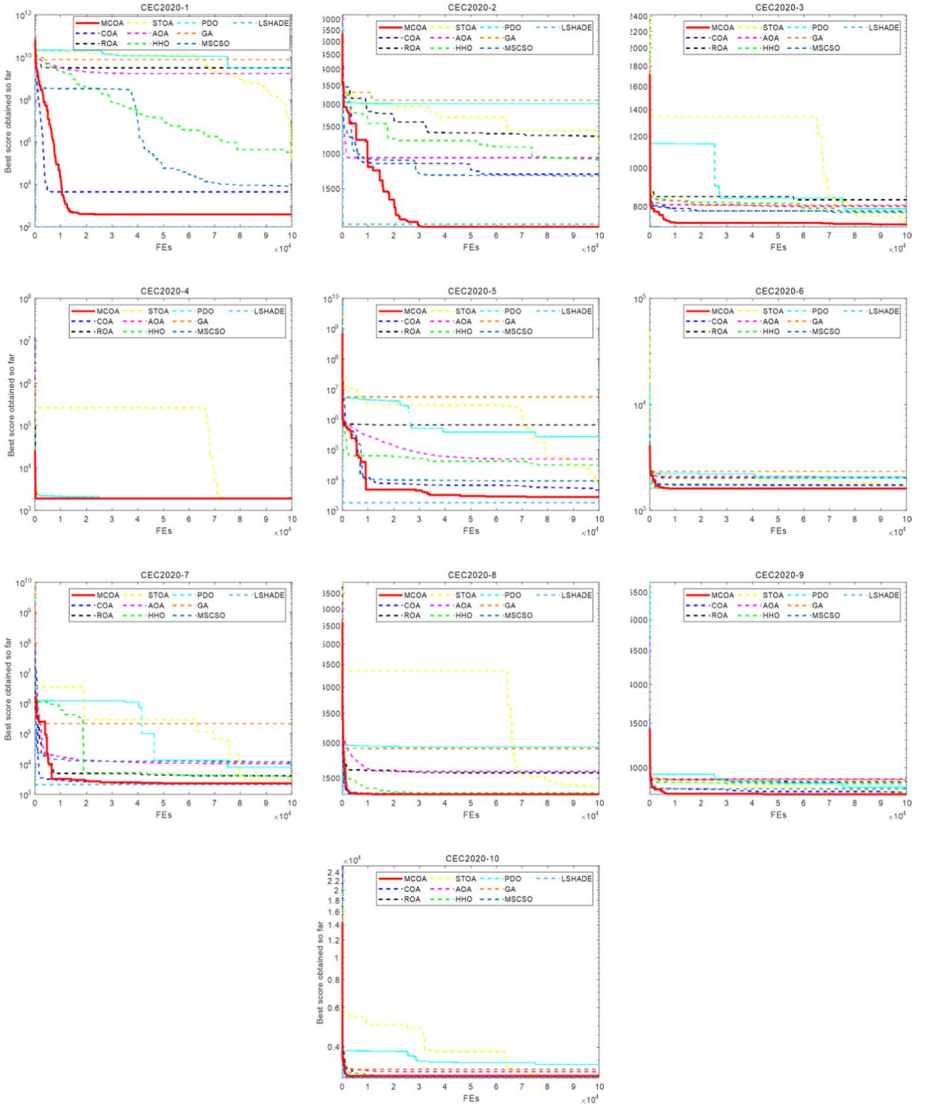


Fig. 5 Convergence curve of MCOA algorithm in IEEE CEC2020

Algorithm 3 IEEE CEC2020 complexity analysis test program

$x = 0.55$

For $i = 1 : 100000$

$x = x + x;$

$x = x / 2;$

$x = x \times x;$

$x = sqrt(x);$

$x = log(x);$

$x = exp(x);$

$x = x / (x + 2);$

End

Computing time for the above = T_0

The experimental data table of algorithm complexity analysis is shown in Table 1. In the complexity analysis of the algorithm, we use the method of comparing MCOA algorithm with other seven metaheuristic algorithms to illustrate the complexity of MCOA. In Table 1, we can see that the complexity of MCOA is much lower than other comparison algorithms such as ROA, STOA, and AOA. However, compared with COA, the complexity of MCOA is slightly higher than that of COA because it takes a certain amount of time to update the location through the environment update mechanism and ghost opposition-based learning strategy. Although the improved strategy of MCOA increases the computation time to a certain extent, the optimization performance of MOCA has been significantly improved through a variety of experiments in section four of this paper, which proves the good effect of the improved strategy.

4 Experimental results and discussion

The experiments are carried out on a 2.50 GHz 11th Gen Intel(R) Core(TM) i7-11,700 CPU with 16 GB memory and 64-bit Windows11 operating system using Matlab R2021a. In order to verify the performance of MCOA algorithm, MCOA is compared with nine metaheuristic algorithms in this subsection. In the experiments, we used the IEEE CEC2020 test function to evaluate the optimization performance of the MCOA algorithm Fig. 4.

Table 5 Wilcoxon rank-sum test results for IEEE CEC2020 test functions

CEC	COA vs MCOA	ROA vs MCOA	STOA vs MCOA	AOA vs MCOA	HHO vs MCOA	PDO vs MCOA	GA vs MCOA	MSCSO vs MCOA	LSHADE vs MCOA
F1	9.426E-01	1.734E-06	1.734E-06	1.734E-06	1.734E-06	1.734E-06	1.734E-06	1.779E-02	1.734E-06
F2	3.881E-04	2.879E-06	2.703E-02	3.405E-05	6.424E-03	2.127E-06	1.734E-06	4.779E-01	1.921E-06
F3	2.412E-04	1.921E-06	4.682E-03	1.734E-06	2.353E-06	1.734E-06	1.734E-06	3.405E-05	1.734E-06
F4	1	1	1	1	1	1	1.734E-06	1	1.734E-06
F5	2.116E-02	4.729E-06	1.251E-04	1.734E-06	4.114E-03	1.734E-06	1.734E-06	3.190E-02	1.734E-06
F6	9.590E-01	7.691E-06	4.196E-04	2.353E-06	6.892E-05	1.734E-06	1.734E-06	2.849E-02	1.025E-05
F7	1.057E-04	1.734E-06	2.127E-06	2.603E-06	5.752E-06	1.734E-06	1.734E-06	1.734E-06	1.734E-06
F8	4.492E-02	1.734E-06	1.734E-06	1.734E-06	2.843E-05	1.734E-06	1.734E-06	3.872E-02	2.843E-05
F9	2.353E-06	1.734E-06	3.882E-06	3.182E-06	4.729E-06	1.734E-06	2.879E-06	4.072E-05	1.742E-04
F10	4.165E-02	1.734E-06	1.108E-02	3.182E-06	5.193E-02	1.734E-06	1.734E-06	6.884E-01	8.451E-01
+/=/−	7/1/2	9/1/0	9/1/0	9/1/0	8/1/1	9/1/0	10/0/0	7/1/2	9/0/1

4.1 Experiments with IEEE CEC2020 test functions

In this subsection, using the Crayfish Optimization Algorithm (COA), Remora Optimization Algorithm (ROA) (Jia et al. 2021), Sooty Tern Optimization Algorithm (STOA) (Dhiman and Kaur 2019), Arithmetic Optimization Algorithm (AOA) (Abualigah et al. 2021), Harris Hawk Optimization Algorithm (HHO) (Heidari et al. 2019), Prairie Dog Optimization Algorithm (PDO) (Ezugwu et al. 2022), Genetic Algorithm (GA) (Mirjalili and Mirjalili 2019),Modified Sand Cat Swarm Optimization Algorithm (MSCSO) (Wu et al. 2022) and a competition algorithm LSHADE (Piotrowski 2018) were compared to verify the optimization effect of MCOA. The parameter Settings of each algorithm are shown in Table 2.

In order to test the performance of MCOA, this paper selects 10 benchmark test functions of IEEE CEC2020 for simulation experiments. Where F1 is a unimodal function, F2–F3 is a multimodal function, F4 is a non-peak function, F5–F7 is a hybrid function, and F8-F10 is a composite function. The parameters of this experiment are uniformly set as follows: the maximum number of evaluation *MaxFEs* is 100,000, the population size *N* is 30, and the dimension size *d* is 10. The MCOA algorithm and the other nine algorithms are run independently for 30 times, and the average fitness value, standard deviation of fitness value and Friedman ranking calculation of each algorithm are obtained. The specific function Settings of the IEEE CEC2020 benchmark functions are shown in Table 3.

Table 6 Comparison results of single strategies (worst data are marked in bold)

Functions	Statistics	GOBLCOA	EUCOA	COA	MCOA
F1	min	1.06E+02	2.59E+02	**3.17E+02**	1.00E+02
	mean	3.18E+03	3.53E+03	**4.38E+03**	2.54E+03
	std	3.27E+03	3.30E+03	**3.73E+03**	2.77E+03
F2	min	1.11E+03	1.11E+03	**1.16E+03**	1.12E+03
	mean	**1.70E+03**	1.59E+03	1.56E+03	1.64E+03
	std	2.77E+02	2.61E+02	2.13E+02	**2.90E+02**
F3	min	7.21E+02	7.17E+02	**7.25E+02**	7.16E+02
	mean	7.55E+02	7.44E+02	**7.75E+02**	7.42E+02
	std	2.63E+01	1.90E+01	**3.39E+01**	1.61E+01
F4	min	1900	1900	1900	1900
	mean	1900	1900	1900	1900
	std	0	0	0	0
F5	min	1.76E+03	2.03E+03	**2.05E+03**	1.76E+03
	mean	3.73E+03	3.71E+03	3.48E+03	**4.28E+03**
	std	2.02E+03	**2.12E+03**	1.45E+03	2.05E+03
F6	min	1.60E+03	1.60E+03	1.60E+03	1.60E+03
	mean	1.66E+03	**1.70E+03**	**1.70E+03**	1.66E+03
	std	7.68E+01	1.09E+02	**1.17E+02**	8.28E+01
F7	min	2.15E+03	2.19E+03	**2.22E+03**	2.21E+03
	mean	**2.53E+03**	2.40E+03	2.39E+03	2.51E+03
	std	2.39E+02	**4.98E+02**	1.63E+02	3.16E+02
F8	min	2.30E+03	2.30E+03	2.30E+03	2.30E+03
	mean	2.30E+03	2.30E+03	2.30E+03	2.30E+03
	std	5.36E-01	1.49E+00	**1.64E+00**	7.68E-01
F9	min	2.50E+03	2.50E+03	**2.73E+03**	2.50E+03
	mean	2.73E+03	2.56E+03	**2.75E+03**	2.54E+03
	std	6.27E+01	**8.96E+01**	7.90E+01	7.87E+01
F10	min	**2.90E+03**	2.60E+03	**2.90E+03**	2.89E+03
	mean	2.92E+03	2.92E+03	**2.93E+03**	2.91E+03
	std	2.42E+01	2.33E+01	**2.96E+01**	2.38E+01

best/worst data and are bold to facilitate clear observation by the reader

4.1.1 Results statistics and convergence curve analysis of IEEE CEC2020 benchmark functions

In order to more clearly and intuitively compare the ability of MCOA and various algorithms to find individual optimal solutions, the average fitness value, standard deviation of fitness value and Friedman ranking obtained by running MCOA and other comparison algorithms independently for 30 times are presented in the form of tables and images. The data and images are shown in Table 4 and Fig. 5 respectively.

In Table 4, mean represents the average fitness value, std represents the standard deviation of fitness value, rank represents the Friedman ranking, Friedman average rank

Table 7 Comparison of different water flow velocity factor of CEC2020 (optimal data are marked in bold)

Functions	Statistics	$c=2$	$c=3$	$c=4$	$c=5$	$c=6$	$c=7$
F1	mean	**2.66E+03**	3.07E+03	4.49E+03	2.70E+03	3.74E+03	3.78E+03
	std	2.76E+03	3.65E+03	2.64E+03	**2.60E+03**	3.95E+03	3.61E+03
F2	mean	**1.51E+03**	1.71E+03	1.68E+03	1.71E+03	1.67E+03	1.63E+03
	std	**2.49E+02**	2.81E+02	3.18E+02	2.86E+02	3.27E+02	2.72E+02
F3	mean	**7.35E+02**	7.44E+02	7.38E+02	7.36E+02	7.40E+02	7.40E+02
	std	**8.80E+00**	2.20E+01	1.49E+01	1.52E+01	1.70E+01	2.14E+01
F4	mean	**1900**	**1900**	**1900**	**1900**	**1900**	**1900**
	std	**0**	**0**	**0**	**0**	**0**	**0**
F5	mean	**3.65E+03**	5.13E+03	4.71E+03	4.91E+03	4.67E+03	4.41E+03
	std	2.08E+03	2.42E+03	2.12E+03	2.38E+03	**1.91E+03**	2.30E+03
F6	mean	**1.63E+03**	1.69E+03	1.67E+03	1.67E+03	1.67E+03	1.68E+03
	std	**4.63E+01**	9.61E+01	6.44E+01	6.34E+01	9.21E+01	7.70E+01
F7	mean	**2.59E+03**	2.82E+03	2.87E+03	2.72E+03	2.77E+03	2.82E+03
	std	**1.81E+02**	4.34E+02	4.98E+02	2.80E+02	3.44E+02	3.29E+02
F8	mean	**2.30E+03**	2.30E+03	2.30E+03	2.30E+03	2.30E+03	2.30E+03
	std	**2.12E+00**	6.90E+00	1.13E+01	2.20E+00	2.20E+00	9.07E+00
F9	mean	**2.53E+03**	2.60E+03	2.58E+03	2.60E+03	2.58E+03	2.62E+03
	std	**6.14E+01**	1.18E+02	1.13E+02	1.19E+02	1.13E+02	1.23E+02
F10	mean	**2.91E+03**	2.93E+03	2.93E+03	2.93E+03	2.93E+03	2.93E+03
	std	6.43E+01	2.96E+01	2.84E+01	**2.24E+01**	2.44E+01	2.28E+01

best/worst data and are bold to facilitate clear observation by the reader

represents the average ranking of the algorithm among all functions, and Friedman rank represents the final ranking of this algorithm. Compared with other algorithms, MCOA achieved the best results in average fitness value, standard deviation of fitness value and Friedman ranking. In unimodal function F1, although MCOA algorithm is slightly worse than LSHADE algorithm, MCOA is superior to other algorithms in mean fitness value, standard deviation of fitness value, Friedman ranking and other aspects. In the multimodal functions F2 and F3, although the average fitness value of MCOA is slightly worse, it also achieves a good result of ranking second. The standard deviation of fitness value in F3 is better than other comparison algorithms in terms of stability. In the peakless function F4, except GA and LSHADE algorithm, other algorithms can find the optimal individual solution stably. In the mixed functions F5, F6, and F7, although the mean fitness value of LSHADE is better than that of MCOA, the standard deviation of the fitness value of MCOA is better than that of the other algorithms compared. Among the composite functions of F8, F9 and F10, the standard deviation of MCOA's fitness value at F8 is slightly worse than that of LSHADE, but the average fitness value and standard deviation of fitness value are the best in other composite functions, and it has achieved the first place in all composite functions. Finally, from the perspective of Friedman average rank, MCOA has a strong comprehensive performance and still ranks first. Through the analysis of the data in Table 4, it can be seen that MCOA ranks first overall and has good optimization effect, and its optimization performance is better than other 9 comparison algorithms.

Figure 5 shows that in the IEEE CEC2020 benchmark functions, for the unimodal function F1, although LSHADE algorithm has a better optimization effect, compared with similar meta-heuristic algorithms, MCOA has a slower convergence rate in the early stage, but can be separated from local optimal and converge quickly in the middle stage. In the multimodal functions F2 and F3, similar to F1, MCOA converges faster in the middle and late stages, effectively exiting the local optimal. Although the convergence speed is slower than that of LSHADE, the optimal value can still be found. In the peak-free function F4, the optimal value can be found faster by all algorithms except LSHADE, STOA and PDO because the function is easy to implement. In the mixed functions F5, F6 and F7, although the convergence rate of MCOA is slightly slower than that of COA algorithm in the early stage, it can still find better values than the other eight algorithms except LSHADE in the later stage. For the composite functions F8, F9 and F10, MCOA can find the optimal value faster than the other nine algorithms.

Based on the above, although LSHADE has a stronger ability to find the optimal value in a small number of functions, MCOA can still find the optimal value in most functions in the later stage, and compared with the other eight pair algorithms of the same type, MCOA has more obvious enhancement in optimization ability and avoidance of local optimization, and has better application effect.

4.1.2 Analysis of Wilcoxon rank sum test results

In the comparison experiment, the different effects of multiple algorithms solving the same problem are used to judge whether each algorithm has high efficiency and more obvious influence on solving the current problem, such as the convergence speed of the convergence curve, the fitness value of the optimal solution, the ability to jump out of the local optimum, etc. At present, only the average fitness value, the standard deviation of fitness value and the convergence curve can not be used as the basis for judging whether the performance of the algorithm is efficient. Therefore, the data and images presented by each algorithm in solving the current problem are comprehensively analyzed, and the Wilcoxon rank sum test is used to further verify the difference between MCOA and the other nine comparison algorithms. In this experiment, the significance level is defined as 5%. If its calculated value is less than 5%, it proves that there is a significant difference between the two algorithms, and if it is greater than 5%, it proves that there is no significant difference between the two algorithms. Table 5 shows the Wilcoxon rank-sum test results of the MCOA algorithm and the other nine comparison algorithms. Where the symbols "+", "−"

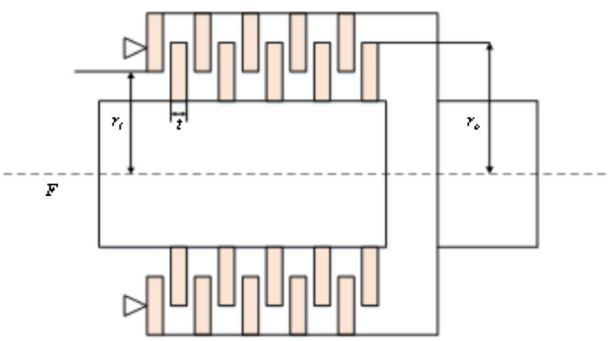


Fig. 6 Schematic diagram of the multi-disc clutch braking problem

and "=" table the performance of MCOA better, worse and equal to the comparison algorithms, respectively.

In the calculation of the function F4 without peak, the value of 1 appears in the comparison of various algorithms such as MCOA, COA, ROA, STOA and other algorithms, indicating that in this function, a variety of algorithms have found the optimal value, there is no obvious difference, which can be ignored. However, in most of the remaining functions, the significance level of MCOA compared with the other nine algorithms is less than 5%, which is a significant difference.

From the overall table, the MCOA algorithm also achieves good results in the Wilcoxon rank-sum test of the IEEE CEC2020 benchmark function, and the contrast ratio with other algorithms is less than 5%, which proves that the MCOA algorithm has a significant difference from the other nine algorithms, and MCOA has better optimization performance. According to the comparison results with the original algorithm, it is proved that MCOA algorithm has a good improvement effect.

4.2 Comparison experiment of single strategy

MCOA adopts two strategies, environment update mechanism and ghost opposition-based learning strategy, to improve COA. In order to prove the effectiveness of these two strategies for algorithm performance optimization, a single strategy comparison experiment is added in this section. In the experiment in this section, EUCOA algorithm which only adds environment update mechanism and GOBLCOA algorithm which only adds ghost opposition-based learning strategy are compared with the basic COA algorithm. The experiments are independently run 30 times in IEEE CEC2020 benchmark test function, and the statistical data obtained are shown in Table 6. In order to make the table easy to view the statistical results, the poor data in the table will be bolded to make the statistical results more clear and intuitive. It can be seen from the table that among the best fitness values, average fitness values and standard deviation of fitness values of the 10 test functions, GOBLCOA and EUCOA account for less bolded data, while most data of the original algorithm COA are bolded in the table, which effectively proves that both the environment update mechanism and the ghost opposition-based learning strategy play a certain role in COA. The comprehensive performance of COA has been significantly improved.

4.3 Parameter sensitivity analysis of water flow velocity factor *c*

In order to better prove the influence of flow velocity coefficient on MCOA, we choose different flow velocity coefficient c values for comparison experiments. Table 7 shows the statistical results of 30 independent runs of different water flow velocity coefficients in CEC2020. The bold sections represent the best results. As can be seen from the table, the result obtained by $c=2$ is significantly better than the other values. Only in individual test functions are the results slightly worse. In the F1 function, $c=5$ has the best std. In the F5 function, std is best at $c=6$. Among F10 functions, $c=5$ has the best std. Among the other test functions, both the mean fitness value and std at water flow velocity factor $c=2$ are optimal. Through the above analysis, it is proved that the water flow velocity factor $c=2$ has a good optimization effect.

Table 8 Experimental results of the multi-disc clutch braking problem

Algorithm	Optimal values for variables					Constraint values	Optimal Weight
	×1	×2	×3	×4	×5		
MCOA	70	90	1	600	2	g1 = 1.1712e-08,g2 = 25.5, g3 = 0.9602,g4 = 9.9999, g5 = 9.9984,g6 = 14.9702, g7 = 4.8190e + 04,g8 = 0.0298	0.235242
COA(Jia et al. 2023b)	67.194	90	1.004733	601.328	2	g1 = 2.8059,g2 = 25.4858, g3 = 0.9644,g4 = 9.9999, g5 = 9.9984,g6 = 14.9697, g7 = 4.7534e + 04,g8 = 0.0303	0.264789
CMVO(Sayed et al. 2018)	70	90	1	910	3	g1 = 0,g2 = 24,g3 = 0.9397,g4 = 9.9999, g5 = 9.9984,g6 = 14.9869, g7 = 1.0971e + 05,g8 = 0.0131	0.313656
MVO(Mirjalili et al. 2016)	70	90	1	910	3	g1 = 0,g2 = 24,g3 = 0.9397,g4 = 9.9999, g5 = 9.9984,g6 = 14.9869, g7 = 1.0971e + 05,g8 = 0.0131	0.313656
SHO(Zhao et al. 2023)	69.685	90	1	600	2	g1 = 0.3143,g2 = 25.5, g3 = 0.9607,g4 = 9.9999, g5 = 9.9984,g6 = 14.9701, g7 = 4.8104e + 04,g8 = 0.0299	0.238470
MFO(Mirjalili 2015)	70	90	1	910	3	g1 = 0,g2 = 24,g3 = 0.9397,g4 = 9.9999, g5 = 9.9984,g6 = 14.9869, g7 = 1.0971e + 05,g8 = 0.0131	0.313656
MPA(Faramarzi et al. 2020)	69.998	90	1	998.958	2	g1 = 0.0016,g2 = 25.5, g3 = 0.9338,g4 = 9.9999, g5 = 9.9984,g6 = 14.9821, g7 = 8.0272e + 04,g8 = 0.0179	0.235258
SCA(Mirjalili 2016)	69.984	90	1	1000	2	g1 = 0.0159,g2 = 25.5, g3 = 0.9337,g4 = 9.9999, g5 = 9.9984,g6 = 14.9821, g7 = 8.0349e + 04,g8 = 0.0179	0.235406

Table 8 (continued)

Algorithm	Optimal values for variables					Constraint values	Optimal Weight
	×1	×2	×3	×4	×5		
SFO(Shadravan et al. 2019)	69.827	90.07	1.061	906.694	2.03	g1 = 0.2520,g2 = 25.2633, g3 = 0.9406,g4 = 9.9999, g5 = 9.9984,g6 = 14.9806, g7 = 7.4063e + 04,g8 = 0.0194	0.255509
WCA(Eskandar et al. 2012)	70	90	1	910	3	g1 = 0,g2 = 24,g3 = 0.9397,g4 = 9.9999, g5 = 9.9984,g6 = 14.9869, g7 = 1.0971e + 05,g8 = 0.0131	0.313656

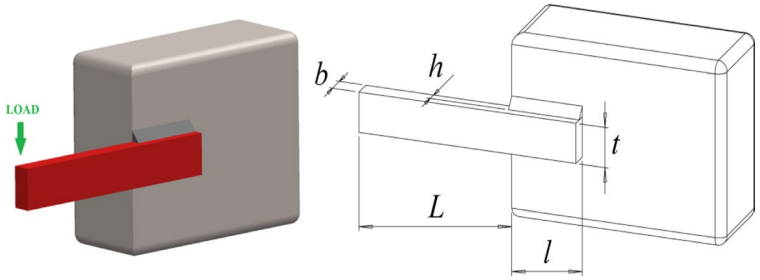


Fig. 7 Schematic diagram of the welded beam design problem

4.4 Experimental summary

In this section, we first test MCOA's optimization performance on the IEEE CEC2020 benchmark function. The improved MCOA is compared with the original algorithm COA and six other meta-heuristic algorithms in the same environment and the experimental analysis is carried out. Secondly, the rank sum test is used to verify whether there are significant differences between MCOA and the other nine comparison algorithms. Finally, three algorithms, EUCOA with environment update mechanism, GOBLCOA with ghost opposition-based learning strategy, COA and MCOA, are tested to improve performance. These three experimental results show that MCOA has a good ability to find optimal solutions and get rid of local optimal solutions.

5 Constrained engineering design problems

With the new development of the era of big data, the solution process becomes complicated and the calculation results become accurate, and more and more people pay close attention to the dynamic development of the feasibility and practicality of the algorithm, so as to ensure that the algorithm has good practical performance on constrained engineering design problems. In order to verify the optimization effect of MCOA in practical applications, four constrained engineering design problems are selected for application testing of MCOA to evaluate the performance of MCOA in solving practical application problems. Every constrained engineering design problems has a minimization objective function (Papaioannou and Koulocheris 2018) that is used to calculate the fitness value for a given problem. In addition, each problem contains a varying number of constraints that are taken into account during the calculation of the objective function. If the constraints are not met, the penalty function (Yeniay 2005) is used to adjust the fitness value. However, the processing of constraints is not the focus of our research, our focus is on the optimization of parameters in a convex region composed of constraints (Liu and Lu 2014). In order to ensure the fairness of the experiment, the parameters of all experiments in this section are set as follows: the maximum evaluation time *MaxFEs* is 10,000 and the overall scale N is 30. In each experiment, all the algorithms were analyzed 500 times and the optimal results were obtained.

5.1 Multi-disc clutch braking problem

In the field of vehicle engineering, there is a common constrained engineering design problems multi-disc clutch braking problem, and the purpose of our algorithm is to minimize the mass of the multi-disc clutch by optimizing eight constraints and five variables, so as to improve the performance of the multi-disc clutch. Among them, the five variables are: inner diameter r_i, outer diameter r_o, brake disc thickness t, driving force F, and surface friction coefficient Z. The specific structure of the multi-disc clutch is shown in Fig. 6.

The mathematical formulation of the multi-disc clutch braking problem is as follows.

Consider:

$$x = \begin{bmatrix} x_1 & x_2 & x_3 & x_4 & x_5 \end{bmatrix} = \begin{bmatrix} r_i & r_o & t & F & Z \end{bmatrix} \tag{22}$$

Objective function:

$$f(x) = \Pi\left(r_o^2 - r_i^2\right) t(Z+1)\rho(\rho = 0.0000078) \tag{23}$$

Subject to:

$$g_1(x) = r_o - r_i - \Delta r \geq 0 \tag{24}$$

$$g_2(x) = l_{max} - (Z+1)(t+\delta) \geq 0 \tag{25}$$

$$g_3(x) = P_{max} - P_{rz} \geq 0 \tag{26}$$

$$g_4(x) = P_{max} v_{sr\ \ max} - P_{rz} v_{sr} \geq 0 \tag{27}$$

$$g_5(x) = v_{sr\ \ max} - v_{sr} \geq 0 \tag{28}$$

$$g_6(x) = T_{max} - T \geq 0 \tag{29}$$

$$g_7(x) = M_h - sM_s \geq 0 \tag{30}$$

$$g_8(x) = T \geq 0 \tag{31}$$

Variable range:

$$\begin{aligned} &60 \leq x_1 \leq 80, 90 \leq x_2 \leq 110, 1 \leq x_3 \leq 3, \\ &600 \leq x_4 \leq 1000, 2 \leq x_5 \leq 9 \end{aligned} \tag{32}$$

Other parameters:

$$M_h = \frac{2}{3}\mu F Z \frac{r_o^3 - r_i^2}{r_o^2 - r_i^3}, P_{rz} = \frac{F}{\Pi\left(r_o^2 - r_i^2\right)}, \tag{33}$$

$$v_{rz} = \frac{2\Pi\left(r_o^3 - r_i^3\right)}{90\left(r_o^2 - r_i^2\right)}, T = \frac{I_z \Pi\ \ n}{30\left(M_h + M_f\right)} \tag{34}$$

Table 9 Experimental results of the welded beam design problem

Algorithm	h	l	t	b	Constraint values	Best weight
MCOA	0.20303	3.31003	9.084002	0.20578	g1 = − 76.7933-,g2 = − 735.0784, g3 = − 0.2358,g4 = − 0.0076, g5 = − 740.8458,g6 = − 0.0816, g7 = − 3.3562	1.70752
COA(Jia et al. 2023b)	0.21349	3.17064	8.85936	0.21406	g1 = − 135.4337,g2 = − 814.5862, g3 = − 0.2358,g4 = − 0.0039, g5 = − 878.3924,g6 = − 0.0869, g7 = − 3.3524	1.72629
TSA(Kiran 2015)	0.24415	6.22306	8.29555	0.2444	g1 = − 7.0335e + 03,g2 = − 33.2970, g3 = − 0.2343,g4 = − 2.5000e-04, g5 = − 3.4937e + 03,g6 = − 0.1192 g7 = − 2.9616	2.38241
CPSO(He and Wang 2007)	0.20236	3.54421	9.04821	0.20572	g1 = − 811.4128,g2 = − 75.8141, g3 = − 0.2356,g4 = − 0.0034, g5 = − 4.4729,g6 = − 0.0774 g7 = − 3.3836	1.73148
GSA(Rashedi and Nezamabadi-Pour 2009)	0.18212	3.85697	10	0.20237	g1 = − 1.2570e + 03,g2 = − 5.0959e + 03 g3 = − 0.2392,g4 = − 0.0202, g5 = − 89.4083,g6 = − 0.0571 g7 = − 3.2247	1.87995
RO(Kaveh and Khayatazad 2012)	0.20368	3.52846	9.00423	0.20724	g1 = − 801.9163,g2 = − 3.9690, g3 = − 0.2355,g4 = − 0.0036, g5 = − 118.6376,g6 = − 0.0787 g7 = − 3.3806	1.73534
IHS(Mahdavi et al. 2007)	0.20572	3.39879	9.03279	0.20598	g1 = − 522.6596,g2 = − 11.0227, g3 = − 0.2355,g4 = − 2.6000e-04, g5 = − 20.2515,g6 = − 0.0807 g7 = − 3.3958	1.72132
MGTOA(Rao et al. 2022)	0.20170	3.32972	9.06967	0.20572	g1 = − 522.6596,g2 = − 11.0227, g3 = − 0.2355,g4 = − 2.6000e-04, g5 = − 20.2515,g6 = − 0.0807 g7 = − 3.3958	1.70521

Table 9 (continued)

Algorithm	h	l	t	b	Constraint values	Best weight
SHO(Zhao et al. 2023)	0.23443	3.45973	7.38249	0.36814	g1 = − 45.9608,g2 = − 216.8211, g3 = − 0.2357,g4 = − 0.0040 g5 = − 13.5526,g6 = − 0.0767 g7 = − 3.3995	2.49298
CBO(Kaveh and Mahdavi 2014)	0.20572	3.47041	9.03727	0.20573	g1 = − 770.9521,g2 = − 5.1107, g3 = − 0.2355,g4 = − 1.5000e-05, g5 = − 0.7537,g6 = − 0.0807 g7 = − 3.3905	1.72466
BSAISA(Wang et al. 2018b)	0.20573	3.47048	9.03662	0.20573	g1 = 0,g2 = 0, g3 = − 5.55e-17,g4 = − 3.4330 g5 = − 0.0807,g6 = − 0.2355 g7 = − 5.46e-12	1.72485
IAPSO(Guedria 2016)	0.20572	3.47048	9.03662	0.2057	g1 = − 1.05e-10,g2 = − 6.91e-10, g3 = − 7.66e-15,g4 = − 3.4330 g5 = − 0.0807,g6 = − 0.2355 g7 = − 5.80e-10	1.72485

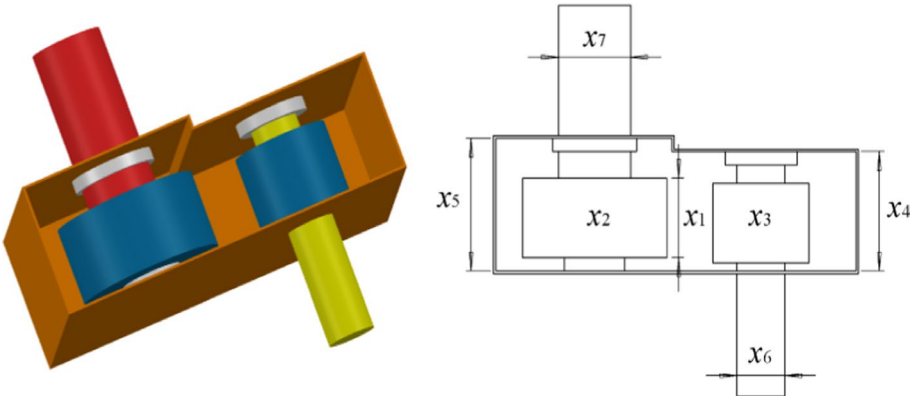


Fig. 8 Schematic diagram of the reducer design problem

$$\Delta r = 20\,\text{mm}, I_z = 55\text{kg mm}^2, P_{\max} = 1\text{MPa}, F_{\max} = 1000\,\text{N}, \tag{35}$$

$$T_{\max} = 15s, \mu = 0.5, s = 1.5, M_s = 40\,\text{Nm}, M_f = 3\,\text{Nm}, \tag{36}$$

$$n = 250\,\text{rpm}, v_{sr\ \max} = 10m/s, l_{\max} = 30\,\text{mm} \tag{37}$$

After calculation and experiments, the experimental results of the multi-disc clutch braking problem are made into a table as shown in Table 8. In Table 8, MCOA concluded that the inner diameter $r_i = 70$, the outer diameter $r_0 = 90$, the thickness of the brake disc $t = 1$, the driving force $F = 600$, and the surface friction coefficient $Z = 2$. At this time, the minimum weight obtained is 0.2352424, it is 11.16% higher than the original algorithm. Compared with MCOA, the other five algorithms in the calculation of this problem show that the optimization effect is far lower than that of MCOA.

5.2 Design problem of welding beam

The welded beam design problem is very common in the field of structural engineering and is constrained not only by four decision variables (welding width h, connecting beam length l, beam height t, and connecting beam thickness b) but also by seven other different conditions. Therefore, it is challenging to solve this problem. The purpose of the optimization algorithm is to achieve the best structural performance of the welded beam and reduce its weight by optimizing the small problems such as the shape, size and layout of the weld under many constraints. The specific structure of the welded beam is shown in Fig. 7.

The mathematical formulation of the welded beam design problem is as follows.

Consider:

$$x = [x_1 x_2 x_3 x_4] = [h\, l\, t\, b] \tag{38}$$

Objective function:

$$f(x) = 1.10471x_1^2x_2 + 0.04811x_3x_4(14.0 + x_2) \tag{39}$$

Subject to:

$$g_1(\vec{x}) = \tau(\vec{x}) - \tau_{\max} \le 0 \tag{40}$$

$$g_2(\vec{x}) = \sigma(\vec{x}) - \sigma_{\max} \le 0 \tag{41}$$

$$g_3(\vec{x}) = \delta(\vec{x}) - \delta_{\max} \le 0 \tag{42}$$

$$g_4(\vec{x}) = x_1 - x_4 \le 0 \tag{43}$$

$$g_5(\vec{x}) = P - P_c(\vec{x}) \le 0 \tag{44}$$

$$g_6(\vec{x}) = 0.125 - x_1 \le 0 \tag{45}$$

$$g_7(\vec{x}) = 1.10471x_1^2 + 0.04811x_3x_4(14.0 + x_2) - 0.5 \le 0 \tag{46}$$

where:

$$\tau(\vec{x}) = \sqrt{(\tau')^2 + 2\tau'\tau''\frac{x_2}{2R} + (\tau'')}, \tau' = \frac{P}{\sqrt{2x_1x_2}}, \tau'' = \frac{MR}{J}, \tag{47}$$

$$M = P\left(L + \frac{x_2}{2}\right), R = \sqrt{\frac{x_2^2}{4} + \left(\frac{x_1 + x_3}{2}\right)^2}, \sigma(\vec{x}) = \frac{6PL}{x_4x_3^2}, \tag{48}$$

$$J = 2\left\{\sqrt{2x_1x_2}\left[\frac{x_x^2}{4} + \left(\frac{x_1 + x_3}{2}\right)^2\right]\right\}, \delta(\vec{x}) = \frac{6PL^3}{Ex_4x_3^2}, \tag{49}$$

Table 10 Experimental results of the reducer design problem

Algorithm	Optimal values for variables							Optimal weight
	×1	×2	×3	×4	×5	×6	×7	
MCOA	3.47635	0.7	17	7.3	7.8	3.34862	5.2768	2988.2713592
COA(Jia et al. 2023b)	3.48159	0.7	17.0313	7.3135	7.8	3.34991	5.27457	2990.67332
CSO(Ahmed et al. 2020)	3.52246	0.7	17.0056	7.3	7.8680	3.35026	5.29422	3010.9478
CS(Yang and Deb 2014)	3.5015	0.7	17.2274	7.605	7.8181	3.352	5.2875	3000.981
FA(Baykasoglu and Ozsoydan 2015)	3.50749	0.7	17	7.7196	8.0808	3.35151	5.28705	3010.1374
GTOA(Zhang and Jin 2020)	3.50237	0.7	17	7.3	7.8000	3.35192	5.28550	2997.29701
TLBO(Rao and Rao 2016)	3.50068	0.7	17	7.8666	7.8000	3.36316	5.28897	3006.4205

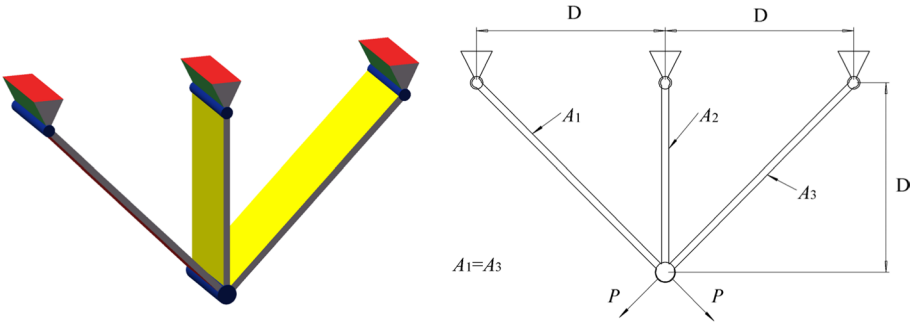


Fig. 9 Schematic diagram of the three-bar truss design problem

$$P_c(\vec{x}) = \frac{\sqrt{\frac{x_3^2 x_4^6}{0}}^{4.013E}}{L^2}, \left(1 - \frac{x_3}{2L}\sqrt{\frac{E}{4G}}\right), \left(1 - \frac{x_3}{2L}\sqrt{\frac{E}{4G}}\right), \tag{50}$$

$$P = 6000\,lb, L = 14\ \ in, \delta_{\max} = 0.25\ \text{in}, E = 30 \times 10^6, psi, \tag{51}$$

$$\tau_{\max} = 13600\quad \text{psi, and}\quad \sigma_{\max} = 30000\quad psi \tag{52}$$

Boundaries:

$$0.1 \le x_i \le 2, i = 1, 4; 0.1 \le x_i \le 10, i = 2.3 \tag{53}$$

The experimental results of the welding beam design problem are shown in Table 9. In the table, the welding width obtained by the MCOA algorithm h=0.203034,the length of the connecting beam is l=3.310032, the height of the beam is t=9.084002, and the thickness of the connecting beam is b=0.20578751. At this time, the minimum weight is 1.707524, it is 1.46% higher than the original algorithm. In the welding beam design problem, the weight determines the application effect of the algorithm in the practical problem. The weight of MCOA algorithm is smaller than that of other algorithms. Therefore, the practical application effect of MCOA is much greater than that of other algorithms.

5.3 Design problem of reducer

A reducer is a mechanical device used to reduce the speed of rotation and increase the torque. Among them, gears and bearings are an indispensable part of the reducer design, which have a great impact on the transmission efficiency, running stability and service life of the reducer. The weight of the reducer also determines the use of the reducer. Therefore, we will adjust the number of teeth, shape, radius and other parameters of the gear in the reducer to maximize the role of the reducer, reduce the friction between the parts, and extend the service life of the reducer. In this problem, a total of seven variables are constrained, which are the gear width x_1, the gear modulus x_2, the gear teeth x_3, the length of the first axis between bearings x_4, the

Table 11 Experimental results of the three-bar truss design problem

Algorithm	×1	×2	Constraint values	Optimal weight
MCOA	0.7887564807	0.4079947953	g1 = − 0.0052,g2 = − 1.4690, g3 = − 0.5363,	263.85438633
COA(Jia et al. 2023b)	0.7618303638	0.4900334526	g1 = − 1.2111e-05,g2 = − 1.3747, g3 = − 0.6253,	264.48151181
MBA(Sadollah et al. 2013)	0.788603	0.408453	g1 = − 4.8780e-07,g2 = − 1.4639, g3 = − 0.5361,	263.8959
GOA(Saremi et al. 2017)	0.788898	0.40762	g1 = − 1.2922e-06,g2 = − 1.4648, g3 = − 0.5352,	263.8959
SSA(Mirjalili et al. 2017)	0.78867	0.40828	g1 = − 1.3025e-05,g2 = − 1.4641, g3 = − 0.5359,	263.85984
EROA(Wang et al. 2022)	0.78645	0.41369	g1 = − 1.6747e-04,g2 = − 1.4576, g3 = − 0.5417,	263.85520

length of the second axis between bearings x_5, the diameter of the first axis x_6 and the diameter of the second axis x_7. The specific structure of the reducer is shown in Fig. 8.

The mathematical model of the reducer design problem is as follows.

Consider:

$$x = [x_1\, x_2\, x_3\, x_4\, x_5\, x_6\, x_7] \tag{54}$$

Objective function:

$$\begin{aligned} f(\vec{x}) = {} & 07854 \times x_1 \times x_2^2 \times (3.3333 \times x_3^2 + 14.9334 \times x_3 - \\ & 43.0934) - 1.508 \times x_1 \times (x_6^2 + x_7^2) + 7.4777 \times x_6^3 + x_7^3 + \\ & 0.7854 \times x_4 \times x_6^2 + x_5 \times x_7^2 \end{aligned} \tag{55}$$

Subject to:

$$g_1(\vec{x}) = \frac{27}{x_1 \times x_2^2 \times x_3} - 1 \le 0 \tag{56}$$

$$g_2(\vec{x}) = \frac{397.5}{x_1 \times x_2^2 \times x_3^2} - 1 \le 0 \tag{57}$$

$$g_3(\vec{x}) = \frac{1.93 \times x_4^3}{x_2 \times x_3 \times x_6^4} - 1 \le 0 \tag{58}$$

$$g_4(\vec{x}) = \frac{1.93 \times x_5^3}{x_2 \times x_3 \times x_7^4} - 1 \le 0 \tag{59}$$

$$g_5(\vec{x}) = \frac{1}{110 \times x_6^3} \times \sqrt{(\frac{745 \times x_4}{x_2 \times x_3})^2 + 16.9 \times 10^6} - 1 \leq 0 \tag{60}$$

$$g_6(\vec{x}) = \frac{1}{85 \times x_7^3} \times \sqrt{(\frac{745 \times x_5}{x_2 \times x_3})^2 + 16.9 \times 10^6} - 1 \leq 0 \tag{61}$$

$$g_7(\vec{x}) = \frac{x_2 \times x_3}{40} - 1 \leq 0 \tag{62}$$

$$g_8(\vec{x}) = \frac{5 \times x_2}{x_1} - 1 \leq 0 \tag{63}$$

$$g_9(\vec{x}) = \frac{x_1}{12 \times x_2} - 1 \leq 0 \tag{64}$$

$$g_{10}(\vec{x}) = \frac{1.5 \times x_6 + 1.9}{x_4} - 1 \leq 0 \tag{65}$$

$$g_{11}(\vec{x}) = \frac{1.1 \times x_7 + 1.9}{x_5} - 1 \leq 0 \tag{66}$$

Boundaries:

$$\begin{aligned} &2.6 \leq x_1 \leq 3.6, 0.7 \leq x_2 \leq 0.8, 17 \leq x_3 \leq 28, 7.3 \leq x_4 \leq 8.3, \\ &7.3 \leq x_5 \leq 8.3, 2.9 \leq x_6 \leq 3.9, 5 \leq x_7 \leq 5.5 \end{aligned} \tag{68}$$

The experimental results of the reducer design problem are shown in Table 10. From Table 10, it is known that the gear width calculated by the MCOA algorithm is $x_1 = 3.47635$, the gear modulus $x_2 = 0.7$, the gear teeth $x_3 = 17$, the length of the first axis between the bearings $x_4 = 7.3$, the length of the second axis between the bearings $\times 5 = 7.8$, and the length of the first axis between the bearings $x_5 = 7.8$. The diameter of the first axis is $x_6 = 3.348620$, the diameter of the second axis is $x_7 = 5.2768$, and the minimum weight is 2988.27135, it is 0.08% higher than the original algorithm. In this experiment, it can be concluded that MCOA has the smallest data among the minimum weights obtained by MCOA and other comparison algorithms in this problem, which proves that MCOA has the best optimization effect in solving such problems.

5.4 Design problem of three-bar truss

Three-bar truss structure is widely used in bridge, building, and mechanical equipment and other fields. However, the size, shape and connection mode of the rod need to be further explored by human beings. Therefore, $A_1 = x_1$ and $A_2 = x_2$ determined by the pairwise property of the system need to be considered in solving this problem. In addition to this, there will be constraints on the total support load, material cost, and other conditions such as cross-sectional area. The structural diagram of the three-bar truss is shown in Fig. 9.

The mathematical formulation of the three-bar truss design problem is as follows. Consider:

$$\vec{x} = [x_1 x_2] = [A_1 A_2] \tag{69}$$

Minimize:

$$f(\vec{x}) = (2\sqrt{2}x_1 + x_2) * l \tag{70}$$

Subject to:

$$g_1(\vec{x}) = \frac{\sqrt{2}x_1 + x_2}{\sqrt{2}x_1^2 + 2x_1 x_2} P - \sigma \leq 0 \tag{71}$$

$$g_2(\vec{x}) = \frac{x_2}{\sqrt{2}x_1^2 + 2x_1 x_2} P - \sigma \leq 0 \tag{72}$$

$$g_3(\vec{x}) = \frac{1}{\sqrt{2}x_1 + x_1} P - \sigma \leq 0 \tag{73}$$

$$l = 100\text{cm}, P = 2\text{kN/cm}^3, \sigma = 2\text{kN/cm}^3 \tag{74}$$

Variable range:

$$0 \leq x_1, x_2 \leq 1 \tag{75}$$

The experimental results of the three-bar truss design problem are shown in Table 11, from which it can be concluded that x_1=0.7887564and x_2=0.4079948of the MCOA algorithm on the three-bar truss design problem. At this time, the minimum weight value is 263.85438633, it is 0.24% higher than the original algorithm. Compared with the minimum weight value of other algorithms, the value of MCOA is the smallest. It is concluded that the MCOA algorithm has a good optimization effect on the three-bar truss design problem.

Table 12 Details of the datasets used

No	Dataset	Instances	Featrues	Categorical
1	Ionosphere	351	34	Biological
2	CLL-SUB-111	111	11,340	Biological
3	TOX-171	171	5748	Biological
4	Prostate-GE	102	5966	Biological
5	Nci9	60	9712	Biological
6	Colon	62	2000	Biological
7	GLI-85	85	22,283	Biological
8	Orlraws10P	100	10,304	Face image
9	Pixraw10P	100	10,000	Face image
10	Yale	165	1024	Face image
11	WarpAR10P	130	2400	Face image
12	WarpPIE10P	210	2420	Face image

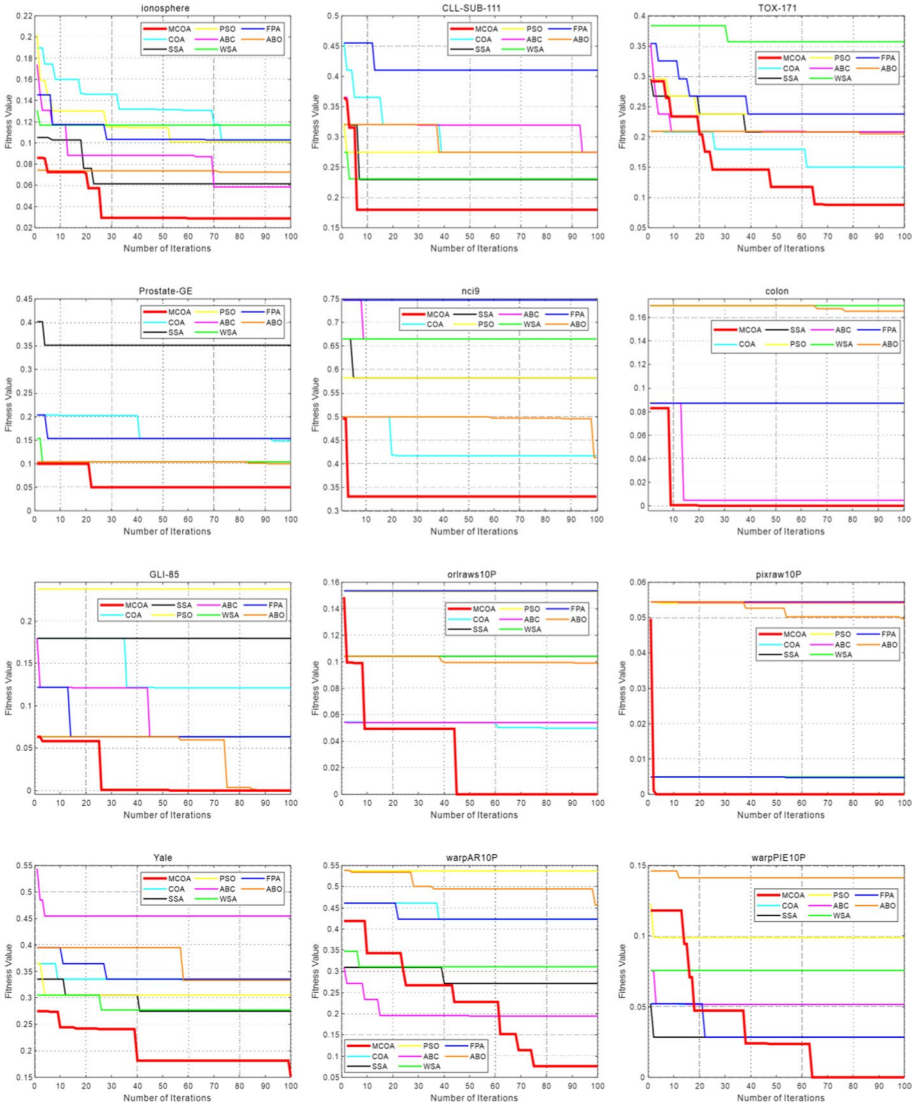


Fig. 10 Convergence curve of FS

The experimental results of four constrained engineering design problems show that MCOA has good optimization performance in dealing with problems similar to constrained engineering design problems. In addition, we will also introduce the high-dimensional feature selection problem of the wrapper method, and further judge whether MCOA has good optimization performance and the ability to deal with diversified problems through the classification and processing effect of data.

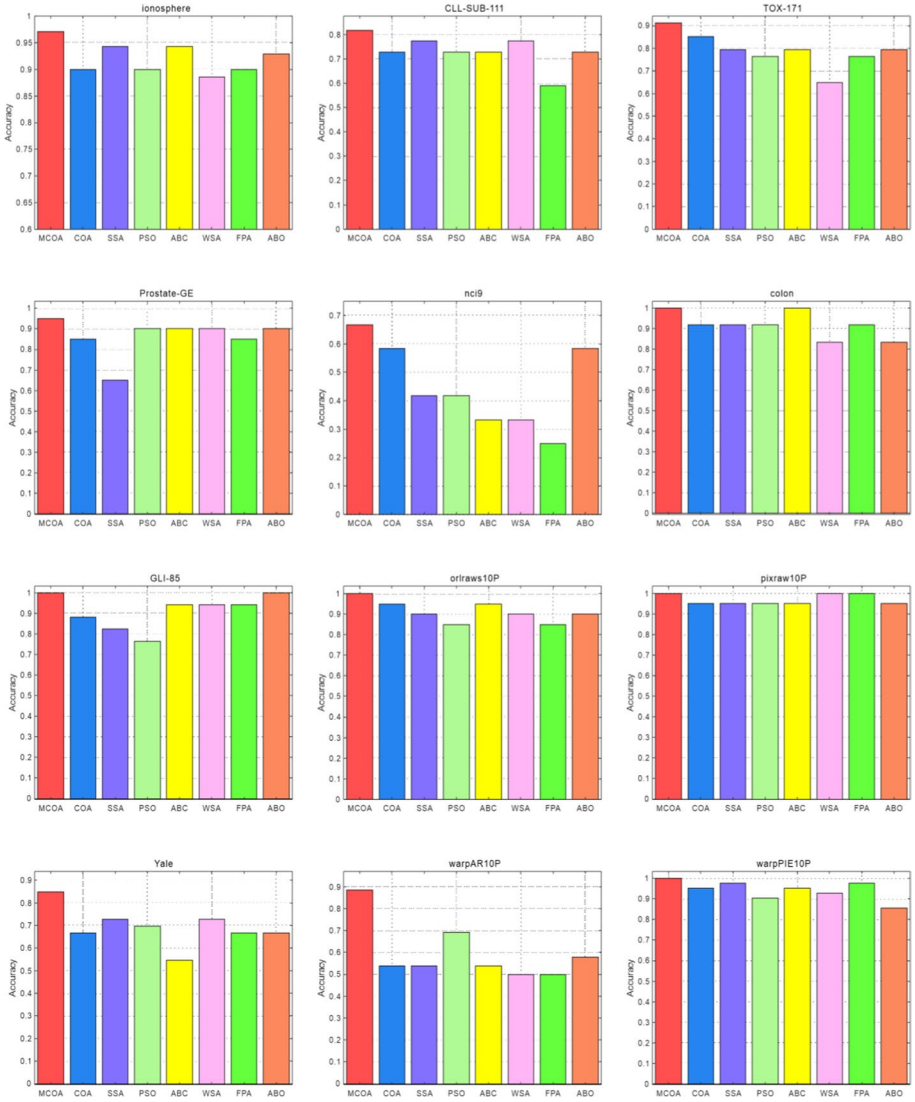


Fig. 11 Comparison plot of verification accuracy of eight algorithms

6 High-dimensional feature selection problem

The objective of feature selection is to eliminate redundant and irrelevant features, thereby obtaining a more accurate model. However, in high-dimensional feature spaces, feature selection encounters challenges such as high computational costs and susceptibility to overfitting. To tackle these issues, this paper propose novel high-dimensional feature selection methods based on metaheuristic algorithms. These methods aim to enhance the efficiency and effectiveness of feature selection in complex, high-dimensional datasets.

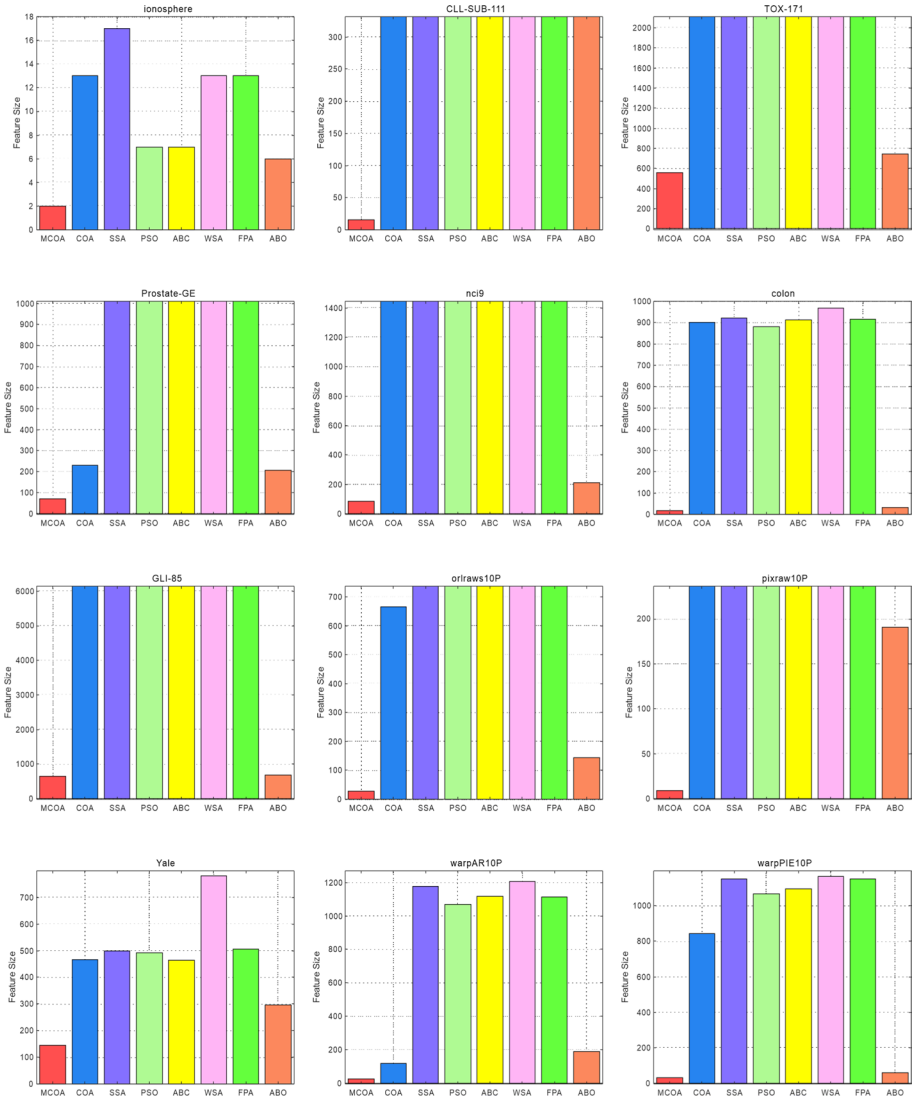


Fig. 12 Comparison plots of feature sizes of the eight algorithms

High-dimensional feature selection, as discussed in reference (Ghaemi and Feizi-Derakhshi 2016), focuses on processing high-dimensional data to extract relevant features while eliminating redundant and irrelevant ones. This process enhances the model's generalization ability and reduces computational costs. The problem of high-dimensional feature selection is often referred to as sparse modeling, encompassing two primary methods: filter and wrapper. Filter methods, also called classifier-independent methods, can be categorized into univariate and multivariate methods. Univariate methods consider individual features independently, leveraging the correlation and dependence within the data to quickly screen and identify the optimal feature subset. On the other hand, multivariate methods assess

Table 13 Results of FS fitness calculation (optimal data are marked in bold)

No	Dataset	Metric	Fitness							
			MCOA	**COA**	**SSA**	**PSO**	**ABC**	**WSA**	**FPA**	**ABO**
1	Ionosphere	mean	4.77E-02	7.42E-02	8.87E-02	6.63E-02	6.56E-02	1.11E-01	8.60E-02	7.91E-02
		std	1.98E-02	3.42E-02	2.40E-02	2.64E-02	1.73E-02	3.70E-02	2.25E-02	2.53E-02
		rank	1	4	7	2	3	8	6	5
2	CLL-SUB-111	mean	1.17E-01	2.41E-01	2.75E-01	2.57E-01	2.32E-01	3.23E-01	3.09E-01	2.89E-01
		std	4.94E-02	8.74E-02	7.73E-02	7.64E-02	8.94E-02	9.98E-02	8.98E-02	6.02E-02
		rank	1	2	6	4	3	8	7	5
3	TOX-171	mean	1.09E-01	1.44E-01	1.85E-01	1.29E-01	1.52E-01	2.44E-01	1.85E-01	2.19E-01
		std	4.02E-02	5.07E-02	6.02E-02	4.80E-02	5.86E-02	6.72E-02	4.68E-02	6.04E-02
		rank	1	3	5	2	4	8	6	7
4	Prostate-GE	mean	2.98E-02	9.50E-02	9.39E-02	8.88E-02	7.90E-02	8.92E-02	1.21E-01	8.75E-02
		std	3.37E-02	5.88E-02	9.72E-02	5.58E-02	6.12E-02	4.84E-02	6.28E-02	6.96E-02
		rank	1	5	6	3	4	7	8	2
5	Nci9	mean	3.26E-01	5.19E-01	5.78E-01	5.16E-01	4.83E-01	5.91E-01	5.70E-01	5.41E-01
		std	7.79E-02	1.14E-01	1.05E-01	1.06E-01	1.12E-01	1.22E-01	1.01E-01	9.83E-02
		rank	1	4	6	5	2	8	7	3
6	Colon	mean	7.88E-05	1.23E-01	1.49E-01	1.36E-01	1.04E-01	1.78E-01	1.45E-01	1.32E-01
		std	5.36E-05	9.83E-02	1.03E-01	9.79E-02	6.88E-02	1.28E-01	8.92E-02	8.20E-02
		rank	1	3	6	5	4	8	7	2
7	GLI-85	mean	2.34E-02	6.27E-02	7.19E-02	8.34E-02	7.19E-02	1.01E-01	6.32E-02	7.13E-02
		std	3.48E-02	7.11E-02	6.06E-02	6.62E-02	4.34E-02	7.62E-02	4.22E-02	6.36E-02
		rank	1	3	6	4	7	8	5	2
8	Orlraws10P	mean	7.46E-03	6.06E-02	8.65E-02	7.89E-02	8.89E-02	9.65E-02	8.65E-02	5.72E-02
		std	1.81E-02	3.76E-02	4.02E-02	3.00E-02	4.57E-02	4.33E-02	4.02E-02	3.68E-02
		rank	1	3	6	4	5	8	7	2

Table 13 (continued)

No	Dataset	Metric	Fitness							
			MCOA	**COA**	**SSA**	**PSO**	**ABC**	**WSA**	**FPA**	**ABO**
9	Pixraw10P	mean	2.49E-03	3.57E-02	4.94E-02	3.19E-02	3.44E-02	3.46E-02	4.45E-02	2.98E-02
		std	1.11E-02	2.44E-02	3.17E-02	3.39E-02	3.73E-02	3.37E-02	3.05E-02	3.37E-02
		rank	1	4	6	3	5	8	7	2
10	Yale	mean	2.73E-01	2.94E-01	3.11E-01	2.88E-01	3.17E-01	3.43E-01	3.11E-01	3.43E-01
		std	3.77E-02	7.11E-02	5.02E-02	6.57E-02	7.50E-02	6.94E-02	7.26E-02	5.18E-02
		rank	1	3	5	2	6	8	4	7
11	WarpAR10P	mean	2.29E-01	4.00E-01	4.10E-01	3.74E-01	3.93E-01	4.43E-01	4.29E-01	3.99E-01
		std	6.88E-02	6.06E-02	7.84E-02	7.92E-02	6.13E-02	1.22E-01	6.45E-02	6.97E-02
		rank	1	5	6	2	4	8	7	3
12	WarpPIE10P	mean	4.05E-02	8.50E-02	1.04E-01	6.23E-02	7.77E-02	9.35E-02	6.96E-02	6.89E-02
		std	2.55E-02	3.20E-02	4.36E-02	4.01E-02	4.04E-02	4.82E-02	4.18E-02	3.68E-02
		rank	1	6	8	2	5	7	4	3
Friedman average rank			1	3.75	6.0833	3.1667	4.3333	7.8333	6.25	3.5833
Friedman rank			1	4	6	2	5	8	7	3

relationships between multiple features simultaneously, aiming to comprehensively select the most informative feature combinations. Wrapper methods offer more diverse solutions. This approach treats feature selection as an optimization problem, utilizing specific performance measures of classifiers and objective functions. Wrapper methods continuously explore and evaluate various feature combinations to find the best set of features that maximizes the model's performance. Unlike filter methods, wrapper methods provide a more customized and problem-specific approach to feature selection.

Filter methods, being relatively single and one-sided, approach the problem of feature selection in a straightforward manner by considering individual features and their relationships within the dataset. However, they might lack the flexibility needed for complex and specific problem scenarios. However, wrapper methods offer tailored and problem-specific solutions. They exhibit strong adaptability, wide applicability, and high relevance to the specific problem at hand. Wrapper methods can be seamlessly integrated into any learning algorithm, allowing for a more customized and targeted approach to feature selection. By treating feature selection as an optimization problem and continuously evaluating different feature combinations, wrapper methods can maximize the effectiveness of the algorithm and optimize its performance to a greater extent compared to filter methods. In summary, wrapper methods provide a more sophisticated and problem-specific approach to feature selection, enabling the algorithm to achieve its maximum potential by selecting the most relevant and informative features for the given task.

6.1 Fitness function

In this subsection, the wrapper method in high-dimensional feature selection is elucidated, employing the classification error rate (CEE) (Wang et al. 2005) as an illustrative example. CEE is utilized as the fitness function or objective function to assess the optimization effectiveness of the feature selection algorithm for the problem at hand. Specifically, CEE quantifies the classification error rate when employing the k-nearest-neighbors (KNN) algorithm (Datasets | Feature Selection @ ASU. 2019), with the Euclidean distance (ED) (The UCI Machine Learning Repository xxxx) serving as the metric for measuring the distance between the current model being tested and its neighboring models. By using CEE as the fitness function, the wrapper method evaluates different feature subsets based on their performance in the context of the KNN algorithm. This approach enables the algorithm to identify the most relevant features that lead to the lowest classification error rate, thereby optimizing the model's performance. By focusing on the accuracy of classification in a specific algorithmic context, the wrapper method ensures that the selected features are highly tailored to the problem and the chosen learning algorithm. This targeted feature selection process enhances the overall performance and effectiveness of the algorithm in handling high-dimensional data.

$$\text{Fitness} = \text{CEE} = \frac{\text{No. of wrongly classified}}{\text{Total number of instances}} \tag{76}$$

$$\text{ED}(Y, X) = \sqrt{\sum_{d=1}^{D} \left(X^d - Y^d\right)^2} \tag{77}$$

where X denotes feat, Y denotes label, both X and Y are specific features in the given data model, and D is the total number of features recorded.

Table 14 FS accuracy calculation results (optimal data are marked in bold)

No	Dataset	Metric	Accuracy							
			MCOA	**COA**	**SSA**	**PSO**	**ABC**	**WSA**	**FPA**	**ABO**
1	Ionosphere	mean	9.53E-01	9.29E-01	9.14E-01	9.36E-01	9.37E-01	8.93E-01	9.17E-01	9.22E-01
		std	2.03E-02	3.44E-02	2.41E-02	2.68E-02	1.76E-02	3.72E-02	2.30E-02	2.60E-02
		rank	1	4	7	2	2	8	6	5
2	CLL-SUB-111	mean	8.82E-01	7.61E-01	7.27E-01	7.45E-01	7.70E-01	6.80E-01	6.93E-01	7.09E-01
		std	4.98E-02	8.83E-02	7.80E-02	7.72E-02	9.03E-02	1.02E-01	9.08E-02	6.15E-02
		rank	1	3	5	4	2	8	7	6
3	TOX-171	mean	8.91E-01	8.59E-01	8.18E-01	8.75E-01	8.51E-01	7.59E-01	8.18E-01	7.81E-01
		std	4.06E-02	5.10E-02	6.08E-02	4.85E-02	5.92E-02	6.79E-02	4.73E-02	6.15E-02
		rank	1	3	6	2	4	8	5	7
4	Prostate-GE	mean	9.70E-01	9.08E-01	9.10E-01	9.15E-01	9.25E-01	9.15E-01	8.83E-01	9.13E-01
		std	3.40E-02	5.91E-02	9.81E-02	5.64E-02	6.18E-02	4.89E-02	6.34E-02	7.05E-02
		rank	1	7	3	4	2	5	8	6
5	Nci9	mean	6.71E-01	4.79E-01	4.21E-01	4.83E-01	5.17E-01	4.08E-01	4.29E-01	4.54E-01
		std	7.87E-02	1.14E-01	1.06E-01	1.07E-01	1.13E-01	1.24E-01	1.02E-01	9.92E-02
		rank	1	4	7	3	2	8	6	5
6	Colon	mean	1	8.79E-01	8.54E-01	8.67E-01	9.00E-01	8.25E-01	8.58E-01	8.67E-01
		std	0	9.92E-02	1.04E-01	9.90E-02	6.95E-02	1.29E-01	9.01E-02	8.29E-02
		rank	1	3	7	4	2	8	6	5
7	GLI-85	mean	9.76E-01	9.41E-01	9.32E-01	9.21E-01	9.32E-01	9.03E-01	9.41E-01	9.29E-01
		std	3.52E-02	7.14E-02	6.12E-02	6.69E-02	4.38E-02	7.70E-02	4.27E-02	6.50E-02
		rank	1	2	4	7	5	8	3	6
8	Orlraws10P	mean	9.93E-01	9.43E-01	9.18E-01	9.25E-01	9.15E-01	9.08E-01	9.18E-01	9.43E-01
		std	1.83E-02	3.73E-02	4.06E-02	3.03E-02	4.62E-02	4.38E-02	4.06E-02	3.73E-02
		rank	1	2	5	4	7	8	5	2

Table 14 (continued)

No	Dataset	Metric	Accuracy							
			MCOA	**COA**	**SSA**	**PSO**	**ABC**	**WSA**	**FPA**	**ABO**
9	Pixraw10P	mean	9.98E-01	9.68E-01	9.55E-01	9.73E-01	9.70E-01	9.70E-01	9.60E-01	9.70E-01
		std	1.12E-02	2.45E-02	3.20E-02	3.43E-02	3.77E-02	3.40E-02	3.08E-02	3.40E-02
		rank	1	6	8	2	3	4	7	4
10	Yale	mean	7.26E-01	7.08E-01	6.91E-01	7.14E-01	6.85E-01	6.59E-01	6.91E-01	6.56E-01
		std	3.87E-02	7.18E-02	5.07E-02	6.63E-02	7.58E-02	7.01E-02	7.33E-02	5.23E-02
		rank	1	3	5	2	6	7	4	8
11	WarpAR10P	mean	7.69E-01	6.00E-01	5.90E-01	6.27E-01	6.08E-01	5.58E-01	5.71E-01	5.98E-01
		std	6.95E-02	6.16E-02	7.92E-02	8.00E-02	6.19E-02	1.23E-01	6.52E-02	7.11E-02
		rank	1	4	6	2	3	7	8	5
12	WarpPIE10P	mean	9.60E-01	9.18E-01	9.00E-01	9.42E-01	9.26E-01	9.11E-01	9.35E-01	9.32E-01
		std	2.57E-02	3.23E-02	4.41E-02	4.05E-02	4.08E-02	4.88E-02	4.22E-02	3.73E-02
		rank	1	6	8	2	5	7	3	4
Friedman Average rank			1	3.91667	5.9167	3.16667	3.58333	7.166667	5.66667	5.25
Friedman rank			1	4	7	2	3	8	6	5

Table 15 Experimental results of Wilcoxon rank sum test for FS

Dataset	COA vs MCOA	SSA vs MCOA	PSO vs MCOA	ABC vs MCOA	WSA vs MCOA	FPA vs MCOA	ABO vs MCOA
Ionosphere	1.97E-05	1.73E-06	1.48E-04	6.31E-05	2.13E-06	1.73E-06	6.33E-06
CLL-SUB-111	1.73E-06	1.73E-06	3.52E-06	2.35E-06	1.73E-06	1.92E-06	1.73E-06
TOX-171	1.97E-05	6.98E-06	3.61E-03	1.74E-04	1.73E-06	4.86E-05	1.73E-06
Prostate-GE	1.80E-05	5.79E-05	1.73E-06	2.05E-04	3.88E-06	3.11E-05	1.97E-05
Nci9	7.69E-06	2.35E-06	5.75E-06	6.32E-05	2.13E-06	4.29E-06	3.18E-06
Colon	3.18E-06	1.73E-06	1.73E-06	1.73E-06	3.18E-06	6.34E-06	4.73E-06
GLI-85	3.11E-05	1.73E-06	3.88E-06	1.73E-06	1.73E-06	5.75E-06	4.29E-06
Orlraws10P	2.88E-06	2.37E-05	1.24E-05	1.36E-05	4.29E-06	4.29E-06	2.02E-06
Pixraw10P	4.28E-06	5.21E-06	1.73E-06	1.73E-06	1.73E-06	4.73E-06	1.73E-06
Yale	3.88E-04	5.29E-04	2.70E-02	1.20E-03	1.89E-04	9.84E-03	1.15E-04
WarpAR10P	5.75E-06	5.75E-06	2.35E-06	3.88E-06	1.73E-06	2.35E-06	3.88E-06
WarpPIE10P	1.89E-04	2.16E-05	5.79E-05	1.15E-04	1.02E-05	7.69E-06	1.13E-05
+/=/−	12/0/0	12/0/0	12/0/0	12/0/0	12/0/0	12/0/0	12/0/0

In the experimental setup, each dataset is partitioned into a training set and a test set, with an 80% and 20% ratio. The training set is initially utilized to select the most characteristic features and fine-tune the parameters of the KNN model. Subsequently, the test set is employed to evaluate and calculate the data model and algorithm performance. To address concerns related to fitting ability and overfitting, hierarchical cross-validation with $K = 10$ was employed in this experiment. In hierarchical cross-validation, the training portion is divided into ten equal-sized subsets. The KNN classifier is trained using 9 out of the 10 folds (K-1 folds) to identify the optimal KNN classifier, while the remaining fold is used for validation purposes. This process is repeated 10 times, ensuring that each subset serves both as a validation set and as part of the training data. This iterative approach is a crucial component of our evaluation methodology, providing a robust assessment of the algorithm's performance. By repeatedly employing replacement validation and folding training, we enhance the reliability and accuracy of our evaluation, enabling a comprehensive analysis of the algorithm's effectiveness across various datasets.

6.2 High-dimensional datasets

In this subsection, the optimization performance of MCOA is assessed using 12 high-dimensional datasets sourced from the Arizona State University (Too et al. 2021) and University of California Irvine (UCI) Machine Learning databases (Chandrashekar and Sahin 2014). By conducting experiments on these high-dimensional datasets, the results obtained are not only convincing but also pose significant challenges. These datasets authentically capture the intricacies of real-life spatial problems, making the experiments more meaningful and applicable to complex and varied spatial scenarios. For a detailed overview of the 12 high-dimensional datasets, please refer to Table 12.

6.3 Experimental results and analysis

In order to assess the effectiveness and efficiency of MCOA in feature selection, we conducted comparative tests using MCOA as well as several other algorithms including COA, SSA, PSO, ABC, WSA (Baykasoğlu et al. 2020), FPA (Yang 2012), and ABO (Qi et al. 2017) on 12 datasets. In this section of the experiment, the fitness value of each algorithm was calculated, and the convergence curve, feature selection accuracy (FS Accuracy), and selected feature size for each algorithm were analyzed. Figures 10, 11 and 12 display the feature selection (FS) convergence curve, FS Accuracy, and selected feature size for the eight algorithms across the 12 datasets. From these figures, it is evident that the optimization ability and prediction accuracy of the MCOA algorithm surpass those of the other seven comparison algorithms. Taking the dataset CLL-SUB-111 as an example in Figs. 11 and 12, MCOA selected 20 features, while the other seven algorithms selected more than 2000 features. Moreover, the prediction accuracy achieved by MCOA was higher than that of the other seven algorithms. Across all 12 datasets, the comparison figures indicate that the MCOA algorithm consistently outperforms the others. Specifically, the MCOA algorithm tends to select smaller feature subsets, leading to higher prediction accuracy and stronger optimization capabilities. This pattern highlights the superior performance of MCOA in feature selection, demonstrating its effectiveness in optimizing feature subsets for improved prediction accuracy.

To address the randomness and instability inherent in experiments, a single experiment may not fully demonstrate the effectiveness of algorithm performance. Therefore, we conducted 30 independent experiments using 12 datasets and 8 algorithms. For each algorithm and dataset combination, we calculated the average fitness value, standard deviation of the fitness value, and Friedman rank. Subsequently, the Wilcoxon rank sum test was employed to determine significant differences between the performance of different algorithms across various datasets. Throughout the experiment, a fixed population size of 10 and a maximum of 100 iterations were used. The 12 datasets were utilized to evaluate the 8 algorithms 300 times (tenfold cross-validation $\times$ 30 runs). It is essential to note that all algorithms were assessed using the same fitness function derived from the dataset, ensuring a consistent evaluation criterion across the experiments. By conducting multiple independent experiments and statistical analyses, the study aimed to provide a comprehensive and robust assessment of algorithm performance. This approach helps in drawing reliable conclusions regarding the comparative effectiveness of the algorithms under consideration across different datasets, accounting for the inherent variability and randomness in the experimental process.

Table 13 presents the average fitness calculation results from 30 independent experiments for the eight algorithms, it is 55.23% higher than the original algorithm. According to the table, in the Ionosphere dataset, MCOA exhibits the best average fitness, albeit with slightly lower stability compared to ABC. Similarly, in the WarpAR10P dataset, MCOA achieves the best average fitness, with stability slightly lower than COA. After conducting Friedman ranking on the fitness calculation results of the 30 independent experiments, it is concluded that although MCOA shows slightly lower stability in some datasets, it ranks first overall. Among the other seven algorithms, PSO ranks second, ABO ranks third, COA ranks fourth, and ABC, SSA, FPA, and WSA rank fifth to ninth, respectively. These results demonstrate that MCOA exhibits robust optimization performance and high stability in solving high-dimensional feature selection problems. Moreover, MCOA outperforms COA, showcasing its superior improvement in solving these complex problems.

Table 14 presents the accuracy calculation results of the eight algorithms for 30 independent experiments, it is 10.85% higher than the original algorithm. According to the table, the average accuracy of MCOA is the highest across all datasets. Notably, in the Colon dataset, MCOA performs exceptionally well with a perfect average accuracy of 100%. However, in the Ionosphere dataset, MCOA exhibits slightly lower stability compared to ABC, and in the WarpAR10P dataset, it is slightly less stable than COA. Upon conducting Friedman ranking on the average accuracy calculation results of 30 independent experiments, it is evident that MCOA ranks first overall. Among the other seven algorithms, PSO ranks second, ABC ranks third, COA ranks fourth, and ABO, FPA, SSA, and WSA rank fifth to ninth, respectively. These results highlight that MCOA consistently achieves high accuracy and stability in solving high-dimensional feature selection problems. Its superior performance across various datasets underscores its effectiveness and reliability in real-world applications.

Table 15 demonstrates that the MCOA algorithm has shown significant results in the Wilcoxon rank sum test for high-dimensional feature selection fitness. The comparison values with other algorithms are less than 5%, indicating that the MCOA algorithm exhibits significant differences compared to the other seven algorithms. This result serves as evidence that MCOA outperforms the other algorithms, showcasing its superior optimization performance. Additionally, when comparing the results with the original algorithm, it becomes evident that the MCOA algorithm has a substantial and positive impact, demonstrating its effectiveness and improvement over existing methods. These findings underscore the algorithm's potential and its ability to provide substantial enhancements in the field of high-dimensional feature selection.

7 Conclusions and future work

The Crayfish Optimization Algorithm (COA) is grounded in swarm intelligence, drawing inspiration from crayfish behavior to find optimal solutions within a specific range. However, COA's limitations stem from neglecting crucial survival traits of crayfish, such as crawling against water to discover better aquatic environments. This oversight weakens COA's search ability, making it susceptible to local optima and hindering its capacity to find optimal solutions. To address these issues, this paper introduces a Modified Crayfish Optimization Algorithm (MCOA). MCOA incorporates an environmental updating mechanism, enabling crayfish to randomly select directions toward better aquatic environments for location updates, enhancing search ability. The addition of the ghost opposition-based learning strategy expands MCOA's search range and promotes escape from local optima. Experimental validations using IEEE CEC2020 benchmark functions confirm MCOA's outstanding optimization performance.

Moreover, MCOA's practical applicability is demonstrated through applications to four constrained engineering problems and high-dimensional feature selection challenges. These experiments underscore MCOA's efficacy in real-world scenarios, but MCOA can only solve the optimization problem of a single goal. In future studies, efforts will be made to further optimize MCOA and enhance its function. We will exploitation multi-objective version of the algorithm to increase the search ability and convergence of the algorithm through non-dominated sorting, multi-objective selection, crossover and mutation, etc., to solve more complex practical problems. It is extended to wireless sensor network coverage, machine learning, image segmentation and other practical applications.

Acknowledgements The authors would like to thank the support of Fujian Key Lab of Agriculture IOT Application, IOT Application Engineering Research Center of Fujian Province Colleges and Universities, Guiding Science and Technology Projects in Sanming City (2023-G-5), Industry-University Cooperation Project of Fujian Province (2021H6039), Fujian Province Industrial Guidance (Key) Project (2022H0053), Sanming Major Science and Technology Project of Industry-University-Research Collaborative Innovation (2022-G-4), and also the anonymous reviewers and the editor for their careful reviews and constructive suggestions to help us improve the quality of this paper.

Author contributions Heming Jia: Methodology, Formal analysis, Investigation, Resources, Funding acquisition, Project administration; Xuelian Zhou: Investigation, Conceptualization, Software, Data Curation, Writing—Original Draft; Jinrui Zhang: Validation, Conceptualization; Laith Abualigah: Supervision, Writing—Review & Editing; Ali Riza Yildiz: Visualization, Writing—Review & Editing; Abdelazim G. Hussien: Writing—Review & Editing.

Data availability The datasets generated during and/or analyzed during the current study are available from the corresponding author on reasonable request.

Declarations

Competing interest We declare that we have no conflict of interest.

References

Abualigah L, Diabat A, Mirjalili S, Abd Elaziz M, Gandomi AH (2021) The arithmetic optimization algorithm. Comput Methods Appl Mech Eng 376:113609. https://doi.org/10.1016/j.cma.2020.113609

Abualigah L, Elaziz MA, Khasawneh AM, Alshinwan M, Ibrahim RA, Al-Qaness MA, Gandomi AH (2022) Meta-heuristic optimization algorithms for solving real-world mechanical engineering design problems: a comprehensive survey, applications, comparative analysis, and results. Neural Comput Appl. https://doi.org/10.1007/s00521-021-06747-4

Ahmed AM, Rashid TA, Saeed SAM (2020) Cat swarm optimization algorithm: a survey and performance evaluation. Comput Intell Neurosci. https://doi.org/10.1155/2020/4854895

Baykasoğlu A, Ozsoydan FB, Senol ME (2020) Weighted superposition attraction algorithm for binary optimization problems. Oper Res Int Journal 20:2555–2581. https://doi.org/10.1007/s12351-018-0427-9

Baykasoglu A, Ozsoydan FB (2015) Adaptive firefly algorithm with chaos for mechanical design optimization problems. Appl Soft Comput 36:152–164. https://doi.org/10.1016/j.asoc.2015.06.056

Belge E, Altan A, Hacıoğlu R (2022) Metaheuristic optimization-based path planning and tracking of quadcopter for payload hold-release mission. Electronics 11(8):1208. https://doi.org/10.3390/electronics11081208

Beyer HG, Schwefel HP (2002) Evolution strategies–a comprehensive introduction. Nat Comput 1:3–52. https://doi.org/10.1023/A:1015059928466

Chandrashekar G, Sahin F (2014) A survey on feature selection methods. Comput Electr Eng 40(1):16–28. https://doi.org/10.1016/j.compeleceng.2013.11.024

Cherrington, M., Thabtah, F., Lu, J., & Xu, Q. (2019, April). Feature selection: filter methods performance challenges. In 2019 International Conference on Computer and Information Sciences (ICCIS) (pp. 1–4). IEEE. https://doi.org/10.1109/ICCISci.2019.8716478

Datasets | Feature Selection @ ASU. Accessed from 3 Oct 2019 https://jundongl.github.io/scikit-feature/OLD/home_old.html.

Deng L, Liu S (2023) Snow ablation optimizer: a novel metaheuristic technique for numerical optimization and engineering design. Expert Syst Appl 225:120069. https://doi.org/10.1016/j.eswa.2023.120069
Dhiman G, Kaur A (2019) STOA: a bio-inspired based optimization algorithm for industrial engineering problems. Eng Appl Artif Intell 82:148–174. https://doi.org/10.1016/j.engappai.2019.03.021
Dorigo M, Birattari M, Stutzle T (2006) Ant colony optimization. IEEE Comput Intell Mag 1(4):28–39. https://doi.org/10.1109/MCI.2006.329691
Eskandar H, Sadollah A, Bahreininejad A, Hamdi M (2012) Water cycle algorithm–a novel metaheuristic optimization method for solving constrained engineering optimization problems. Comput Struct 110:151–166. https://doi.org/10.1016/j.compstruc.2012.07.010
Espejo, P. G., Ventura, S., & Herrera, F. (2009) A survey on the application of genetic programming to classification. IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews). 40(2): 121–144. https://doi.org/10.1109/TSMCC.2009.2033566
Ezugwu AE, Agushaka JO, Abualigah L, Mirjalili S, Gandomi AH (2022) Prairie dog optimization algorithm. Neural Comput Appl 34(22):20017–20065. https://doi.org/10.1007/s00521-022-07530-9
Faramarzi A, Heidarinejad M, Mirjalili S, Gandomi AH (2020) Marine predators algorithm: a nature-inspired metaheuristic. Expert Syst Appl 152:113377. https://doi.org/10.1016/j.eswa.2020.113377
Formato RA (2007) Central force optimization. Prog Electromagn Res 77(1):425–491. https://doi.org/10.2528/PIER07082403
Ghaemi M, Feizi-Derakhshi MR (2016) Feature selection using forest optimization algorithm. Pattern Recogn 60:121–129. https://doi.org/10.1016/j.patcog.2016.05.012
Guedria NB (2016) Improved accelerated PSO algorithm for mechanical engineering optimization problems. Appl Soft Comput 40:455–467. https://doi.org/10.1016/j.asoc.2015.10.048
He Q, Wang L (2007) An effective co-evolutionary particle swarm optimization for constrained engineering design problems. Eng Appl Artif Intel 20:89–99. https://doi.org/10.1016/j.engappai.2006.03.003
Heidari AA, Mirjalili S, Faris H, Aljarah I, Mafarja M, Chen H (2019) Harris hawks optimization: algorithm and applications. Futur Gener Comput Syst 97:849–872. https://doi.org/10.1016/j.future.2019.02.028
Jacob DIJ, Darney DPE (2021) Artificial bee colony optimization algorithm for enhancing routing in wireless networks. J Artif Intell Capsule Networks 3(1):62–71. https://doi.org/10.36548/jaicn.2021.1.006
Jia H, Peng X, Lang C (2021) Remora optimization algorithm. Expert Syst Appl 185:115665. https://doi.org/10.1016/j.eswa.2021.115665
Jia H, Wen Q, Wu D, Wang Z, Wang Y, Wen C, Abualigah L (2023a) Modified beluga whale optimization with multi-strategies for solving engineering problems. J Comput Design Eng 10(6):2065–2093. https://doi.org/10.1093/jcde/qwad089
Jia H, Rao H, Wen C, Mirjalili S (2023b) Crayfish optimization algorithm. Artif Intell Rev. https://doi.org/10.1007/s10462-023-10567-4
Jia H, Lu C, Wu D, Wen C, Rao H, Abualigah L (2023c) An improved reptile search algorithm with ghost opposition-based learning for global optimization problems. J Comput Design Eng. https://doi.org/10.1093/jcde/qwad048
Jović, A., Brkić, K., & Bogunović, N. (2015). A review of feature selection methods with applications. In 2015 38th international convention on information and communication technology, electronics and microelectronics (MIPRO) (pp. 1200–1205). IEEE. https://doi.org/10.1109/MIPRO.2015.7160458
Kamal M, Mortazavi A, Cakici Z (2023) Optimal design of RC bracket and footing systems of precast industrial buildings using fuzzy differential evolution incorporated virtual mutant. Arabian J Sci Eng. https://doi.org/10.3934/mbe.2022263
Kandemir EC, Mortazavi A (2022) Optimization of seismic base isolation system using a fuzzy reinforced swarm intelligence. Adv Eng Softw 174:103323. https://doi.org/10.1016/j.advengsoft.2022.103323
Kaveh A (2017) Applications of metaheuristic optimization algorithms in civil engineering. Springer International Publishing, Basel, Switzerland. https://doi.org/10.1007/978-3-319-48012-1
Kaveh A, Khayatazad M (2012) A new meta-heuristic method: ray optimization. Comput Struct 112:283–294. https://doi.org/10.1016/j.compstruc.2012.09.003
Kaveh A, Mahdavi V (2014) Colliding bodies optimization: a novel meta-heuristic method. Comput Struct 139:18–27. https://doi.org/10.1016/j.compstruc.2014.04.005
Kira, K., & Rendell, L. A. (1992). The feature selection problem: Traditional methods and a new algorithm. In Proceedings of the tenth national conference on Artificial intelligence (pp. 129–134). https://doi.org/10.5555/1867135.1867155
Kiran MS (2015) TSA: tree-seed algorithm for continuous optimization. Expert Syst Appl 42(19):6686–6698. https://doi.org/10.1016/j.eswa.2015.04.055

Kumar S, Tejani GG, Mirjalili S (2019) Modified symbiotic organisms search for structural optimization. Eng with Comput 35(4):1269–1296. https://doi.org/10.1007/s00366-018-0662-y
Kuo HC, Lin CH (2013) Cultural evolution algorithm for global optimizations and its applications. J Appl Res Technol 11(4):510–522
Liu X, Lu P (2014) Solving nonconvex optimal control problems by convex optimization. J Guid Control Dyn 37(3):750–765. https://doi.org/10.2514/1.62110
Lu P, Ye L, Zhao Y, Dai B, Pei M, Tang Y (2021) Review of meta-heuristic algorithms for wind power prediction: methodologies, applications and challenges. Appl Energy 301:117446. https://doi.org/10.1016/j.apenergy.2021.117446
Ma Y, Zhang X, Song J, Chen L (2021) A modified teaching–learning-based optimization algorithm for solving optimization problem. Knowl-Based Syst 212:106599. https://doi.org/10.1016/j.knosys.2023.110554
Mahdavi M, Fesanghary M, Damangir E (2007) An improved harmony search algorithm for solving optimization problems. Appl Math Comput 188(2):1567–1579. https://doi.org/10.1016/j.amc.2006.11.033
Mahdavi S, Rahnamayan S, Deb K (2018) Opposition based learning: a literature review. Swarm Evol Comput 39:1–23. https://doi.org/10.1016/j.swevo.2017.09.010
Meloni C, Pacciarelli D, Pranzo M (2004) A rollout metaheuristic for job shop scheduling problems. Ann Oper Res 131:215–235. https://doi.org/10.1023/B:ANOR.0000039520.24932.4b
Mirjalili S (2015) Moth-flame optimization algorithm: a novel nature-inspired heuristic paradigm. Knowl-Based Syst 89:228–249. https://doi.org/10.1016/j.knosys.2015.07.006
Mirjalili S (2016) SCA: a sine cosine algorithm for solving optimization problems. Knowl-Based Syst 96:120–133. https://doi.org/10.1016/j.knosys.2015.12.022
Mirjalili S, Mirjalili SM, Hatamlou A (2016) Multi-verse optimizer: a nature-inspired algorithm for global optimization. Neural Comput Appl 27:495–513. https://doi.org/10.1007/s00521-015-1870-7
Mirjalili S, Gandomi AH, Mirjalili SZ, Saremi S, Faris H, Mirjalili SM (2017) Salp swarm algorithm: a bio-inspired optimizer for engineering design problems. Adv Eng Softw 114:163–191. https://doi.org/10.1016/j.advengsoft.2017.07.002
Mirjalili, S., & Mirjalili, S. (2019). Genetic algorithm. Evolutionary Algorithms and Neural Networks: Theory and Applications, 43–55. https://doi.org/10.1007/978-3-319-93025-1_4
Moghdani R, Salimifard K (2018) Volleyball premier league algorithm. Appl Soft Comput 64:161–185. https://doi.org/10.1016/j.asoc.2017.11.043
Moloodpoor M, Mortazavi A (2022) Simultaneous optimization of fuel type and exterior walls insulation attributes for residential buildings using a swarm intelligence. Int J Environ Sci Technol 19(4):2809–2822. https://doi.org/10.1007/s13762-021-03323-0
Moloodpoor M, Mortazavi A, Özbalta N (2021) Thermo-economic optimization of double-pipe heat exchanger using a compound swarm intelligence. Heat Transfer Res. https://doi.org/10.1615/HeatTransRes.2021037293
Mortazavi A (2019) Comparative assessment of five metaheuristic methods on distinct problems. Dicle Üniversitesi Mühendislik Fakültesi Mühendislik Dergisi 10(3):879–898. https://doi.org/10.24012/dumf.585790
Papaioannou G, Koulocheris D (2018) An approach for minimizing the number of objective functions in the optimization of vehicle suspension systems. J Sound Vib 435:149–169. https://doi.org/10.1016/j.jsv.2018.08.009
Piotrowski AP (2018) L-SHADE optimization algorithms with population-wide inertia. Inf Sci 468:117–141. https://doi.org/10.1016/j.ins.2018.08.030
Qi X, Zhu Y, Zhang H (2017) A new meta-heuristic butterfly-inspired algorithm. Journal of Computational Science 23:226–239. https://doi.org/10.1016/j.jocs.2017.06.003
Rao H, Jia H, Wu D, Wen C, Li S, Liu Q, Abualigah L (2022) A modified group teaching optimization algorithm for solving constrained engineering optimization problems. Mathematics 10(20):3765. https://doi.org/10.3390/math10203765
Rao, R. V., & Rao, R. V. (2016). Teaching-learning-based optimization algorithm (pp. 9–39). Springer International Publishing. https://doi.org/10.1016/j.cad.2010.12.015
Rashedi E, Nezamabadi-Pour HS (2009) GSA: a gravitational search algorithm. Inform Sci 179:2232–2248. https://doi.org/10.1016/j.ins.2009.03.004
Razmjooy, N., Ashourian, M., & Foroozandeh, Z. (Eds). (2021). Metaheuristics and optimization in computer and electrical engineering. https://doi.org/10.1007/978-3-030-56689-0
Sadollah A, Bahreininejad A, Eskandar H, Hamdi M (2013) Mine blast algorithm: a new population based algorithm for solving constrained engineering optimization problems. Appl Soft Comput 13:2592–2612. https://doi.org/10.1016/j.asoc.2012.11.026

Saremi S, Mirjalili S, Lewis A (2017) Grasshopper optimisation algorithm: theory and application. Adv Eng Softw 105:30–47. https://doi.org/10.1016/j.advengsoft.2017.01.004
Sayed GI, Darwish A, Hassanien AE (2018) A new chaotic multi-verse optimization algorithm for solving engineering optimization problems. J Exp Theor Artif Intell 30(2):293–317. https://doi.org/10.1080/0952813X.2018.1430858
Shadravan S, Naji HR, Bardsiri VK (2019) The sailfish optimizer: a novel nature-inspired metaheuristic algorithm for solving constrained engineering optimization problems. Eng Appl Artif Intell 80:20–34. https://doi.org/10.1016/j.engappai.2019.01.001
Simon D (2008) Biogeography-based optimization. IEEE Trans Evol Comput 12(6):702–713. https://doi.org/10.1109/TEVC.2008.919004
Storn R, Price K (1997) Differential evolution–a simple and efficient heuristic for global optimization over continuous spaces. J Global Optim 11:341–359. https://doi.org/10.1023/A:1008202821328
Su H, Zhao D, Heidari AA, Liu L, Zhang X, Mafarja M, Chen H (2023) RIME: a physics-based optimization. Neurocomputing 532:183–214. https://doi.org/10.1016/j.neucom.2023.02.010
Talatahari S, Azizi M, Gandomi AH (2021) Material generation algorithm: a novel metaheuristic algorithm for optimization of engineering problems. Processes 9(5):859. https://doi.org/10.3390/pr9050859
Markelle Kelly, Rachel Longjohn, Kolby Nottingham, The UCI Machine Learning Repository, https://archive.ics.uci.edu
Too J, Mafarja M, Mirjalili S (2021) Spatial bound whale optimization algorithm: an efficient high-dimensional feature selection approach. Neural Comput Appl 33:16229–16250. https://doi.org/10.1007/s00521-021-06224-y
Wang L, Zhang Y, Feng J (2005) On the Euclidean distance of images. IEEE Trans Pattern Anal Mach Intell 27(8):1334–1339. https://doi.org/10.1109/TPAMI.2005.165
Wang D, Tan D, Liu L (2018a) Particle swarm optimization algorithm: an overview. Soft Comput 22:387–408. https://doi.org/10.1007/s00500-016-2474-6
Wang H, Hu Z, Sun Y, Su Q, Xia X (2018b) Modified backtracking search optimization algorithm inspired by simulated annealing for constrained engineering optimization problems. Comput Intell Neurosci. https://doi.org/10.1155/2018/9167414
Wang S, Hussien AG, Jia H, Abualigah L, Zheng R (2022) Enhanced remora optimization algorithm for solving constrained engineering optimization problems. Mathematics 10(10):1696. https://doi.org/10.3390/math10101696
Wu D, Rao H, Wen C, Jia H, Liu Q, Abualigah L (2022) Modified sand cat swarm optimization algorithm for solving constrained engineering optimization problems. Mathematics 10(22):4350. https://doi.org/10.3390/math10224350
Yang, X. S. (2011). Metaheuristic optimization: algorithm analysis and open problems. In International symposium on experimental algorithms (pp. 21–32). Berlin, Heidelberg: Springer Berlin Heidelberg. https://doi.org/10.1007/978-3-642-20662-7_2
Yang, X. S. (2012, September). Flower pollination algorithm for global optimization. In International conference on unconventional computing and natural computation (pp. 240–249). Berlin, Heidelberg: Springer Berlin Heidelberg. https://doi.org/10.1007/978-3-642-32894-7_27
Yang XS, Deb S (2014) Cuckoo search: recent advances and applications. Neural Comput Appl 24:169–174. https://doi.org/10.1007/s00521-013-1367-1
Yeniay Ö (2005) Penalty function methods for constrained optimization with genetic algorithms. Math Comput Appl 10(1):45–56. https://doi.org/10.3390/mca10010045
Yıldız BS, Kumar S, Panagant N, Mehta P, Sait SM, Yildiz AR, Mirjalili S (2023) A novel hybrid arithmetic optimization algorithm for solving constrained optimization problems. Knowledge-Based Syst 271:110554. https://doi.org/10.1016/j.knosys.2023.110554
Yuan Y, Shen Q, Wang S, Ren J, Yang D, Yang Q, Mu X (2023) Coronavirus mask protection algorithm: a new bio-inspired optimization algorithm and its applications. J Bionic Eng. https://doi.org/10.1007/s42235-023-00359-5
Zhang Y, Jin Z (2020) Group teaching optimization algorithm: a novel metaheuristic method for solving global optimization problems. Expert Syst Appl 148:113246. https://doi.org/10.1016/j.eswa.2020.113246
Zhang YJ, Wang YF, Tao LW, Yan YX, Zhao J, Gao ZM (2022a) Self-adaptive classification learning hybrid JAYA and Rao-1 algorithm for large-scale numerical and engineering problems. Eng Appl Artif Intell 114:105069. https://doi.org/10.1016/j.engappai.2022.105069
Zhang Y, Wang Y, Li S, Yao F, Tao L, Yan Y, Gao Z (2022b) An enhanced adaptive comprehensive learning hybrid algorithm of Rao-1 and JAYA algorithm for parameter extraction of photovoltaic models. Math Biosci Eng 19(6):5610–5637. https://doi.org/10.3934/mbe.2022263

Zhang YJ, Yan YX, Zhao J, Gao ZM (2022c) AOAAO: The hybrid algorithm of arithmetic optimization algorithm with aquila optimizer. IEEE Access 10:10907–10933. https://doi.org/10.1109/ACCESS.2022.3144431

Zhang YJ, Yan YX, Zhao J, Gao ZM (2022d) CSCAHHO: chaotic hybridization algorithm of the Sine Cosine with Harris Hawk optimization algorithms for solving global optimization problems. PLoS ONE 17(5):e0263387. https://doi.org/10.1371/journal.pone.0263387

Zhang YJ, Wang YF, Yan YX, Zhao J, Gao ZM (2022e) LMRAOA: An improved arithmetic optimization algorithm with multi-leader and high-speed jum** based on opposition-based learning solving engineering and numerical problems. Alex Eng J 61(12):12367–12403. https://doi.org/10.1016/j.aej.2022.06.017

Zhao J, Zhang Y, Li S, Wang Y, Yan Y, Gao Z (2022) A chaotic self-adaptive JAYA algorithm for parameter extraction of photovoltaic models. Math Biosci Eng 19:5638–5670. https://doi.org/10.3934/mbe.2022264

Zhao S, Zhang T, Ma S, Wang M (2023) Sea-horse optimizer: a novel nature-inspired meta-heuristic for global optimization problems. Appl Intell 53(10):11833–11860. https://doi.org/10.1007/s10489-022-03994-3

Authors and Affiliations

Heming Jia[1] · Xuelian Zhou[1] · Jinrui Zhang[1] · Laith Abualigah[2,3] · Ali Riza Yildiz[4] · Abdelazim G. Hussien[5]

✉ Heming Jia
jiaheming@fjsmu.edu.cn

[1] School of Information Engineering, Sanming University, Sanming 365004, China

[2] Hourani Center for Applied Scientific Research, Al-Ahliyya Amman University, Amman 19328, Jordan

[3] MEU Research Unit, Middle East University, Amman 11831, Jordan

[4] Department of Mechanical Engineering, Bursa Uludağ University, 16059 Görükle, Bursa, Turkey

[5] Department of Computer and Information Science, Linköping University, 58183 Linköping, Sweden