International Journal of Machine Learning and Cybernetics
https://doi.org/10.1007/s13042-025-02792-w

ORIGINAL ARTICLE

Multi-strategy boosted dung beetle optimizer and its applications for photovoltaic models and engineering applications

Xiaoyu Liu[1] · Qingke Zhang[1] · Junqing Li[1] · Huaxiang Zhang[1]

Received: 18 November 2024 / Accepted: 22 August 2025

Abstract
Parameters identification of photovoltaic models and engineering problems with constraints are recognized as complex optimization tasks in real-world applications. In this paper, an upgraded variant of Dung Beetle Optimizer (MSDBO) is proposed. MSDBO incorporates three key strategies. Firstly, dynamic population size variation dynamically adjusts the population size to balance global exploration in the early stages and local exploitation in the later stages. Secondly, forced ranking strategy prioritizes higher-quality solutions, accelerating the convergence speed by focusing on the most promising candidates. Finally, boundary control strategy ensures that candidate solutions remain within the feasible region of the search space, improving the robustness of the algorithm and preventing infeasible solutions. These strategies collectively contribute to the improved performance of MSDBO in terms of convergence speed, solution accuracy, and computational efficiency. By leveraging information from high-performing individuals, the algorithm is guided toward more effective solutions. To verify the performance of MSDBO, it was compared with other advanced optimization algorithms on the CEC 2014, CEC 2019, and CEC 2011 test suites. In addition, MSDBO was applied to two types of real-world optimization problems: several specific engineering problems with constraints and parameters identification of photovoltaic models. The experimental results indicate that MSDBO performs competitively compared to other algorithms. Statistical analysis, including the Friedman rank test, reveals that MSDBO ranks first among all tested algorithms, suggesting its effectiveness in both benchmark and real-world applications.

Keywords Swarm intelligence · Dung beetle optimizer (DBO) · Metaheuristic algorithms · Photovoltaic models · Engineering applications

1 Introduction

Traditional optimization algorithms, such as linear programming, branch binding, and dynamic programming, are limited in their ability to solve complex optimization problems in real-world engineering fields. Those algorithms require a precise mathematical formula of such problems and are only applicable to small-scale problems. However, with the continuous development of science and engineering fields, many complex engineering optimization problems emerge and are difficult to solve [1, 2]. To address these challenges, metaheuristic algorithms have emerged as a new approach for solving optimization problems, particularly for characteristics functions defined in continuous space, such as nonlinear and non-convex functions [3, 4].

Due to their inherent randomness and the ability to treat problems as black boxes, meta-heuristics can efficiently identify global optimal solutions for given problems. Metaheuristic algorithms can be roughly divided into the following two categories: Evolutionary Computation (EC) and Swarm Intelligence (SI) [5]. EC algorithms are inspired by the process of natural evolution. They use genetic operators to create new candidate solutions, and the selection of best solutions is based on a fitness function. While the SI algorithms are inspired by the collective behavior of social animals. They use simple rules to govern the behavior of a group of agents.

Metaheuristic algorithms(MAs), are gaining popularity due to their ability to calculate solutions through heuristic search strategies, greatly increasing their applicability in

✉ Qingke Zhang
tsingke@sdnu.edu.cn

[1] School of Information Science and Engineering, Shandong Normal University, Jinan 250358, China

Published online: 22 September 2025

practical problems. Additionally, these algorithms does not require the features and gradient information of problems to be solved. Unlike traditional deterministic methods, metaheuristic algorithms are composed of simple rules and random collaboration to perform search tasks. Based on these advantages, they have been successful applied to a wide range of optimization problems such as image processing [6, 7], classification [8], and feature selection [9].

In this context, metaheuristic algorithms are employed to optimize various parameters such as feature matching, transformation estimation, and blending, aiming to create seamless and visually appealing stitched images. For example, algorithms like the Genetic Algorithm or Particle Swarm Optimization can be utilized to search for the optimal alignment of overlapping image regions, thereby enhancing the overall quality of the stitched panorama [10]. In [11], an enhanced moth flame optimization algorithm was proposed and applied to COVID-19 CT image segmentation, achieving better results compared to other methods. In the area of engineering design, metaheuristic algorithms have been employed to solve complex optimization problems. In [12], a modified moth flame optimization algorithm was utilized to solve three-bar truss design and tension/compression spring design problems, demonstrating its effectiveness in real-world engineering applications. Additionally, in [13], a hybrid moth flame optimization algorithm was applied to global optimization problems and showed superior performance compared to traditional optimization algorithms. In [14], an improved whale optimization algorithm based on the Fibonacci search principle was proposed for global optimization, further expanding the application and improvement of metaheuristic algorithms.

In addition, works such as [15–18] have also made valuable contributions to the continuous evolution and diverse applications of metaheuristic algorithms. They have delved into different aspects like improving algorithm robustness, expanding its applicability to multi-objective problems, or integrating it with emerging technologies. Overall, it is evident that metaheuristic algorithms are evolving and finding increasing relevance in multiple domains, and a comprehensive exploration of their recent applications is essential for further advancements in related research areas.

Regardless of the type of MAs, Exploration and exploitation are critical characteristics of them for solving optimization problems effectively. Exploration refers to the ability of a search algorithm to find diverse solutions scattered across the search space, while exploitation emphasizes the process of intensifying the search in promising regions of the solution space to improve existing solutions or find better ones. Empirical evidence has shown that the exploration-exploitation balance is closely related to the convergence rate of a search method. While exploitation strategies can speed up convergence to the global optimum, they increase the likelihood of becoming trapped in local optima. Conversely, exploration strategies increase the chances of discovering regions with a higher likelihood of containing the global optimum within the search space, but at the cost of slower convergence [19].

The question of how to achieve exploration and exploitation of solutions in metaheuristic algorithms remains an open subject. In recent years, MAs have gained significant attention as they offer efficient solutions to various optimization problems. Among the available algorithms, the Dung Beetle Optimizer (DBO) [20] has emerged as a promising swarm intelligence optimization algorithm that is inspired by the behavior of dung beetles. The DBO algorithm employs five different update rules to find high-quality solutions and has been successfully applied to several practical engineering design problems.

The Dung Beetle Optimization (DBO) algorithm has shown to achieve better optimization performance than PSO, GWO, and other algorithms. However, due to the uneven population distribution on exploration and exploitation, the algorithm has a greater risk of becoming trapped in local optima and has limitations in convergence and search efficiency. Specifically, during the reproduction phase, if the local optimum coincides with the origin, the spawning process can cause an excessive concentration of dung beetles at this point, leading to stagnation and preventing further algorithmic progression. Additionally, the convergence factor in DBO does not effectively balance global exploration during the early stages with local exploitation in the later stages, resulting in suboptimal solution accuracy and hindering the algorithm's ability to fully exploit the solution space.

In response to these challenges, several improvement strategies have been proposed. First, dynamically adjusting the balance between foraging and spawning dung beetles, can facilitate better global search capabilities in the early stages and more effective local exploration in the later stages. Furthermore, modifications to the optimal spawning region can help the algorithm more effectively escape from local optima, thereby improving its overall robustness and performance.

These proposed enhancements to the DBO algorithm are in line with broader efforts to improve the efficacy of swarm intelligence algorithms. Recent advancements in metaheuristic algorithms have focused on improving exploration and exploitation capabilities, balancing computational efficiency, and overcoming limitations such as slow convergence rates and premature convergence to local optima. Recent advancements have focused on addressing these issues and enhancing the performance of existing algorithms.

For instance, the Modified EDO (MEDO) combines the Salp Swarm Algorithm (SSA) with Quadratic Interpolation (QI), significantly improving local search precision and preventing premature convergence through SSA's global migration mechanism [21]. Similarly, the Dynamic Hunting Algorithm (DHL) has been improved through the mDHL variant, which incorporates Levy Flight to enhance each hunter's tracking ability, along with a local escape operator and quasi-opposition learning to improve exploration and exploitation balance [22].

Additionally, the Worst Moth Disruption Strategy (WMFO) has been introduced to overcome issues of population stagnation and low diversity in the Moth Fly Optimization (MFO) algorithm, ensuring a more adaptive search process [23]. The AEFA-CSR (Artificial Electric Field Employing Cuckoo Search Algorithm with Refraction Learning) is another hybrid approach that enhances the Cuckoo Search (CS) method by improving global exploration while also using Refraction Learning to refine local exploitation [24].

These advancements reflect a growing trend towards developing adaptive and hybrid algorithms capable of handling a wider range of optimization tasks with improved accuracy and efficiency.

To achieve a better balance between exploration and exploitation, many researchers choose to improve the performance of their algorithms by introducing new strategies, combining existing algorithms, or proposing new algorithms altogether. Therefore, this paper proposes an enhanced DBO algorithm for global optimization and engineering applications called MSDBO. The proposed algorithm aims to further improve the convergence and search efficiency of DBO. MSDBO integrates multiple strategies of the DBO algorithm, including the dynamic population size variation, the forced ranking strategy, and the boundary control strategy. The proposed algorithm outperforms the original DBO algorithm in terms of optimization accuracy and search efficiency.

The main highlights and contributions of this paper are summarized as follows.

(1) An multi-strategy boosted DBO algorithm is developed by incorporating three strategies. Firstly, dynamic population size variation is introduced to balance exploration and exploitation during the optimization process. Secondly, the forced ranking strategy is introduced to expand the exploration ability of the optimal solution. Finally, a boundary control strategy is employed to fully utilize the information of excellent individuals in the population to accelerate the convergence. These modifications aim to strengthen the exploration and exploitation capabilities of the DBO algorithm, potentially leading to better optimization performance.
(2) The convergence of the proposed MSDBO algorithm is evaluated by comparing it with other advanced algorithms using the CEC 2014 and CEC 2019 test suites [25, 26]. The CEC test suites are widely recognized as standard benchmarks for evaluating optimization algorithms, and include a set of challenging optimization problems with various characteristics. The experimental results indicate that the proposed algorithm performs competitively on most of the test functions compared to other advanced algorithms.
(3) The ability of the proposed MSDBO algorithm to solve practical engineering optimization problems is confirmed by applying it to four engineering optimization examples. The optimization objectives and constraints of each example are carefully defined to reflect the real-world engineering scenarios. The results show that the proposed algorithm can generate satisfactory solutions for these problems and performs comparably or better than other advanced algorithms in terms of solution accuracy and convergence speed. This suggests that the algorithm has potential for application in various engineering domains.

The reminder of this paper is organized as follows. In Sect. 2, a brief description of MAs and the DBO algorithm are provided to given a foundation for understanding the proposed modifications. Section 3 introduces the three strategies that are used to enhance the DBO algorithm and presents the proposed MSDBO algorithm in detail. In Sect. 4, the performance of the MSDBO algorithm is compared with other advanced algorithms on commonly used benchmark test suites (CEC 2014 and CEC 2019). Section 5 focuses on the applications of the MSDBO algorithm to several constrained optimization problems in CEC 2011 and four practical engineering optimization problems. Section 6 applies MSDBO and the competing algorithms to the Parameters identification of photovoltaic models. Finally, Sect. 7 concludes this paper with some remarks for future research work.

2 Literature review

2.1 Metaheuristic algorithms

In recent decades, metaheuristic algorithms have garnered widespread attention across various scientific disciplines, including computer science, engineering, and mathematics [27]. These algorithms, designed to optimize complex problems, are often inspired by a wide range of natural and

conceptual phenomena, such as evolution, collective behavior, random search processes, and more. Metaheuristics are typically classified into four main categories based on their underlying principles: Evolutionary-based algorithms, Swarm-based algorithms, Physics-based algorithms, and Human-based algorithms. Each category reflects distinct sources of inspiration-ranging from the behavior of living organisms and genetic processes to the laws of physics and human activities-providing diverse approaches to solving optimization challenges.

Evolutionary-based algorithms are inspired by concepts from biology, particularly genetics and evolutionary processes, including natural selection, genetic variation, and inheritance.Key algorithms in this category include the Genetic Algorithm (GA) [28] and Differential Evolution (DE) [29], both of which replicate natural reproduction processes and incorporate genetic operations like mutation, crossover, and selection.

Swarm-based algorithms are inspired by the collective behaviors of various organisms, including birds, insects, fish, and other animals. Notable examples include Particle Swarm Optimization (PSO) [30], Artificial Bee Colony (ABC) [31], and Whale Optimization Algorithm (WOA) [32]. PSO mimics the coordinated movements of bird flocks and fish schools, while ABC is based on the hierarchical foraging behaviors of bees, and WOA is inspired by the social behaviors of humpback whales. Additionally, swarm-based algorithms are often influenced by natural behaviors such as hunting, migration, flight strategies, and foraging. Other algorithms in this category include the Fossa Optimization Algorithm (FOA) [33], Addax Optimization Algorithm (AOA) [34], Remora Optimization Algorithm (ROA) [35], Harris Hawks Optimization (HHO) [36], Slap Swarm Algorithm (SSA) [37], Reptile Search Algorithm (RSA) [38], Dragonfly Algorithm (DA) [39], Coati Optimization Algorithm (COA) [40], and Dung Beetle Optimizer (DBO) [20], among others, each drawing on specific animal behaviors for optimization purposes.

Physics-based algorithms are grounded in physical principles, including laws, forces, and phenomena. One of the most prominent examples is Simulated Annealing (SA) [41], which is inspired by the annealing process in metallurgy. Other algorithms, such as the Momentum Search Algorithm (MSA) [42] and Spring Search Algorithm (SSA) [43], draw on physical forces and Newton's laws of motion. Additionally, algorithms like the Gravitational Search Algorithm (GSA) [44], Special Relativity Search (SRS) [45], and Water Cycle Algorithm (WCA) [46] are based on various physical concepts.

Human-based algorithms are inspired by cognitive processes, behaviors, and decision-making patterns observed in humans. These algorithms often model human activities such as learning, memory, and personal development. For example, Teaching-Learning Based Optimization (TLBO) [47] is based on educational processes, where solutions represent learners, and optimization occurs through interactions between teachers and students. The Growth Optimizer (GO) [48] is inspired by the learning and reflective processes individuals undergo during their growth and development within society. Additionally, algorithms like Potter Optimization Algorithm (POA) [49], Carpet Weaving Optimization (CWO) [50], Dollmaker Optimization Algorithm (DOA) [51],Poor and Rich Optimization algorithm (PRO) [52], and Sculptor Optimization Algorithm (SOA) [53] draw on various aspects of human cognition and social interactions to guide the search for optimal solutions.

2.2 Dung beetle optimizer

The Dung Beetle Optimizer (DBO) algorithm is a novel metaheuristic algorithm inspired by the behavior of dung beetles. The algorithm divides the population into four sub-populations, each performing a specific activity: ball rolling, reproduction, foraging, and thievery, as shown in Fig. 1. The exploration of these subpopulations is used to update the position and ultimately seek the optimal solution. In the DBO algorithm, assuming a population size of N, the position of each dung beetle in the D-dimensional search space is represented by Eq. (1).

$$\boldsymbol{X} = \begin{vmatrix} x_{1,1} & x_{1,2} & \cdots & x_{1,j} & \cdots & x_{1,D} \\ x_{2,1} & x_{2,2} & \cdots & x_{2,j} & \cdots & x_{2,D} \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ x_{i,1} & x_{i,2} & \cdots & x_{i,j} & \cdots & x_{i,D} \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ x_{N,1} & x_{N,2} & \cdots & x_{N,j} & \cdots & x_{N,D} \end{vmatrix} \tag{1}$$

where $x_{i,j}$ denotes the position of the i^{th} dung beetle in the j^{th} dimension. The fitness of each dung beetle is calculated based on its position information, as shown in Eq. (2).

$$\vec{fitness} = \begin{pmatrix} f(\vec{x}_1) \\ f(\vec{x}_2) \\ \vdots \\ f(\vec{x}_i) \\ \vdots \\ f(\vec{x}_N) \end{pmatrix} \tag{2}$$

where $\vec{x}_i = [x_{i,1}, x_{i,2}, \ldots, x_{i,D}]$ and $f(\vec{x}_i)$ is the fitness value of the i_{th} dung beetle. The four subpopulations in the DBO algorithm are crucial for the optimization process, as they enable the algorithm to balance exploration and exploitation and enhance the equilibrium search capability.

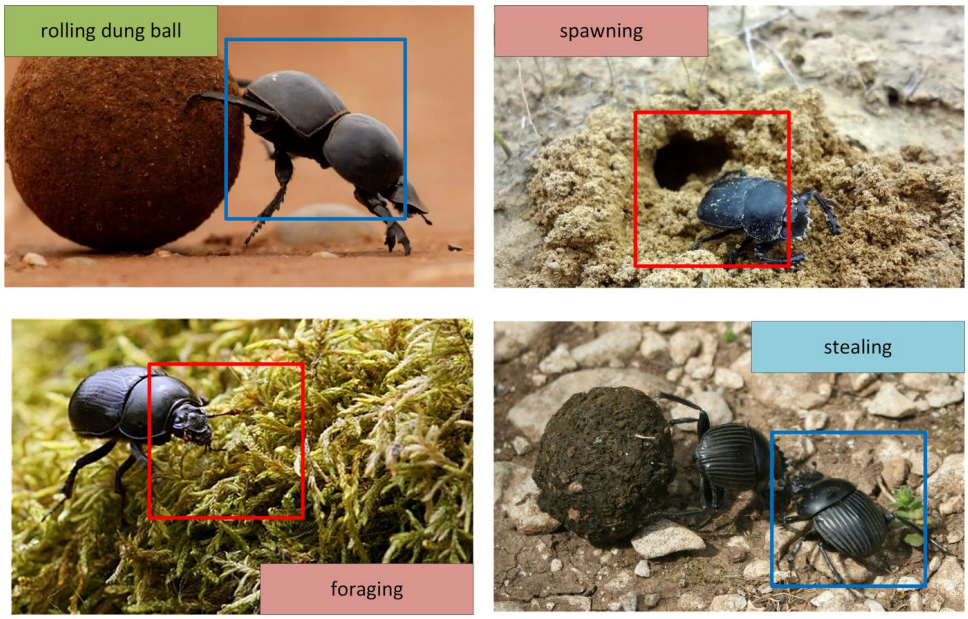


Fig. 1 Four behaviors of dung beetles

The primary objective of ball rolling dung beetles is to transport their fecal balls quickly and efficiently, thus avoiding competition with other dung beetles [54]. During the ball rolling process, the position of the dung beetles rolling the ball is continuously updated. This process can be represented by Eq. (3).

$$\begin{aligned} &\vec{x}_i\left(t+1\right)=\vec{x}_i\left(t\right)+\alpha\times k\times\vec{x}_i\left(t-1\right)+b\times\Delta\vec{x},\\ &\Delta\vec{x}=\left|\vec{x}_i\left(t\right)-\vec{x}_{worst}\right|, \end{aligned} \tag{3}$$

where t representsthe number of current iterations, and $\vec{x}_i\left(t\right)$ represents the position of the ith dung beetle at the tth iteration. The symbol $k\in(0,0.2]$ represents a constant parameter for the deflection coefficient, and b is a constant parameter belonging to (0,1). α is a natural coefficient with a value of -1 or 1 is set to be 1 or -1 by the probability method, which is used to simulate the changes in light intensity, and x_{worst} represents the global worst position.

When a dung beetle encounters an obstacle in its path, it must adjust its direction and find a new route to continue rolling its fecal ball. In the DBO algorithm, the dung beetle uses the tangent function to calculate a new rolling direction, which it mimics through a dance-like behavior [55].

This dance allows the beetle to adjust its position and orientation, enabling it to navigate around obstacles and continue rolling the ball towards its destination. Under these circumstances, the position of the ball rolling dung beetle is updated according to the Eq. (4).

$$\vec{x}_i\left(t+1\right)=\vec{x}_i\left(t\right)+\tan\left(\theta\right)\times\left|\vec{x}_i\left(t\right)-\vec{x}_i\left(t-1\right)\right| \tag{4}$$

where θ is belonging to $[0,\pi]$ representing the deflection angle, and it should be noted that the position of dung beetle is not be updated when $\theta=0,\frac{\pi}{2},or\ \pi$.

Choosing a suitable location to lay eggs is crucial for dung beetles to ensure the safety and survival of their offspring. Female beetles must carefully select a location that provides adequate protection and resources to support the growth and development of their young. To simulate this behavior, the DBO algorithm includes a boundary selection strategy that defines the breeding areas for female beetles, which is defined by Eq. (5).

$$\begin{aligned} \vec{Lb}^{*}&=\max\left(\vec{x}_{lbest}\times\left(1-R\right),\vec{Lb}\right),\\ \vec{Ub}^{*}&=\min\left(\vec{x}_{lbest}\times\left(1+R\right),\vec{Ub}\right), \end{aligned} \tag{5}$$

where $\vec{x}_{lbest}$ represents the current local optimal position, $\vec{Lb}^{*}$ and $\vec{Ub}^{*}$ indicate the lower and upper bounds of the spawning area, respectively. $R=1-t/maxiter$ and *maxiter* indicates the maximum number of iterations, and $\vec{Lb}$ and $\vec{Ub}$ represent the lower and upper bounds of the optimization problem, respectively.

During the iteration process of the DBO algorithm, the position of the brood ball undergoes dynamic changes. These changes are determined by the Eq. (6).

$$\vec{x}_i\left(t+1\right)=\vec{x}_{lbest}+\vec{b}_1\times\left(\vec{x}_i\left(t\right)-\vec{Lb}^{*}\right)+\vec{b}_2\times\left(\vec{x}_i\left(t\right)-\vec{Ub}^{*}\right) \tag{6}$$

where $\vec{x}_i\left(t\right)$ represents the position of the ith brood ball at the tth iteration. $\vec{b}_1$ and $\vec{b}_2$ are two independent random vectors of size $1\times D$, where D is the dimension of the optimization problem.

In addition to simulating the behavior of ball rolling and reproduction dung beetles, the DBO algorithm also incorporates the foraging process of small dung beetles in nature. By mimicking the behavior of small dung beetles as they search for food, the DBO algorithm is able to more accurately simulate the foraging behavior of dung beetles and optimize solutions based on their behavior.

To define the optimal foraging area for small dung beetles, the DBO algorithm uses the following boundary definition.

$$\begin{aligned} \vec{Lb}^{b} &= \max\left(\vec{x}_{gbest} \times (1-R), \vec{Lb}\right), \\ \vec{Ub}^{b} &= \min\left(\vec{x}_{gbest} \times (1+R), \vec{Ub}\right), \end{aligned} \tag{7}$$

where $\vec{x}_{gbest}$ represents the global optimal position, and $\vec{Lb}^{b}$ and $\vec{Ub}^{b}$ represent the lower and upper bounds of the optimal foraging area, respectively. Therefore, the position update of the small dung beetles is shown as Eq. (8).

$$\vec{x}_i(t+1) = \vec{x}_i(t) + C_1 \times \left(\vec{x}_i(t) - \vec{Lb}^{b}\right) + C_2 \times \left(\vec{x}_i(t) - \vec{Ub}^{b}\right) \tag{8}$$

where $\vec{x}_i(t)$ represents the position information of the ith small dung beetles in the tth iteration, while C_1 represents a random number that follows the normal distribution, and C_2 represents a random vector that belongs to (0,1).

Some dung beetles are known to engage in thieving behavior, in which they steal faeces from others. The DBO algorithm incorporates this behavior by assuming that the location for thieving dung beetles to compete is the global best position, represented by $\vec{x}_{gbest}$. During the iteration process, the position of the thieving dung beetles is updated according to the Eq. (9).

$$\vec{x}_i(t+1) = \vec{x}_{gbest} + S \times \vec{g} \times \left(|\vec{x}_i(t) - \vec{x}_{lbest}| + |\vec{x}_i(t) - \vec{x}_{gbest}|\right) \tag{9}$$

where $\vec{x}_i(t)$ represents the location information of the ith thief at the tth iteration, and $\vec{g}$ is a size $1 \times D$ random vector following a normal distribution, and S is a constant value, defined as 1 in the original algorithm.

3 MSDBO

Although the Dung Beetle Optimizer (DBO) has demonstrated superior performance in optimizing problems, achieving the optimal solution remains highly challenging for this algorithm. Furthermore, its ability to solve complex problems is not entirely satisfactory. In response to these limitations, the MSDBO has been developed, incorporating three novel strategies to enhance the DBO algorithm's exploration and exploitation capabilities, promote population diversity, and accelerate convergence. The proposed strategies include dynamic population size variation, forced ranking strategy, and a novel boundary control strategy, which are used to enhance the performance of the DBO algorithm in solving various optimization problems. The details of proposed MSDBO algorithm are shown in this section.

3.1 Dynamic population size variation

In the Dung Beetle Optimizer (DBO), the size of the four subpopulations is fixed as the number of iterations increases, which can result in an imbalance between exploration and exploitation during the iteration process. To overcome this limitation and improve the overall efficiency of the DBO algorithm, a novel approach to dynamically adjust the size of the sub-populations was designed.

The dynamic population size variation strategy aims to balance the exploration and exploitation capabilities of the algorithm by dynamically adjusting the population size based on a new population division formula, as shown in Eq. (10). It aims to enhance the exploration and exploitation capabilities of the population search in both the early and late stages of the iteration process.

$$\begin{aligned} Pnum_1 &= p_1 \times (N - (k - (\exp(-t/\max iter))) \times N), \\ Pnum_2 &= (1-p_1) \times (N - (k - (\exp(-t/\max iter))) \times N), \\ Pnum_3 &= p_2 \times ((k - (\exp(-t/\max iter))) \times N), \\ Pnum_4 &= (1-p_2) \times ((k - (\exp(-t/\max iter))) \times N), \end{aligned} \tag{10}$$

where $Pnum_1$, $Pnum_2$, $Pnum_3$ and $Pnum_1$ represents the subpopulation size of ball rolling, reproduction, forage and thievery dung beetles, where N indicates the population size of the dung beetles. k denotes a constant between 1 and 2, and t represents the current iteration.

Specifically, the proposed strategy adjusts the size of four subpopulations to enhance the algorithm's performance. The proposed strategy gradually increases the size of ball rolling and reproduction dung beetles, which represent the exploration subpopulations, as the size of iterations increases. This increase in the exploration subpopulations enables the algorithm to explore new regions in the search space and avoid premature convergence. Conversely, the proposed strategy reduces the size of the forage and thievery subpopulations, which represent the exploitation subpopulations, as the number of iterations increases.

This reduction in the exploitation subpopulations allows the algorithm to focus on exploiting promising solutions and converging towards the optimal solution. It is worth noting that the proposed strategy dynamically adjusts the subpopulation size to balance global exploration and local exploitation at different stages of the search process. By doing so, the algorithm can efficiently explore the search

space in the early stages and exploit promising solutions in the later stages, which can lead to improved performance in solving a range of optimization problems.

3.2 Forced ranking strategy

Evolutionary theory, particularly the principle of natural selection, has inspired the development of numerous optimization algorithms aimed at improving solution quality. One of the key concepts derived from natural selection is the survival of the fittest, which can be effectively translated into a forced ranking strategy. This strategy enhances population optimization by increasing the likelihood of discovering superior solutions. The forced ranking strategy proposed in this paper can be integrated into various optimization algorithms, serving as a mechanism to improve performance.

The forced ranking strategy operates by selecting individuals with poorer fitness after each iteration and replacing them with new individuals generated using the formula in Eq. (11). This mechanism ensures that the population evolves towards better solutions with each iteration, improving search efficiency and overall solution quality.

$$\vec{x}_i\left(t+1\right)=\vec{x}_{elite}+r_1\times R\times\left(\vec{Ub}-\vec{Lb}\right) \tag{11}$$

where x_{elite} indicates elite individuals with randomly selected fitness values in the top p percent of the population. *Ub* and *Lb* denote the upper and lower bounds of the search space, respectively. r_1 denotes a random number between 0 and 1, and $R=1-t/maxiter$ represents the dynamic radius, where *t* represents the current iteration, and *maxiter* denotes the maximum number of iterations.

As depicted in Fig. 2, the proposed forced ranking strategy works by ordering the population based on fitness. This approach enhances the algorithm's exploration ability and mitigates premature convergence by selecting less adapted individuals for elimination and replacing them with new candidates generated according to a dynamic evolution formula. This process is repeated iteratively, ensuring that the population gradually converges towards high-quality solutions. The integration of this strategy improves population diversity and encourages the discovery of new high-quality solutions, making it a simple yet effective addition to optimization algorithms. Moreover, it can be easily customized and adapted to different optimization problems, further enhancing its versatility.

Specifically, this approach accelerates convergence to the global optimum by producing high-quality individuals that are passed down to the next generation. As a result, the fitness level of the entire population gradually improves over time. Additionally, by selecting higher-ranked individuals to update and replace lower-ranked ones, the strategy increases population diversity, making it more likely to find the global optimum rather than local optima.

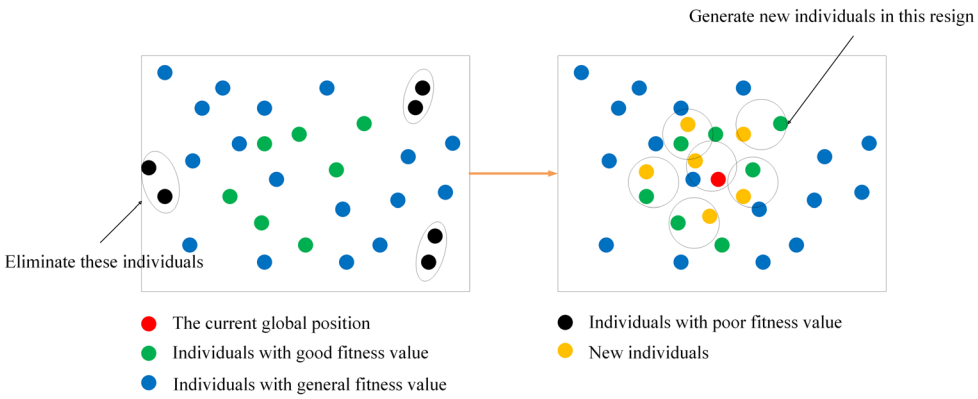
It is important to note that the selection of the top *p* percent of individuals should be determined based on the specific characteristics of the problem and algorithm. Furthermore, Eq. (11) generates new individuals using a dynamic search radius, which changes based on the current iteration number. This elastic search range mechanism ensures large-scale exploration during the early stages of optimization and smaller-scale exploitation in the later stages, thus achieving a balanced exploration-exploitation trade-off throughout the search process.

3.3 Boundary control strategy

In the iterative process of optimization algorithms, individuals may exceed the search boundary, which can negatively impact the search process and solution quality. Existing approaches to address this issue typically involve assigning fixed upper or lower limit values to individuals outside the boundary [56], or performing delayed estimation based on the boundary values. However, these methods do not always effectively utilize population information.

To overcome this limitation, a novel boundary control strategy based on the position information of the elite individuals is proposed. This approach leverages the information

Fig. 2 Forced ranking strategy



from the current iteration population to adjust the position of individuals outside the boundary, rather than relying on fixed limit values or delayed estimation. The proposed method is formulated as Eq. (12), which randomly places individuals outside the boundary nearby individuals with better fitness. In the next iteration, they will be searched in positions that are better than their original positions. Therefore, it can improve the exploitation ability of DBO.

$$\vec{x}_{i,j}(t+1) = \vec{x}_{elite,j} + r_2 \times (\vec{x}_{elite,j} - \vec{x}_{gbest,j}) \tag{12}$$

where $\vec{x}_{gbest}$ and $\vec{x}_{elite}$ represent the global optimal solution and elite individuals with randomly selected fitness values in The top p percent of the population of the *i*th iteration, respectively, and r_2 is a random number in the range [0, 1].

The proposed boundary control strategy represents a significant improvement over existing approaches, as it effectively leverages population information to adjust the position of individuals outside the boundary. By considering the position of both the global optimal individual and the elite individuals, the approach provides a more accurate and comprehensive representation of the population, which ultimately leads to improved solution quality and search efficiency. The pseudocode of the boundary control strategy is shown in **Algorithm** 1.

Input: $x_i(t)$, Lb, Ub, X^b, X^e
Output: $x_i(t)$
for $j \leftarrow 1$ *to* D **do**
  if $x_{i,j}(t) \leqslant Lb$ *or* $x_{i,j}(t) \geqslant Ub$ **then**
    Update $x_{i,j}(t)$ by using Eq.(12);
for $j \leftarrow 1$ *to* D **do**
  if $x_{i,j}(t) \leqslant Lb$ **then**
    $x_{i,j} = Lb$;
  if $x_{i,j}(t) \geqslant Ub$ **then**
    $x_{i,j} = Ub$;

Algorithm 1 Boundary control strategy

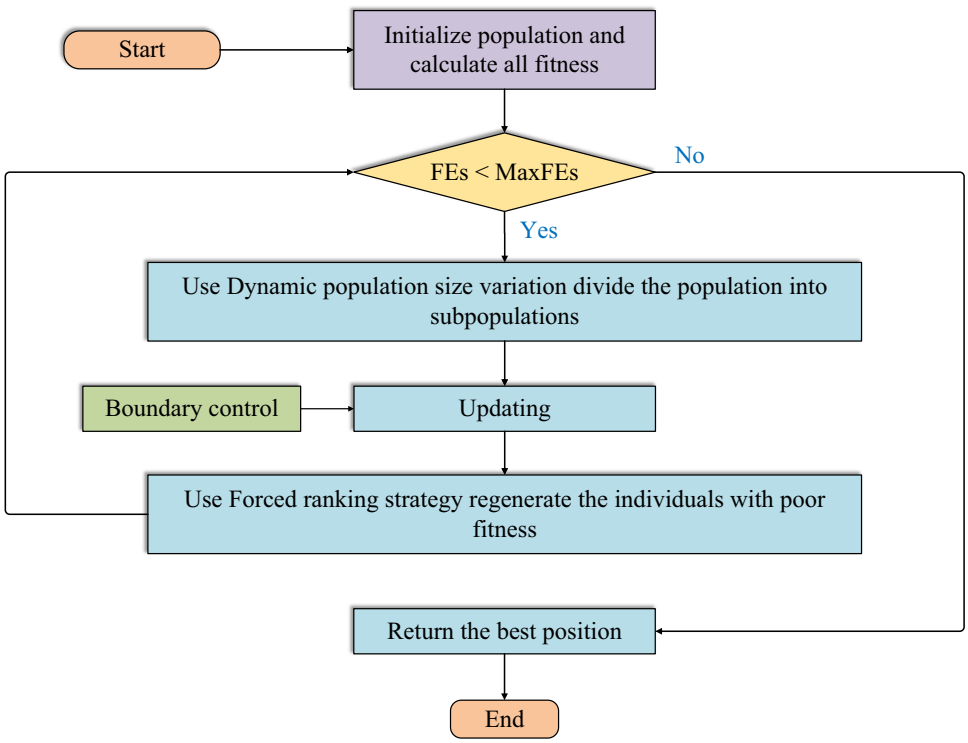


Fig. 3 Flowchart of MSDBO

By integrating the proposed strategies into DBO, a new algorithm called MSDBO has been developed, which delivers a considerable improvement over the original algorithm. MSDBO merges the strengths of the DBO with the benefits of the suggested strategies, providing a more resilient and efficient approach for solving optimization problems. The pseudo code of MSDBO is presented in **Algorithm** 2, and the flowchart of MSDBO can be seen in Fig. 3.

3.4 Complexity analysis

In this study, the complexity analysis of MSDBO primarily focuses on the time required for operations performed within each generation. Let the population size and problem dimension be denoted by N and D, respectively. Initially, the cost of initializing and evaluating candidate solutions is $O(N \times D)$. Following this, the cost associated with the execution of operators within the loop is $O((N-1) \times D)$. Additionally, sorting the fitness values utilizes the quicksort algorithm, which has a time complexity of $O(N \times \log N)$. Consequently, the overall complexity of MSDBO can be expressed as $O(N \times \log N) + O((N-1) \times D)$.

Input: The maximum iterations $maxiter$, the size of the particle's population N
Output: Optimal position x_{gbest} and its fitness value $gbestfitness$
Initialization the particle's population $i \leftarrow 1, 2, ..., N$ and define its relevant paramaters;
while $t \leqslant maxiter$ **do**
  for $i \leftarrow 1$ *to* N **do**
    if $i <= Pnum_1$ **then**
      $\delta = rand\,(1)$;
      if $\delta < 0.9$ **then**
        $\eta = rand\,(1)$;
        if $\eta > \lambda$ **then**
          $\alpha = 1$;
        else
          $\alpha = -1$;
        Update the position by using Eq. (3);
      else
        if $\theta = 0$ *or* $\theta = \pi/2$ *or* $\theta = \pi$ **then**
          x_i is not updated;
        else
          Update the position by using Eq. (4);
    if $i > Pnum_1$ *and* $i <= Pnum_1 + Pnum_2$ **then**
      for $i \leftarrow 1to\ n$ **do**
        Update the position by using Eq. (6);
    if $i > Pnum_1 + Pnum_2$ *and* $i <= Pnum_1 + Pnum_2 + Pnum_3$ **then**
      Update the position by using Eq. (8);
    if $i > Pnum_1 + Pnum_2 + Pnum_3$ **then**
      Update the position of thief by using Eq. (9);
    if $x_{i,j}\,(t) \leqslant Lb$ *or* $x_{i,j}\,(t) \geqslant Ub$ **then**
      Update the position using **Algorithm 1**;
  if $f(x_i\,(t)) < fitness\,(i)$ **then**
    Update it;
  if $x_i\,(t) = x_{worst}$ **then**
    Update the position of the dung beetle by using Eq. (11);
  t=t+1;
return

Algorithm 2 The procedure of MSDBO

4 Experimental studies

To validate the effectiveness of the proposed MSDBO, multiple experiments and comprehensive comparisons with other advanced optimization algorithms were conducted. Specifically, the performance of the proposed MSDBO algorithm was assessed on two sets of benchmark functions: 30 functions from the CEC 2014 and 10 competition functions from the CEC 2019. These benchmark functions represent a diverse range of optimization problems with different characteristics, enabling us to evaluate the capabilities of MSDBO across a variety of problems.The performance of MSDBO was compared with several other popular optimization algorithms, including GA, WOA, HHO, GWO, RSA, SSA, and DBO. All comparison algorithms were implemented with their default setting parameters. The information and parameter settings of these algorithms are detailed in Table 1. The mean and standard deviation of the fitness values were calculated for each algorithm, and a non-parametric statistical test, the Friedman sorting test, was used to test for significant differences between the results of MSDBO and the other algorithms at a 5% significance level. This comprehensive analysis enables us to evaluate the performance of MSDBO relative to other advanced optimization algorithms and demonstrate its effectiveness in solving complex optimization problems.

The experiments and analysis were conducted on a computer system with a Core i5 CPU running on Windows 11 (64-bit). The hardware and software used in the experiments

Table 1 The algorithm information and parameter settings

Algorithm	Time	Parameters srttings
GA	1975	$pc = 0.8, pm = 0.2$
WOA	2016	$b = 1a_1 = 2 - (2 \times FEs/MaxFEs)$, $a_2 = -1 - (FEs/MaxFEs)$
HHO	2019	$E_0 = 2 \times rand - 1$, $E_1 = 2 - 2 \times (FEs/MaxFEs)$
GWO	2014	$\alpha = 2 - (2 \times FEs/MaxFEs)$
RSA	2022	$\alpha = 0.1, \beta = 0.005$
SSA	2017	$c_1 = 2 \times \exp\left((4 \times FEs/MaxFEs)^2\right)$
PRO	2019	$P_{mut} = 0.06$
DA	2020	$r = (ub - lb)/10$, $Delta_{max} = (ub - lb)/10$
SRS	2023	$\mu = 1, m = 1$
COA	2023	$iguana = lb + rand * (ub - lb)$
DBO	2022	$k = \lambda = 0.1, b = 0.3, S = 0.5$
MSDBO	2024	$p = 0.2, p_1 = 0.5, p_3 = 0.6$

were carefully selected to ensure reliable and consistent results, allowing us to draw meaningful conclusions about the performance of MSDBO.

4.1 Results on CEC 2014 benchmark functions

Table 1 presents the results of the experiments, which compare the performance of MSDBO with several other advanced optimization algorithms on CEC 2014. The experiments were conducted 30 times with a population size of 10 and 100000 function evaluations. The table shows the average and standard deviation of the fitness function values for each algorithm after 30 runs. In most cases, MSDBO achieved better average fitness function values than the original DBO algorithm and other comparison algorithms. Moreover, in 20 out of 30 cases, MSDBO achieved better convergence results than DBO.

The results for functions F1-F30 indicate that MSDBO performs competitively against most of the compared algorithms. This is attributed to its ability to reduce the likelihood of falling into local optima and to balance exploration and exploitation effectively. For function F14, the MSDBO algorithm obtains second rank after SSA, but outperforms other six optimization algorithms (including GA, WOA, HHO, GWO, RSA, and DBO algorithms). For functions F18, the developed MSDBO algorithm achieves third rank after HHO and SSA. Furthermore, in functions F21-F30, MSDBO demonstrates competitive performance compared to several well-known optimizers.

The three qualitative metrics of search history, diversity history, and trajectory in Fig. 4 are used to analyze the performance of MSDBO in optimizing different types of functions, which are $F1$, $F6$, $F10$, $F16$, $F24$ and $F28$. The first column illustrates the 3D location distribution of all solutions across each benchmark function, providing insights into the domain's topology where MSDBO conducts its search for the optimal solution.

The second column presents the search history of MSDBO. It is evident that some historical optimal solutions are scattered throughout the solution space, while the majority are concentrated around the global optimum. This suggests that MSDBO can quickly approximate the optimal solution and transition to the exploitation phase, contributing to improved solution accuracy.

The third column showcases the diversity history of MSDBO during its optimization process. The fluctuations in diversity highlight the algorithm's ability to balance exploration and exploitation. At the early stages, a relatively higher diversity is maintained, facilitating the exploration of a broader solution space. As the optimization progresses, the diversity decreases, reflecting a shift toward exploiting promising regions.

In the fourth column, the trajectory of MSDBO's search on the CEC 2014 benchmark functions is depicted. The visualization shows how the search agents evolve across iterations. Initially, the agents explore the solution space broadly, and as the search progresses, the trajectories converge toward the global optimum. This behavior reflects the algorithm's ability to adjust the search process and reach optimal solutions efficiently, consistent with MSDBO's design that balances exploration and exploitation.

Figure 5 shows the convergence curves of MSDBO and the comparison algorithms, indicating that MSDBO converges faster than DBO and other algorithms. These results suggest that the modifications in MSDBO have improved its performance and addressed some limitations of the original DBO.

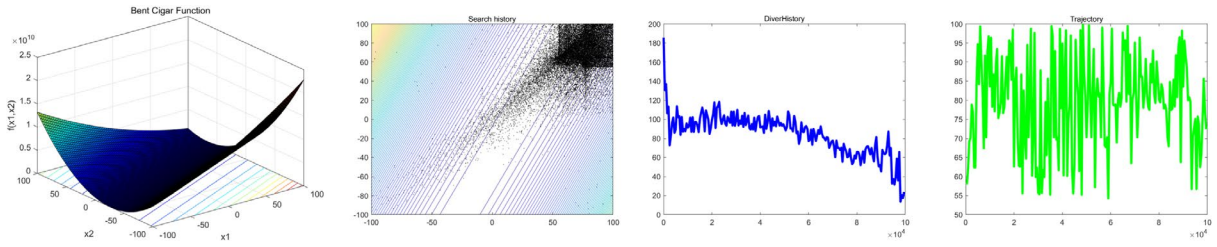
In addition to performing better than the standard DBO algorithm, MSDBO also achieved competitive results compared to other optimization algorithms in many cases, highlighting its potential for searching optimal solutions. These results demonstrate that MSDBO can be a valuable tool for solving optimization problems in various engineering applications.

To further support the conclusions drawn from the experimental results, statistical methods such as the Wilcoxon rank sum test and the Friedman test were used to assess the variability of the algorithms. Table 2 presents the statistical results, including the final ranking of the algorithms according to the Friedman sorting test. The indicator 'w/t/l' in Table 2 is obtained based on the Wilcoxon signed-rank test, and it indicates the number of test problems in which the MSDBO performed better, similar, or worse compared to competing algorithms in the condition of $\alpha = 0.05$. The results show that MSDBO ranked first, indicating that the modifications made to the standard DBO algorithm in this work are capable of enhancing its performance and overcoming its limitations.

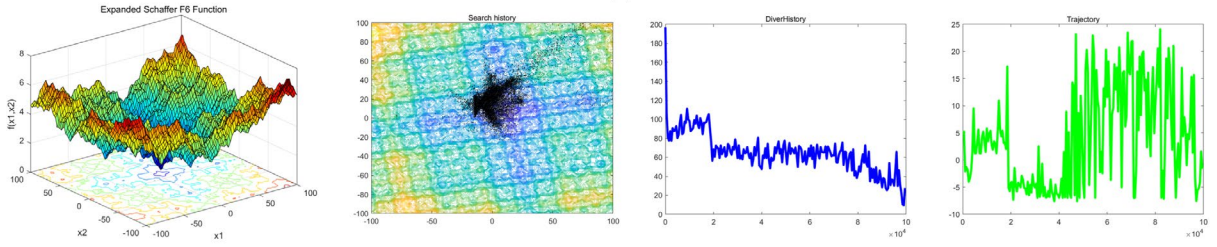
The F-Rank gives the comprehensive Friedman rank. The smaller the value, the higher the ranking. MSDBO ranks at the top with an F-Rank of 3.82, indicating its excellent comprehensive performance on the overall set of test functions. RSA ranks lower with an F-Rank of 9.11, reflecting its relatively poor overall performance. The rankings of other algorithms also correspondingly reflect their positions in the comprehensive performance of this series of tests. Overall, in the CEC 2014 benchmark tests, MSDBO demonstrates strong comprehensive performance, with notable results in terms of mean values, stability, and overall ranking.

4.2 Results on CEC 2019 benchmarks functions

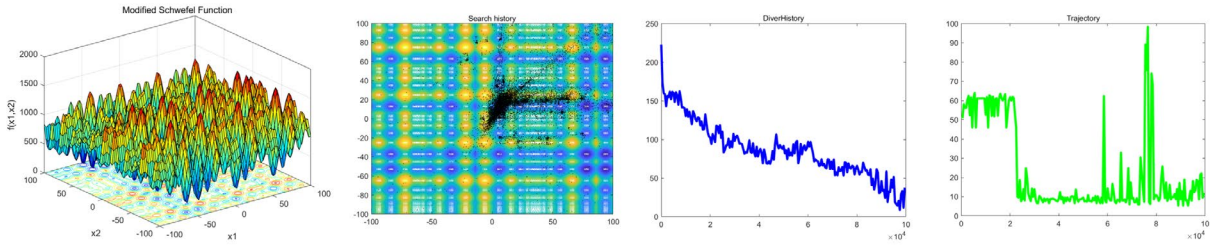
In this section, the performance of the proposed MSDBO algorithm is evaluated on the CEC 2019 benchmark suite,



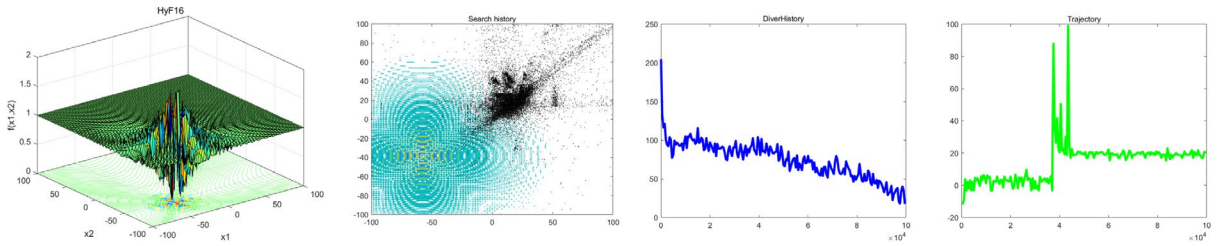
(a) F1



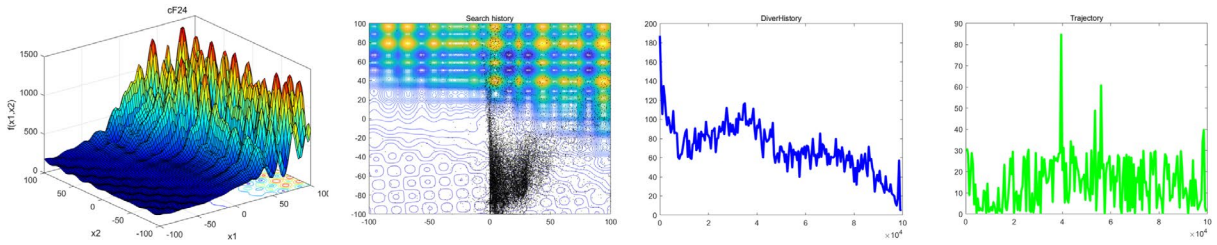
(b) F6



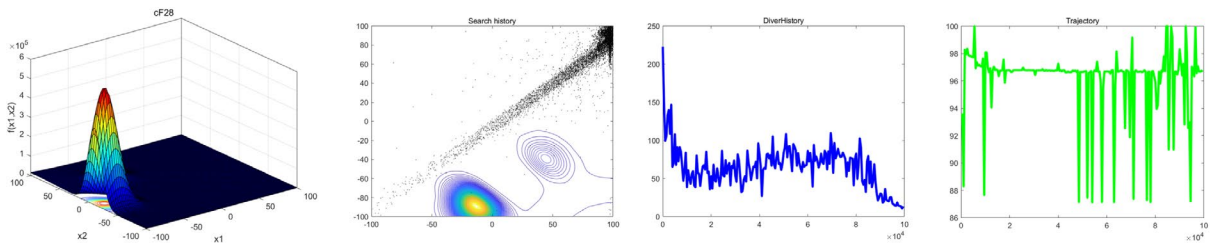
(c) F10



(d) F16



(e) F24



(f) F28

Fig. 4 Qualitative metrics: search history, diversity history, and trajectory of CEC 2014

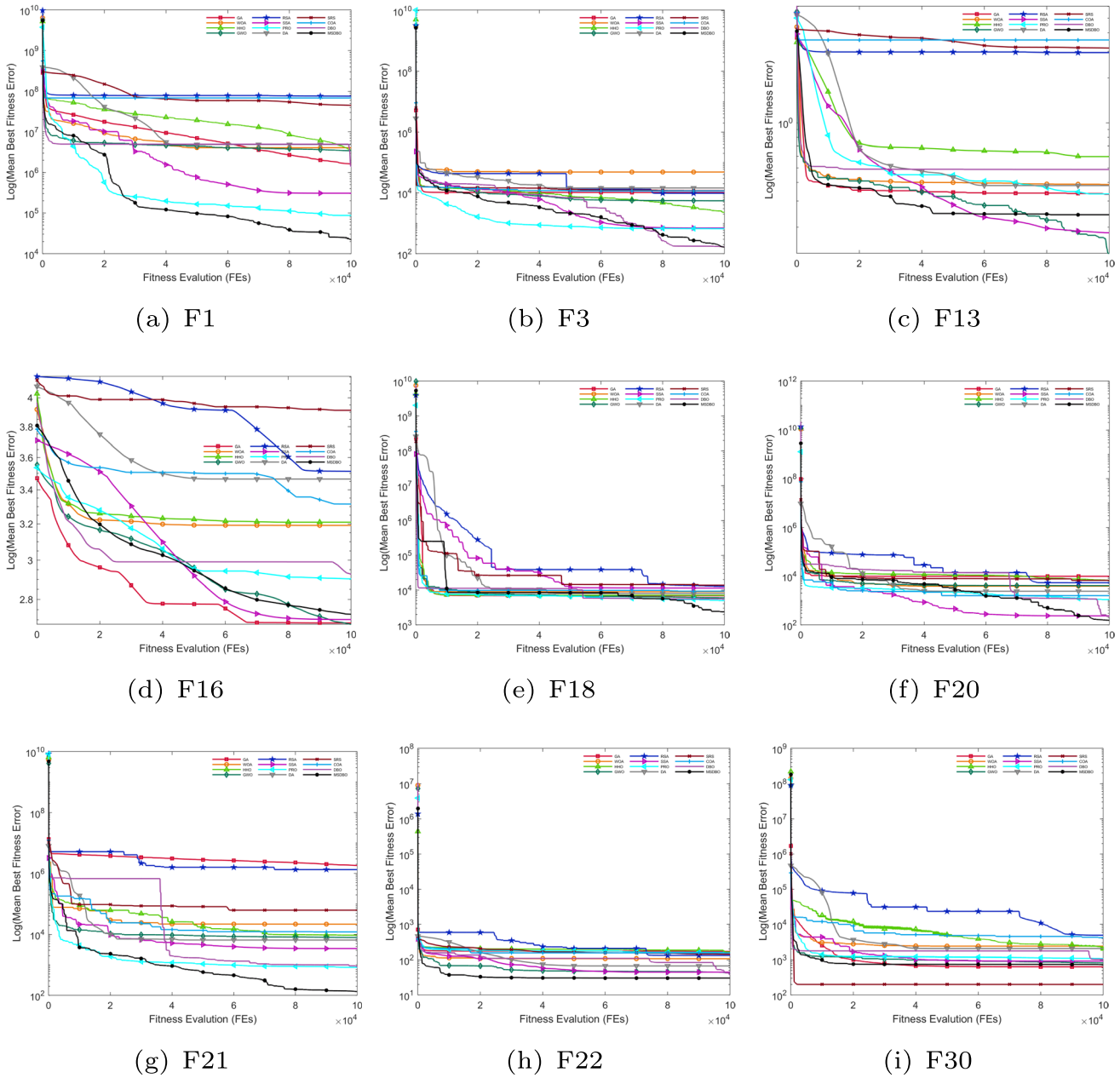


Fig. 5 Convergence curves of different algorithms obtained on the CEC 2014 benchmarks with 10 dimensions

and the results are compared with several well-known optimization algorithms. The test suite consists of 10 problems and was proposed as part of the Congress on Evolutionary Computation (CEC) 2019 100-Digit Challenge Special Conference and Single Objective Numerical Optimization Competition. This competition aimed to promote the development of new optimization algorithms and variations, and the test suite was designed to challenge the limits of optimization algorithms in terms of scalability and accuracy. The details of the test suite and the experimental settings required for the competition are described in [26]. The function graphs of the CEC 2019 test suite are shown in the Fig. 8.

During the testing process, the optimization ability and stability of the MSDBO algorithm were evaluated based on the average and standard deviation of the experimental results. The average ranking was calculated using the results from 30 independent experiments, as presented in Table 3. The results indicate that MSDBO performed competitively across all ten optimization problems, ranking among the top-performing algorithms.

The comprehensive ranking provided by F-Rank (Friedman rank) reflects the overall performance of each

Table 2 Results of CEC 2014 benchmark test functions

Function	Item	GA	WOA	HHO	GWO	RSA	SSA	PRO	DA	SRS	COA	DBO	MSDBO
F1	mean	1.62E+06	4.10E+06	3.64E+06	3.46E+06	7.58E+07	3.08E+05	8.67E+04	4.84E+06	4.50E+07	6.83E+07	1.41E+06	**2.21E+04**
	std	9.59E+05	3.29E+06	2.60E+06	3.57E+06	4.95E+07	2.27E+05	5.75E+04	5.43E+06	4.04E+07	1.59E+07	1.60E+06	**1.84E+04**
	p-Value	7.94E-03	7.94E-03	7.94E-03	7.94E-03	7.94E-03	7.94E-03	3.17E-02	7.94E-03	7.94E-03	7.94E-03	9.52E-02	1.00E+00
	h	+	+	+	+	+	+	+	+	+	+	≈	≈
F2	mean	4.54E+03	6.81E+04	3.14E+05	6.25E+03	3.34E+09	1.03E+03	6.66E+03	3.41E+06	7.70E+09	5.09E+09	6.50E+03	1.38E+03
	std	4.98E+03	3.93E+04	9.08E+04	5.17E+03	1.81E+09	1.77E+03	3.97E+03	1.13E+06	2.89E+09	2.47E+09	5.57E+03	2.09E+03
	p-Value	3.10E-01	7.94E-03	7.94E-03	1.51E-01	7.94E-03	8.41E-01	5.56E-02	7.94E-03	7.94E-03	7.94E-03	1.35E-01	1.00E+00
	h	≈	+	+	≈	+	≈	≈	+	+	+	≈	≈
F3	mean	1.03E+04	4.84E+04	2.25E+03	5.45E+03	9.65E+03	7.06E+02	6.67E+02	1.45E+04	1.18E+04	1.15E+04	1.69E+02	**1.64E+02**
	std	4.14E+03	2.02E+04	1.06E+03	3.27E+03	1.55E+03	4.85E+02	1.60E+02	9.87E+03	3.76E+03	2.46E+03	3.66E+02	**2.16E+02**
	p-Value	7.94E-03	7.94E-03	7.94E-03	7.94E-03	7.94E-03	1.59E-02	1.59E-02	7.94E-03	7.94E-03	7.94E-03	3.10E-01	1.00E+00
	h	+	+	+	+	+	+	+	+	+	+	≈	≈
F4	mean	3.67E+00	2.23E+01	2.19E+01	3.06E+01	7.50E+02	8.76E+00	1.42E+01	3.53E+01	5.29E+02	1.52E+03	3.58E+01	8.76E+00
	std	3.51E+00	1.80E+01	2.72E+01	1.33E+01	5.80E+02	1.47E+01	1.88E+01	4.54E-01	1.33E+02	8.26E+02	5.17E+01	1.47E+01
	p-Value	1.00E+00	2.22E-01	1.51E-01	1.59E-02	7.94E-03	1.00E+00	5.48E-01	7.94E-03	7.94E-03	7.94E-03	9.52E-02	1.00E+00
	h	≈	≈	≈	+	+	≈	≈	+	+	+	≈	≈
F5	mean	2.00E+01	2.01E+01	2.01E+01	2.04E+01	2.03E+01	2.00E+01	2.04E+01	2.04E+01	2.02E+01	2.04E+01	2.02E+01	**2.01E+01**
	std	2.50E-04	8.26E-02	4.28E-02	4.15E-02	1.48E-01	1.82E-02	7.04E-02	1.45E-01	4.34E-02	7.48E-02	1.48E-01	**1.55E-01**
	p-Value	9.52E-02	1.00E+00	8.41E-01	3.17E-02	1.51E-01	2.22E-01	5.56E-02	3.17E-02	3.10E-01	3.17E-02	8.41E-01	1.00E+00
	h	≈	≈	≈	+	≈	≈	≈	+	≈	+	≈	≈
F6	mean	6.37E+00	7.37E+00	7.21E+00	1.72E+00	8.78E+00	2.97E+00	3.02E+00	7.46E+00	1.03E+01	1.03E+01	4.32E+00	3.79E+00
	std	8.20E-01	2.39E+00	1.33E+00	8.77E-01	1.43E+00	1.48E+00	1.51E+00	1.52E+00	1.20E+00	4.74E-01	2.58E+00	9.28E+01
	p-Value	1.59E-02	1.59E-02	7.94E-03	1.59E-02	7.94E-03	3.10E-01	4.21E-01	7.94E-03	7.94E-03	7.94E-03	6.90E-01	1.00E+00
	h	+	+	+	-	+	≈	≈	+	+	+	≈	≈
F7	mean	4.13E-01	8.00E-01	8.46E-01	7.06E-01	6.25E+01	1.82E-01	3.94E-01	7.27E-01	1.58E+02	6.47E+01	1.13E-01	2.90E-01
	std	4.01E-01	4.95E-01	2.03E-01	1.15E+00	2.87E+01	7.78E-02	1.23E-01	4.26E-01	3.60E+01	2.39E+01	6.21E-02	3.04E-01
	p-Value	5.48E-01	9.52E-02	3.17E-02	4.21E-01	7.94E-03	8.41E-01	3.10E-01	1.51E-01	7.94E-03	7.94E-03	5.48E-01	1.00E+00
	h	≈	≈	+	≈	+	≈	≈	≈	+	+	≈	≈
F8	mean	2.58E-05	3.17E+01	2.66E+01	9.79E+00	6.45E+01	2.03E+01	3.10E+01	4.44E+01	7.14E+01	6.10E+01	1.52E+01	2.49E+01
	std	2.11E-05	4.20E+00	1.01E+01	3.44E+00	1.72E+01	4.02E+00	1.27E+01	1.38E+01	9.29E+00	1.53E+01	5.26E+00	9.56E+00
	p-Value	7.94E-03	3.10E-01	6.90E-01	7.94E-03	7.94E-03	5.48E-01	5.48E-01	5.56E-02	7.94E-03	7.94E-03	9.52E-02	1.00E+00
	h	-	≈	≈	-	+	≈	≈	≈	+	+	≈	≈
F9	mean	1.89E+01	3.99E+01	2.78E+01	1.49E+01	5.91E+01	1.78E+01	4.66E+01	4.69E+01	6.24E+01	5.20E+01	3.22E+01	3.24E+01
	std	8.02E+00	1.32E+01	8.66E+00	7.08E+00	1.69E+01	7.51E+00	4.74E+00	1.02E+01	3.99E+00	5.33E+00	1.10E+01	7.26E+00
	p-Value	3.17E-02	5.48E-01	3.10E-01	1.59E-02	7.94E-03	3.17E-02	1.59E-02	3.17E-02	7.94E-03	7.94E-03	1.00E+00	1.00E+00
	h	-	≈	≈	-	+	-	+	+	+	+	≈	≈
F10	mean	1.76E-01	5.28E+02	3.38E+02	1.65E+02	8.58E+02	5.26E+02	1.09E+03	8.83E+02	1.08E+03	1.36E+03	5.91E+02	1.81E+02
	std	8.17E-02	1.45E+02	2.05E+02	1.44E+02	2.03E+02	2.66E+02	2.68E+02	5.36E+02	5.17E+01	1.30E+02	3.76E+02	2.72E+02
	p-Value	7.94E-03	9.52E-02	4.21E-01	6.90E-01	1.59E-02	5.56E-02	7.94E-03	1.51E-01	7.94E-03	7.94E-03	5.56E-02	1.00E+00
	h	-	≈	≈	≈	+	≈	+	≈	+	+	≈	≈

Table 2 (continued)

Function	Item	GA	WOA	HHO	GWO	RSA	SSA	PRO	DA	SRS	COA	DBO	MSDBO
F11	mean	7.91E+02	1.08E+03	8.67E+02	4.36E+02	1.58E+03	6.60E+02	9.31E+02	1.13E+03	1.58E+03	1.70E+03	9.93E+02	7.37E+02
	std	3.02E+02	4.24E+02	3.29E+02	3.75E+02	2.00E+02	1.99E+02	4.23E+02	9.45E+01	2.07E+02	1.88E+02	1.86E+02	2.14E+02
	p-Value	8.41E-01	1.51E-01	6.90E-01	3.10E-01	7.94E-03	6.90E-01	3.10E-01	1.59E-02	7.94E-03	7.94E-03	1.51E-01	1.00E+00
	h	≈	≈	≈	≈	+	≈	≈	+	+	+	≈	≈
F12	mean	1.34E-01	8.68E-01	6.34E-01	7.64E-01	5.74E-01	2.60E-01	1.86E-01	5.30E-01	4.40E-01	1.33E+00	4.49E-01	**2.29E-01**
	std	1.28E-01	2.95E-01	3.67E-01	6.59E-01	3.76E-01	7.17E-02	2.96E-01	2.68E-01	9.49E-02	1.09E-01	3.82E-01	**8.99E-02**
	p-Value	2.22E-01	7.94E-03	3.17E-02	6.90E-01	9.52E-02	8.41E-01	2.22E-01	7.94E-03	7.94E-03	7.94E-03	2.22E-01	1.00E+00
	h	≈	+	+	≈	≈	≈	≈	+	+	+	≈	≈
F13	mean	3.32E-01	3.87E-01	5.96E-01	1.34E-01	2.95E+00	1.83E-01	3.35E-01	3.81E-01	3.11E+00	3.58E+00	4.88E-01	2.43E-01
	std	1.40E-01	1.13E-01	3.26E-01	5.20E-02	8.29E-01	8.41E-02	7.49E-02	2.16E-01	1.13E+00	7.74E-01	1.39E-01	1.14E-01
	p-Value	3.10E-01	9.52E-02	1.51E-01	9.52E-02	7.94E-03	4.21E-01	3.10E-01	3.10E-01	7.94E-03	7.94E-03	3.17E-02	1.00E+00
	h	≈	≈	≈	≈	+	≈	≈	≈	+	+	+	≈
F14	mean	4.14E-01	2.72E-01	1.70E-01	2.77E-01	2.10E+00	1.62E-01	3.41E-01	2.31E-01	2.80E+01	1.83E+01	5.47E-01	2.45E-01
	std	3.31E-01	4.22E-02	7.66E-02	1.84E-01	1.33E+00	2.89E-02	1.99E-01	1.05E-01	9.14E+00	7.67E+00	3.01E-01	7.88E-02
	p-Value	5.48E-01	5.48E-01	1.51E-01	8.41E-01	7.94E-03	5.56E-02	5.48E-01	8.41E-01	7.94E-03	7.94E-03	3.17E-02	1.00E+00
	h	≈	≈	≈	≈	+	≈	≈	≈	+	+	≈	≈
F15	mean	4.25E+00	4.92E+00	6.68E+00	1.23E+00	2.74E+02	1.72E+00	2.63E+00	3.19E+00	1.05E+03	1.26E+03	2.48E+00	3.10E+00
	std	4.98E+00	2.83E+00	1.94E+00	5.47E-01	4.54E+02	1.09E+00	1.04E+00	1.04E+00	3.07E+02	1.80E+03	1.91E+00	8.31E-01
	p-Value	8.41E-01	5.48E-01	7.94E-03	7.94E-03	7.94E-03	9.52E-02	4.21E-01	1.00E+00	7.94E-03	7.94E-03	3.10E-01	1.00E+00
	h	≈	≈	+	-	+	≈	≈	≈	+	+	≈	≈
F16	mean	2.69E+00	3.19E+00	3.21E+00	2.67E+00	3.51E+00	2.70E+00	2.90E+00	3.46E+00	3.91E+00	3.31E+00	2.92E+00	2.72E+00
	std	6.93E-01	1.81E-01	3.17E-01	2.32E-01	4.43E-01	3.98E-01	1.14E-01	2.27E-01	8.54E-02	1.73E-01	5.23E-01	3.01E+01
	p-Value	6.90E-01	3.17E-02	5.56E-02	1.00E+00	7.94E-03	8.41E-01	6.90E-01	7.94E-03	7.94E-03	7.94E-03	3.10E-01	1.00E+00
	h	≈	+	≈	≈	+	≈	≈	+	+	+	≈	≈
F17	mean	8.92E+05	2.64E+04	1.58E+04	6.81E+03	2.01E+05	1.40E+03	1.76E+03	1.41E+04	3.91E+05	1.91E+05	1.54E+04	5.07E+03
	std	3.85E+05	4.26E+04	2.53E+04	5.31E+03	2.01E+05	1.52E+03	7.43E+02	1.63E+04	1.43E+05	1.84E+05	1.18E+04	3.78E+03
	p-Value	7.94E-03	5.48E-01	8.41E-01	5.48E-01	7.94E-03	9.52E-02	4.21E-01	3.10E-01	7.94E-03	7.94E-03	3.10E-01	1.00E+00
	h	+	≈	≈	≈	+	≈	≈	≈	+	+	≈	≈
F18	mean	6.82E+03	8.86E+03	7.03E+03	5.94E+03	1.27E+04	1.13E+04	4.97E+03	7.91E+03	1.37E+04	9.22E+03	5.42E+03	**2.39E+03**
	std	1.05E+04	7.27E+03	6.74E+03	4.91E+03	8.57E+03	1.14E+04	1.84E+03	8.08E+03	8.22E+03	6.05E+03	5.16E+03	**1.74E+03**
	p-Value	8.41E-01	3.17E-02	3.10E-01	1.51E-01	1.59E-02	2.22E-01	3.17E-02	4.21E-01	7.94E-03	3.17E-02	6.90E-01	1.00E+00
	h	≈	+	≈	≈	+	≈	+	≈	+	+	≈	≈
F19	mean	1.21E+00	4.96E+00	4.72E+00	1.99E+00	8.50E+00	2.44E+00	2.08E+00	5.24E+00	3.76E+01	2.79E+01	2.52E+00	2.38E+00
	std	1.10E+00	1.49E+00	1.44E+00	5.33E-01	2.07E+00	5.23E-01	1.21E+00	2.27E+00	2.49E+01	1.48E+01	1.10E+00	4.93E-01
	p-Value	1.51E-01	7.94E-03	7.94E-03	3.10E-01	7.94E-03	1.00E+00	6.90E-01	1.59E-02	7.94E-03	7.94E-03	5.48E-01	1.00E+00
	h	≈	+	+	≈	+	≈	≈	+	+	+	≈	≈
F20	mean	9.80E+03	3.83E+03	6.79E+03	4.04E+03	5.31E+03	2.36E+02	1.08E+03	2.38E+03	6.65E+03	1.57E+03	2.21E+02	**1.56E+02**
	std	1.04E+04	3.95E+03	3.85E+03	4.04E+03	4.28E+03	1.52E+02	7.19E+02	2.78E+03	2.41E+03	1.33E+03	2.09E+02	**1.25E+02**
	p-Value	7.94E-03	7.94E-03	7.94E-03	2.22E-01	7.94E-03	4.21E-01	7.94E-03	5.56E-02	7.94E-03	7.94E-03	5.48E-01	1.00E+00
	h	+	+	+	≈	+	≈	+	≈	+	+	≈	≈

Table 2 (continued)

Function	Item	GA	WOA	HHO	GWO	RSA	SSA	PRO	DA	SRS	COA	DBO	MSDBO
F21	mean	1.83E+06	2.17E+04	9.33E+03	7.94E+03	1.33E+06	3.43E+03	8.22E+02	6.53E+03	6.28E+04	1.21E+04	8.54E+02	**1.32E+02**
	std	1.41E+06	1.11E+04	9.74E+03	3.33E+03	1.44E+06	4.96E+03	8.06E+01	6.58E+03	7.60E+04	1.55E+04	3.08E+02	**1.12E+02**
	p-Value	7.94E-03	7.94E-03	7.94E-03	7.94E-03	7.94E-03	7.94E-03	7.94E-03	7.94E-03	7.94E-03	7.94E-03	7.94E-03	1.00E+00
	h	+	+	+	+	+	+	+	+	+	+	+	≈
F22	mean	1.70E+02	1.08E+02	1.65E+02	4.33E+01	1.31E+02	4.47E+01	1.49E+02	6.73E+01	1.43E+02	1.55E+02	3.80E+01	**3.03E+01**
	std	1.02E+02	8.98E+01	7.21E+01	4.68E+00	1.04E+02	1.13E+01	4.30E+01	5.70E+01	4.91E+01	1.08E+02	1.20E+01	**9.11E+00**
	p-Value	5.56E-02	9.52E-02	7.94E-03	7.94E-03	7.94E-03	5.56E-02	7.94E-03	5.56E-02	7.94E-03	5.56E-02	2.22E-01	1.00E+00
	h	≈	≈	+	+	+	≈	+	≈	+	≈	≈	≈
F23	mean	3.29E+02	3.30E+02	2.00E+02	3.32E+02	2.00E+02	3.29E+02	2.00E+02	3.41E+02	2.00E+02	2.00E+02	3.29E+02	**2.00E+02**
	std	1.73E-04	4.47E-01	0.00E+00	2.48E+00	0.00E+00	3.28E-11	0.00E+00	1.36E+01	0.00E+00	0.00E+00	6.15E-04	**0.00E+00**
	p-Value	7.94E-03	7.94E-03	1.00E+00	7.94E-03	1.00E+00	7.94E-03	1.00E+00	7.94E-03	1.00E+00	1.00E+00	7.94E-03	1.00E+00
	h	+	+	≈	+	≈	+	≈	+	≈	≈	+	≈
F24	mean	1.51E+02	1.54E+02	1.93E+02	1.34E+02	1.91E+02	1.27E+02	2.00E+02	1.51E+02	2.00E+02	2.00E+02	1.48E+02	1.44E+02
	std	3.27E+01	2.04E+01	1.46E+01	2.43E+01	2.04E+01	8.46E+00	1.41E-06	1.09E+01	0.00E+00	0.00E+00	3.01E+01	3.58E+01
	p-Value	4.21E-01	4.21E-01	9.52E-02	1.00E+00	1.98E-01	8.41E-01	1.51E-01	5.48E-01	1.27E-01	1.27E-01	5.48E-01	1.00E+00
	h	≈	≈	≈	≈	≈	≈	≈	≈	≈	≈	≈	≈
F25	mean	1.88E+02	1.92E+02	2.00E+02	1.92E+02	1.99E+02	2.01E+02	2.00E+02	1.81E+02	2.00E+02	2.00E+02	1.79E+02	**1.53E+02**
	std	2.43E+01	9.37E+00	0.00E+00	1.76E+01	2.02E+00	1.29E+00	0.00E+00	2.82E+01	0.00E+00	7.11E-01	3.09E+01	**2.95E+01**
	p-Value	5.56E-02	5.56E-02	7.94E-03	3.17E-02	7.94E-03	7.94E-03	7.94E-03	9.52E-02	7.94E-03	7.94E-03	1.98E-01	1.00E+00
	h	≈	≈	+	+	+	+	+	≈	+	+	≈	≈
F26	mean	1.20E+02	1.01E+02	1.00E+02	1.00E+02	1.03E+02	1.00E+02	2.00E+02	1.00E+02	1.03E+02	1.03E+02	1.00E+02	**1.00E+02**
	std	4.45E+01	1.07E-01	1.36E-01	2.93E-02	1.73E+00	5.05E-02	0.00E+00	1.80E-01	8.44E-01	1.70E+00	9.70E-02	**1.21E-01**
	p-Value	5.56E-02	1.59E-02	3.10E-01	5.56E-02	7.94E-03	9.52E-02	7.94E-03	2.22E-01	7.94E-03	7.94E-03	9.52E-02	1.00E+00
	h	≈	+	≈	≈	+	≈	+	≈	+	+	≈	≈
F27	mean	4.25E+02	4.29E+02	2.00E+02	3.50E+02	3.39E+02	3.09E+02	3.48E+02	3.25E+02	2.00E+02	1.92E+02	1.65E+02	3.96E+02
	std	3.21E+01	5.83E+01	0.00E+00	2.55E+01	2.11E+02	1.72E+02	1.58E+02	1.77E+02	0.00E+00	1.84E+01	2.17E+02	7.65E+00
	p-Value	3.17E-02	7.94E-03	7.94E-03	7.94E-03	6.90E-01	6.90E-01	1.51E-01	2.22E-01	7.94E-03	7.94E-03	6.90E-01	1.00E+00
	h	+	+	-	-	≈	≈	≈	≈	-	-	≈	≈
F28	mean	5.62E+02	6.03E+02	2.00E+02	4.01E+02	4.18E+02	3.72E+02	4.67E+02	6.43E+02	2.00E+02	2.00E+02	5.20E+02	5.21E+02
	std	6.98E+01	1.13E+02	0.00E+00	5.58E+01	1.24E+02	4.46E+00	3.29E+02	2.67E+02	0.00E+00	0.00E+00	1.25E+02	9.38E+01
	p-Value	4.21E-01	4.21E-01	7.94E-03	1.59E-02	2.22E-01	7.94E-03	3.81E-01	2.22E-01	7.94E-03	7.94E-03	1.00E+00	1.00E+00
	h	≈	≈	-	-	≈	-	≈	≈	-	-	≈	≈
F29	mean	3.37E+02	3.46E+05	1.01E+03	6.04E+02	1.36E+05	1.69E+03	2.54E+06	3.81E+03	2.00E+02	4.43E+02	1.07E+03	3.76E+02
	std	1.23E+02	7.71E+05	5.78E+02	1.75E+02	3.02E+05	1.53E+03	1.65E+06	5.41E+03	0.00E+00	2.00E+02	7.75E+02	7.62E+01
	p-Value	6.90E-01	7.94E-03	1.51E-01	5.56E-02	7.94E-03	7.94E-03	1.59E-02	5.56E-02	7.94E-03	6.90E-01	5.56E-02	1.00E+00
	h	≈	+	≈	≈	+	+	+	≈	-	≈	≈	≈
F30	mean	6.33E+02	2.42E+03	1.84E+03	8.17E+02	4.93E+03	9.20E+02	1.08E+03	2.06E+03	2.00E+02	4.20E+03	1.03E+03	7.31E+02
	std	2.08E+02	8.10E+02	5.99E+02	1.75E+02	2.93E+03	4.59E+02	2.61E+02	8.66E+02	0.00E+00	3.30E+03	3.69E+02	2.26E+02
	p-Value	3.10E-01	7.94E-03	7.94E-03	4.21E-01	7.94E-03	6.90E-01	5.56E-02	7.94E-03	7.94E-03	1.51E-01	2.22E-01	1.00E+00
	h	≈	+	+	≈	+	≈	≈	+	-	≈	≈	≈

Table 2 (continued)

Function	Item	GA	WOA	HHO	GWO	RSA	SSA	PRO	DA	SRS	COA	DBO	MSDBO
	w/t/l	**8/19/3**	**15/15/0**	**13/15/2**	**8/16/6**	**24/6/0**	**6/22/2**	**11/19/0**	**14/16/0**	**23/3/4**	**23/5/2**	**4/26/0**	0/30/0
	F-Rank	**5.71**	**7.43**	**6.33**	**4.78**	**9.11**	**4.12**	**5.99**	**7.61**	**8.88**	**9.02**	**5.16**	**3.82**

algorithm. MSDBO ranks among the top with an F-Rank of 3.85, indicating strong overall performance on the test functions. In contrast, RSA has a lower ranking with an F-Rank of 8.64, reflecting its comparatively weaker performance. The rankings of other algorithms also align with their performance in the tests. Statistical analyses, including the Wilcoxon rank sum test, further support MSDBO's competitive performance compared to the other eleven algorithms on the CEC 2019 test suite.

Figure 9 illustrates the convergence behavior of the algorithms, showing that MSDBO converges rapidly on the test functions. This suggests that the three strategies integrated into MSDBO?dynamic population size variation, forced ranking strategy, and boundary control strategy?contribute to its effectiveness in addressing different aspects of the optimization process. By combining their strengths, MSDBO achieves competitive results in terms of optimization accuracy and convergence speed. These findings suggest that MSDBO has potential as a tool for solving complex optimization problems, and its performance is competitive with other state-of-the-art algorithms.

4.3 Parameter analysis

The parameters have a significant role in determining the efficiency of MAs. Consequently, experiments were conducted to investigate the effect of various values of p on the effectiveness of MSDBO. Throughout the experiments, the value of p was varied in the range of 0.2 to 0.8.

Table 4 presents the statistical results for MSDBO with the parameter p on all CEC 2014 functions. It is clear from Table 4 that MSDBO has distinct performance with different values of the parameters. Overall, when $p = 0.2$, MSDBO achieves the best average ranking among all algorithms, with a ranking of 1.87. However, no optimal average results are obtained for $p = 0.5$, and $p = 0.8$. Based on the observations above, parameter p is set to the same 0.2.

Similarly, sensitivity analysis was conducted on the parameters p_1 and p_2 of the proposed algorithm, and three different combinations of p_1 and p_2 parameters were tested on all 30 functions of CEC 2014. The test results are shown in Table 5. It can be seen that when the parameter combination is $p_1 = 0.5$ and $p_2 = 0.6$, the algorithm achieves the best results. Therefore, this set of parameter combinations is used for algorithm testing.

4.4 Effects of three strategies

In order to investigate the effectiveness of the proposed dynamic population size variation, forced ranking strategy, and boundary control strategy, MSDBO is compared to the original DBO, a DBO algorithm with added proposed

Table 3 Results of CEC 2019 benchmark test functions

Function	Item	GA	WOA	HHO	GWO	RSA	SSA	PRO	DA	SRS	COA	DBO	MSDBO
F1	mean	1.73E+07	1.19E+06	1.00E+00	5.99E+03	1.00E+00	2.86E+05	1.00E+00	1.86E+07	1.00E+00	1.00E+00	1.76E+05	**1.00E+00**
	std	1.36E+07	5.86E+05	0.00E+00	6.15E+03	0.00E+00	2.33E+05	8.39E-14	1.99E+07	0.00E+00	0.00E+00	3.51E+05	**2.46E-13**
	p-Value	7.94E-03	7.94E-03	1.67E-01	7.94E-03	1.67E-01	7.94E-03	3.81E-01	7.94E-03	1.67E-01	1.67E-01	7.94E-03	1.00E+00
	h	+	+	≈	+	≈	+	≈	+	≈	≈	+	≈
F2	mean	1.05E+04	6.72E+03	4.96E+00	1.72E+02	5.00E+00	2.79E+02	4.21E+00	4.66E+03	4.40E+00	5.00E+00	7.69E+02	**4.42E+00**
	std	2.68E+03	2.47E+03	7.89E-02	1.55E+02	1.30E-04	1.54E+02	8.03E-02	2.53E+03	3.36E-01	0.00E+00	1.65E+03	**3.27E-01**
	p-Value	7.94E-03	7.94E-03	4.76E-02	7.94E-03	4.76E-02	7.94E-03	1.51E-01	7.94E-03	7.46E-01	4.76E-02	6.90E-01	1.00E+00
	h	+	+	+	+	+	+	≈	+	≈	+	≈	≈
F3	mean	5.77E+00	2.72E+00	2.50E+00	1.39E+00	7.60E+00	3.20E+00	5.56E+00	8.10E+00	4.75E+00	7.22E+00	4.26E+00	1.41E+00
	std	2.82E+00	1.06E+00	8.12E-01	2.34E-01	9.95E-01	2.36E+00	1.04E+00	1.21E+00	5.38E-01	4.66E-01	2.82E+00	2.03E-15
	p-Value	7.94E-03	7.94E-03	7.94E-03	1.35E-01	7.94E-03	7.94E-03	7.94E-03	7.94E-03	7.94E-03	7.94E-03	7.14E-02	1.00E+00
	h	+	+	+	≈	+	+	+	+	+	+	≈	≈
F4	mean	2.63E+01	5.18E+01	4.17E+01	1.03E+01	9.98E+01	1.59E+01	7.90E+01	5.41E+01	7.83E+01	8.14E+01	2.52E+01	2.61E+01
	std	9.43E+00	1.67E+01	9.12E+00	3.26E+00	1.21E+01	5.12E+00	2.59E+01	1.27E+01	1.28E+01	1.30E+01	6.01E+00	1.24E+01
	p-Value	1.00E+00	3.17E-02	5.56E-02	7.94E-03	7.94E-03	2.22E-01	7.94E-03	1.59E-02	7.94E-03	7.94E-03	8.41E-01	1.00E+00
	h	≈	+	≈	-	+	≈	+	+	+	+	≈	≈
F5	mean	1.45E+00	1.82E+00	2.22E+00	1.76E+00	4.99E+01	1.14E+00	1.45E+00	1.72E+00	9.58E+01	6.57E+01	1.28E+00	1.42E+00
	std	4.39E-01	4.17E-01	3.24E-01	1.09E+00	2.40E+01	2.83E-02	1.12E-01	1.20E-01	1.76E+01	9.18E+00	6.54E-02	2.37E-01
	p-Value	6.90E-01	5.56E-02	7.94E-03	6.90E-01	7.94E-03	1.59E-02	1.00E+00	5.56E-02	7.94E-03	7.94E-03	6.90E-01	1.00E+00
	h	≈	≈	+	≈	+	-	≈	≈	+	+	≈	≈
F6	mean	4.90E+00	8.51E+00	6.06E+00	2.19E+00	1.14E+01	2.41E+00	1.64E+00	6.79E+00	1.17E+01	1.01E+01	5.89E+00	4.73E+00
	std	1.61E+00	2.35E+00	1.60E+00	9.03E-01	8.97E-01	1.69E+00	6.54E-01	2.42E+00	1.79E+00	1.03E+00	2.10E+00	8.43E-01
	p-Value	8.41E-01	1.59E-02	9.52E-02	7.94E-03	7.94E-03	9.52E-02	7.94E-03	9.52E-02	7.94E-03	7.94E-03	3.10E-01	1.00E+00
	h	≈	+	≈	-	+	≈	-	≈	+	+	≈	≈
F7	mean	7.19E+02	1.37E+03	1.06E+03	6.06E+02	1.61E+03	1.01E+03	1.32E+03	1.23E+03	2.03E+03	1.84E+03	1.19E+03	8.37E+02
	std	2.54E+02	3.06E+02	3.39E+02	2.83E+02	1.62E+02	3.94E+02	3.09E+02	3.95E+02	2.73E+02	1.99E+02	4.40E+02	2.72E+02
	p-Value	5.48E-01	3.17E-02	3.10E-01	2.22E-01	7.94E-03	5.48E-01	3.17E-02	1.51E-01	7.94E-03	7.94E-03	2.22E-01	1.00E+00
	h	≈	+	≈	≈	+	≈	+	≈	+	+	≈	≈
F8	mean	3.87E+00	4.26E+00	4.55E+00	3.98E+00	4.95E+00	3.85E+00	3.96E+00	4.48E+00	5.15E+00	4.86E+00	4.24E+00	**3.75E+00**
	std	3.60E-01	3.32E-01	2.71E-01	6.49E-01	1.62E-01	2.71E-01	1.33E-01	2.14E-01	2.16E-01	2.62E-01	3.15E-01	**4.52E-01**
	p-Value	4.21E-01	1.51E-01	3.17E-02	3.10E-01	7.94E-03	5.48E-01	1.51E-01	3.17E-02	7.94E-03	7.94E-03	5.56E-02	1.00E+00
	h	≈	≈	+	≈	+	≈	≈	+	+	+	≈	≈
F9	mean	1.31E+00	1.30E+00	1.33E+00	1.14E+00	2.10E+00	1.20E+00	1.23E+00	1.34E+00	3.63E+00	3.30E+00	1.36E+00	1.35E+00
	std	8.32E-02	8.09E-02	7.48E-02	3.70E-02	1.04E+00	6.70E-02	1.53E-02	1.22E-01	3.85E-01	9.11E-01	9.05E-02	9.49E-02
	p-Value	5.48E-01	4.21E-01	6.90E-01	1.59E-02	4.21E-01	3.17E-02	1.51E-01	6.90E-01	7.94E-03	7.94E-03	1.00E+00	1.00E+00
	h	≈	≈	≈	-	≈	-	≈	≈	+	+	≈	≈
F10	mean	2.10E+01	2.11E+01	2.11E+01	2.14E+01	2.12E+01	2.09E+01	2.13E+01	2.13E+01	2.11E+01	2.13E+01	2.13E+01	2.11E+01
	std	2.88E-04	7.81E-02	5.73E-02	7.96E-02	9.47E-02	1.12E-01	1.59E-01	7.10E-02	4.84E-02	1.30E-01	9.45E-02	1.26E-01
	p-Value	6.90E-01	4.21E-01	1.51E-01	1.59E-02	1.51E-01	2.22E-01	5.56E-02	5.56E-02	1.51E-01	3.17E-02	5.56E-02	1.00E+00
	h	≈	≈	≈	+	≈	≈	≈	≈	≈	+	≈	≈

Table 3 (continued)

Function	Item	GA	WOA	HHO	GWO	RSA	SSA	PRO	DA	SRS	COA	DBO	MSDBO
	w/t/l	**3/7/0**	**6/4/0**	**4/6/0**	**3/4/3**	**7/3/0**	**3/5/2**	**3/6/1**	**5/5/0**	**7/3/0**	**9/1/0**	**1/9/0**	**0/10/0**
	F-Rank	**5.88**	**7.12**	**6.05**	**4.55**	**8.64**	**4.13**	**5.4**	**8.6**	**8.57**	**9.07**	**6.12**	**3.85**

dynamic population (called DBO_DP), and an algorithm not utilizing boundary control strategy (called DBO_DPFR). The four algorithms were tested on six functions from CEC 2014, respectively, and the outcomes are as presented in Table 6.

The curves of convergence for DBO, DBO_DP, DBO_DPFR and MSDBO in $F2$, $F9$, $F13$, $F17$, $F22$ and $F25$ for all six functions of functions of CEC 2014 are illustrated in Fig. 6.

The dynamic adjustment of population size is aimed at improving the algorithm's adaptability during the optimization process. By changing the population size dynamically, the algorithm can balance between exploration and exploitation more effectively. A smaller population size might promote quicker convergence in the later stages of the optimization process, while a larger population could help explore the solution space more thoroughly in the earlier stages.

From the comparison with DBO and DBO_DP, it is evident that dynamic population size variation enhances convergence speed and solution accuracy. This strategy is particularly useful when the problem's landscape is complex and requires a balance between global exploration and local exploitation. The results highlight that MSDBO, with this strategy, achieves a faster and more accurate convergence than its counterparts.

The forced ranking strategy introduces a mechanism that influences the selection process in the algorithm. By imposing a forced ranking, the algorithm gives higher priority to solutions with better fitness, ensuring that more promising solutions are considered first, thus accelerating the optimization process. This strategy helps in improving the convergence speed by focusing on higher-quality candidates in each iteration.

The DBO_DPFR (which incorporates forced ranking but not the boundary control strategy) shows improved performance over DBO and DBO_DP. This suggests that forced ranking contributes significantly to guiding the algorithm towards better solutions more efficiently. By focusing on the most promising solutions, it results in faster convergence.

Boundary control ensures that the candidate solutions remain within the feasible region of the search space, preventing them from wandering outside the problem's constraints. This strategy is particularly useful when dealing with constrained optimization problems or when the search space has defined boundaries. It helps improve the algorithm's robustness by avoiding solutions that violate the constraints, thereby improving both the quality and reliability of the solutions.

The comparison between MSDBO and DBO_DPFR suggests that boundary control has a significant role in enhancing the performance of the algorithm. DBO_DPFR,

Table 4 MSDBO on functions in CEC 2014 of 10D with the p varying from 0.2 to 0.8

Function	Item	p = 0.2	p = 0.8	p = 0.5	Function	Item	p = 0.2	p = 0.8	p = 0.5
F1	Mean	2.06E+04	1.75E+04	2.21E+04	F17	Mean	5.41E+03	4.08E+03	5.07E+03
F2	Mean	1.61E+03	2.44E+03	1.38E+03	F18	Mean	3.05E+03	5.28E+03	2.39E+03
F3	Mean	3.29E+02	3.70E+02	1.64E+02	F19	Mean	3.00E+00	2.39E+00	2.38E+00
F4	Mean	2.87E+01	2.09E+01	8.76E+00	F20	Mean	9.62E+02	3.20E+02	1.56E+02
F5	Mean	2.01E+01	2.00E+01	2.01E+01	F21	Mean	2.56E+02	2.25E+02	1.32E+02
F6	Mean	4.04E+00	3.53E+00	3.79E+00	F22	Mean	3.10E+01	4.86E+01	3.03E+01
F7	Mean	5.00E-01	3.40E-01	2.90E-01	F23	Mean	2.00E+02	2.00E+02	2.00E+02
F8	Mean	1.69E+01	1.69E+01	2.49E+01	F24	Mean	1.29E+02	1.46E+02	1.44E+02
F9	Mean	2.17E+01	2.63E+01	3.24E+01	F25	Mean	1.70E+02	1.55E+02	1.53E+02
F10	Mean	2.04E+02	1.94E+02	1.81E+02	F26	Mean	1.00E+02	1.00E+02	1.00E+02
F11	Mean	9.68E+02	7.76E+02	7.37E+02	F27	Mean	3.19E+02	3.68E+02	3.96E+02
F12	Mean	1.59E-01	3.07E-01	2.29E-01	F28	Mean	4.96E+02	4.54E+02	5.21E+02
F13	Mean	3.27E-01	3.27E-01	2.43E-01	F29	Mean	5.25E+02	3.45E+05	3.76E+02
F14	Mean	2.34E-01	2.77E-01	2.45E-01	F30	Mean	1.27E+03	1.38E+03	7.31E+02
F15	Mean	3.40E+00	1.98E+00	3.10E+00		**w/t/l**	**1/29/0**	**0/29/1**	**0/30/0**
F16	Mean	3.09E+00	2.87E+00	2.72E+00		**F-rank**	**2.12**	**2**	**1.87**

Table 5 MSDBO on functions in CEC 2014 of 10D with different p_1 and P_2

Function	Item	p_1=0.4 p_2=0.5	p_1=0.6 p_2=0.8	p_1=0.5 p_2=0.6	Function	Item	p_1=0.4 p_2=0.5	p_1=0.6 p_2=0.8	p_1=0.5 p_2=0.6
F1	Mean	1.58E+04	2.48E+04	2.21E+04	F17	Mean	2.64E+03	4.73E+03	5.07E+03
F2	Mean	2.78E+03	1.27E+03	1.38E+03	F18	Mean	9.87E+03	7.40E+03	2.39E+03
F3	Mean	3.08E+02	3.84E+02	1.64E+02	F19	Mean	2.71E+00	4.14E+00	2.38E+00
F4	Mean	3.45E+01	1.49E+01	8.76E+00	F20	Mean	2.43E+03	7.04E+02	1.56E+02
F5	Mean	2.02E+01	2.02E+01	2.01E+01	F21	Mean	1.01E+02	2.85E+02	1.32E+02
F6	Mean	4.02E+00	3.68E+00	3.79E+00	F22	Mean	2.70E+01	2.47E+01	3.03E+01
F7	Mean	4.65E-01	2.51E-01	2.90E-01	F23	Mean	2.00E+02	2.00E+02	2.00E+02
F8	Mean	1.63E+01	1.57E+01	2.49E+01	F24	Mean	1.34E+02	1.39E+02	1.44E+02
F9	Mean	2.28E+01	2.71E+01	3.24E+01	F25	Mean	1.58E+02	1.69E+02	1.53E+02
F10	Mean	2.39E+02	2.33E+02	1.81E+02	F26	Mean	1.00E+02	1.00E+02	1.00E+02
F11	Mean	6.35E+02	3.01E+02	7.37E+02	F27	Mean	3.18E+02	4.03E+02	3.96E+02
F12	Mean	3.87E-01	6.20E-01	2.29E-01	F28	Mean	6.22E+02	4.61E+02	5.21E+02
F13	Mean	3.14E-01	3.39E-01	2.43E-01	F29	Mean	4.47E+02	4.32E+02	3.76E+02
F14	Mean	2.19E-01	3.25E-01	2.45E-01	F30	Mean	7.54E+02	1.28E+03	7.31E+02
F15	Mean	2.21E+00	3.81E+00	3.10E+00		**w/t/l**	**1/29/0**	**4/25/1**	**0/30/0**
F16	Mean	2.69E+00	3.10E+00	2.72E+00		**F-rank**	**2.04**	**2.12**	**1.83**

which lacks boundary control, performs slightly worse than MSDBO, where boundary control is employed. This shows that boundary control helps avoid infeasible regions of the search space, resulting in more accurate solutions and preventing the algorithm from wasting resources on invalid solutions.

MSDBO combines all three strategies: dynamic population size variation, forced ranking, and boundary control, leading to its superior performance. From the results in both the table and the convergence curves (Fig. 6), it is clear that MSDBO achieves faster and more accurate convergence than the individual strategies alone.

The dynamic population size variation helps adjust the exploration-exploitation balance; the forced ranking strategy prioritizes better solutions, enhancing convergence speed; and the boundary control ensures that only feasible solutions are considered, making the algorithm more robust and reliable. The synergistic effect of these strategies results in MSDBO outperforming the other variants (DBO_DP, and DBO_DPFR) across various test functions. The combined strategies not only accelerate the convergence but also improve the quality of the final solutions, demonstrating the effectiveness of MSDBO in solving optimization problems efficiently.

By analyzing the contribution of each strategy, it can be observed that each one provides a distinct benefit to the algorithm. Their combination contributes to improvements in both convergence speed and solution quality.

Table 6 Experimental results of of DBO, DBO_DP, DBO_DPFR and MSDBO on selected functions

Function	Item	DBO	DBO_DP	DBO_DPFR	MSDBO
F2	Mean	6.50E+03	5.11E+03	5.76E+03	1.38E+03
	Std	5.57E+03	5.35E+03	5.75E+03	2.09E+03
	p-Value	1.35E-01	2.22E-01	2.06E-01	1.00E+00
	h	≈	≈	≈	≈
F9	Mean	3.22E+01	3.32E+01	2.65E+01	3.24E+01
	Std	1.10E+01	1.23E+01	5.02E+00	7.26E+00
	p-Value	1.00E+00	6.90E-01	2.22E-01	1.00E+00
	h	≈	≈	≈	≈
F13	Mean	4.88E-01	3.97E-01	3.09E-01	2.43E-01
	Std	1.39E-01	1.12E-01	7.29E-02	1.14E-01
	p-Value	3.17E-02	9.52E-02	4.21E-01	1.00E+00
	h	+	≈	≈	≈
F17	Mean	1.54E+04	1.38E+04	7.46E+03	5.07E+03
	Std	1.18E+04	1.65E+04	5.50E+03	3.78E+03
	p-Value	3.10E-01	1.51E-01	2.22E-01	1.00E+00
	h	≈	≈	≈	≈
F22	Mean	3.80E+01	4.51E+01	2.53E+01	3.03E+01
	Std	1.20E+01	2.72E+01	3.80E+00	9.11E+00
	p-Value	2.22E-01	2.22E-01	6.90E-01	1.00E+00
	h	≈	≈	≈	≈
F25	Mean	1.79E+02	1.61E+02	1.56E+02	1.53E+02
	Std	3.09E+01	2.30E+01	1.99E+01	2.95E+01
	p-Value	1.98E-01	5.48E-01	1.00E+00	1.00E+00
	h	≈	≈	≈	≈
	w/t/l	**1/5/0**	**0/6/0**	**0/6/0**	**0/6/0**
	F-rank	**2.85**	**2.83**	**2.35**	**1.97**

5 MSDBO for engineering design problems

In this section, firstly, the effectiveness of the proposed algorithm in engineering applications is evaluated on several problems from the CEC 2011 [57] test suite. A representative set of optimization problems is selected, spanning various domains such as parameter estimation, potential energy optimization, chemical process control, and network planning. For instance, the parameter estimation problem of frequency-modulated (FM) sound waves seeks to minimize the error between the synthesized and target sound waves, posing a six-dimensional, complex multimodal optimization challenge. The Lennard–Jones potential problem involves minimizing molecular potential energy, with its dimensionality varying based on the number of atoms, while exhibiting a significant number of local minima. The bifunctional catalyst blend optimal control problem focuses on optimizing the catalyst blend ratio in a chemical process to maximize the benzene concentration. These problems differ in dimensionality, constraint conditions, and variable bounds, providing a rich and diverse set of scenarios for evaluating the performance of optimization algorithms.

Then, the practical applications of the proposed MSDBO algorithm in tackling several engineering design challenges frequently encountered in industrial settings are investigated. Specifically, the welded beam design problem, the pressure vessel design problem, the speed reducer design problem, and the tubular column design problem are addressed. In this study, the exterior penalty function method [58] is used to handle constraints by converting them into penalty functions and adding them to the objective function. Violations of constraints are penalized based on a dynamically adjusted penalty parameter, guiding the optimization process towards feasible solutions. This approach ensures the algorithm converges to an optimal solution while satisfying the constraints. The effectiveness and superiority of the proposed algorithm are demonstrated in comparison to other existing algorithms commonly employed in the industry. To assess the performance of the algorithm, a comparative analysis with established algorithms widely used in engineering design applications was conducted. This comparative evaluation allows for an accurate assessment of the strengths and weaknesses of the MSDBO algorithm in relation to its counterparts.

5.1 Results on CEC 2011 benchmarks functions

In the field of optimization algorithm research, it is crucial to test the performance of algorithms through practical problems. This paper focuses on several real-world optimization problems in the CEC 2011 test suite. By using relevant test functions and specific algorithms, we conduct in-depth research and analyze the convergence characteristics and performance of the algorithms in different problem scenarios, aiming to provide valuable references for the development and application of optimization algorithms.

These algorithms include Equilibrium Optimizer (EO) [59], Teaching-Learning-based Optimization (TLBO) [60], Whale Optimization Algorithm (WOA), Grey Wolf Optimizer (GWO), Kepler Optimization Algorithm (KOA) [61], and Dung Beetle Algorithm (DBO). According to the requirements of the CEC 2011 competition, each algorithm is run independently 25 times. In terms of function evaluations (FEs), the results of the algorithms are recorded at 1000 FEs respectively, ensuring that the convergence of the algorithms at different computational stages can be comprehensively observed. A uniform random initialization method is adopted for all algorithms within the prescribed search space during initialization to ensure the fairness and randomness of the experiment and to avoid unreasonable influences on the results due to initialization differences.

The convergence curve is shown in Fig. 7. From the convergence curves of the F1-F4 functions, in the initial stage, the objective function values of all algorithms are generally high and the descent speed is fast. This indicates that the algorithms can quickly explore the solution space and find

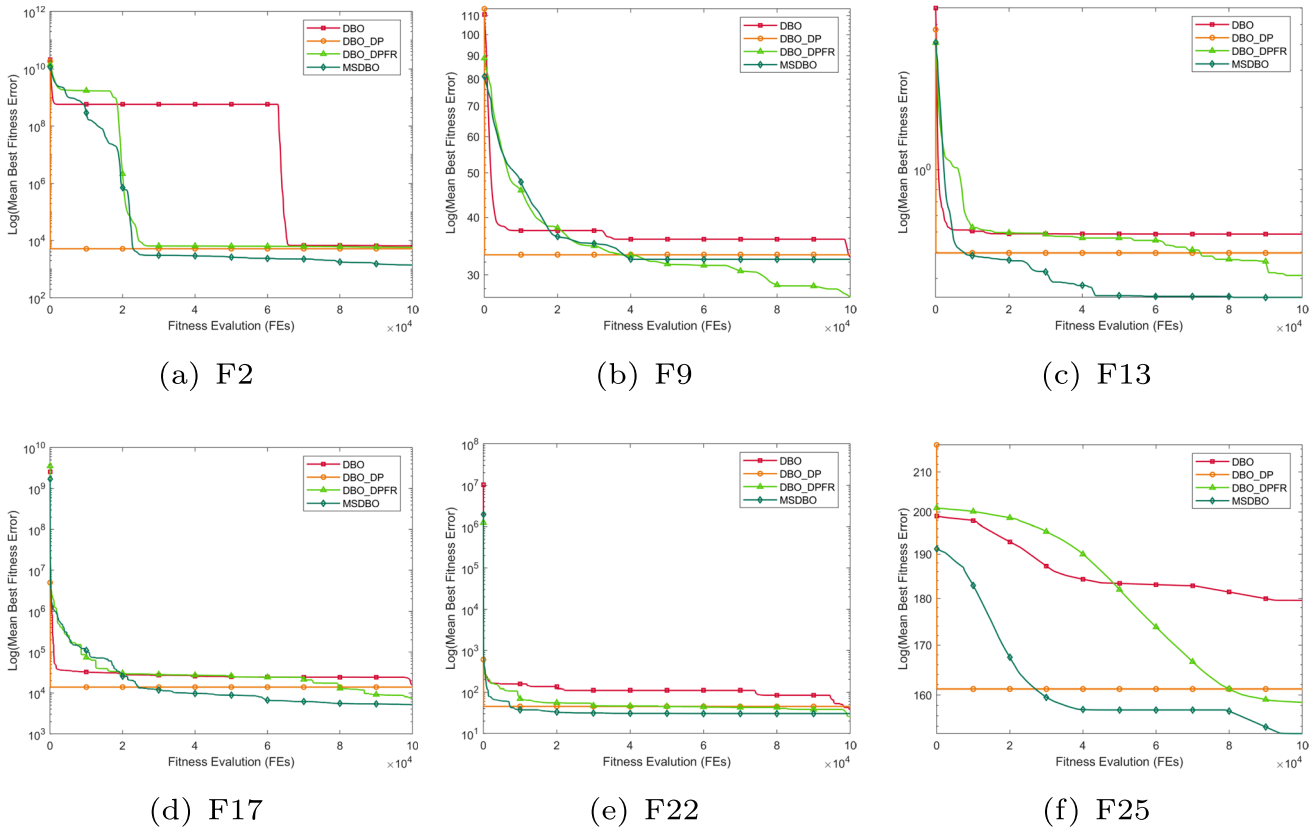


Fig. 6 Convergence process of DBO, DBO_DP, DBO_DPFR and MSDBO on selected functions

some good solution regions in the early search. With the increase of FEs, the descent speed gradually slows down and the curves gradually flatten, indicating that the search difficulty increases when the algorithms approach the optimal solution, and more refined search strategies are required to further optimize the solution.

In the F1 function, by observing the curves, it can be seen that at 500 FEs, the performance of each algorithm has already shown obvious differentiation. Among them, MSDBO performs relatively well, and its fitness value is low, indicating that this algorithm can quickly approach the optimal solution region during the search process. In the F2 and F3 functions, the final fitness value of all algorithms is close to the optimal, showing its effectiveness and stability in dealing with this function. In the F4 function, MSDBO and other algorithms perform similarly in the early stage, but in the 800-1000 FEs interval, the convergence speed of MSDBO increases significantly, and finally obtains a better result. This may be due to its unique update strategy or parameter adjustment mechanism, which enables it to jump out of the local optimal and find a better solution in the later stage.

5.2 Welded beam design problem

The welding beam design problem aims to minimize the manufacturing cost of the beam while meeting specific constraints. This problem is of significant importance in the industry, as welding beams are widely used in various applications. Figure 10 illustrates the structure of a welded beam, which consists of beam A and the welds required to connect it to object B. The optimization constraints include the shear stress and bending stress in the beam, the buckling load on the bar, and the bending stiffness of the end of the beam. These constraints ensure that the beam can withstand the expected loads and maintain its shape during use.

The optimization variables in the welding beam design problem are the thickness (h) of the weld, the length (l) of the clamped bar, the height (t) of the steel, and the thickness (b) of the steel. Adjusting these variables can help to minimize the manufacturing cost of the beam while ensuring that the beam meets the specified constraints.

Mathematically, the welding beam design problem can be formulated as follows.

Consider:

$$\vec{z} = [z_1\ z_2\ z_3\ z_4] = [h\ l\ t\ b] \tag{13}$$

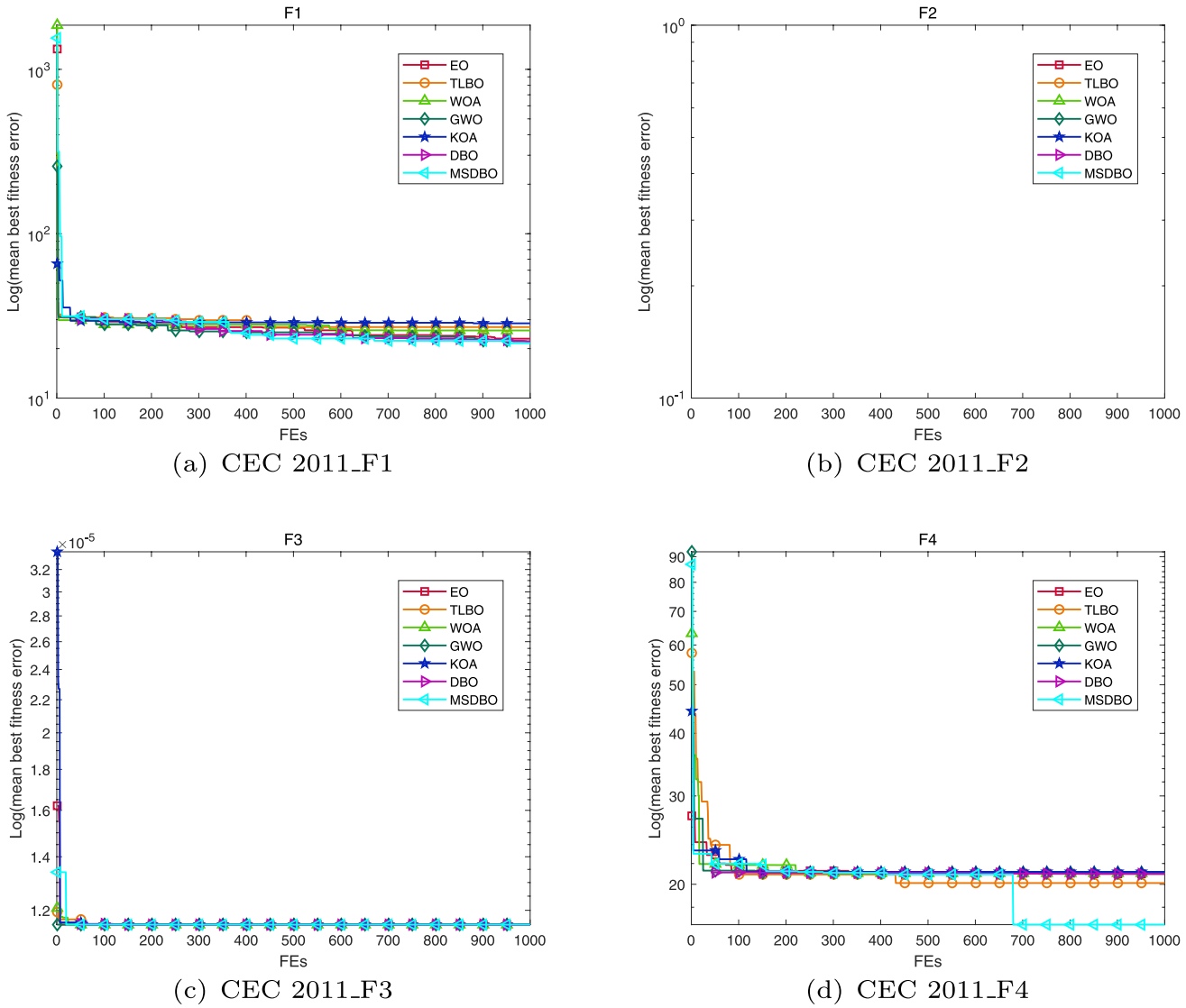


Fig. 7 Convergence curves of different algorithms on engineering design problems in CEC 2011

Minimize:

$$f(\vec{z}) = 1.10471 z_1^2 z_2 + 0.04811 z_3 z_4 (14.0 + z_2) \tag{14}$$

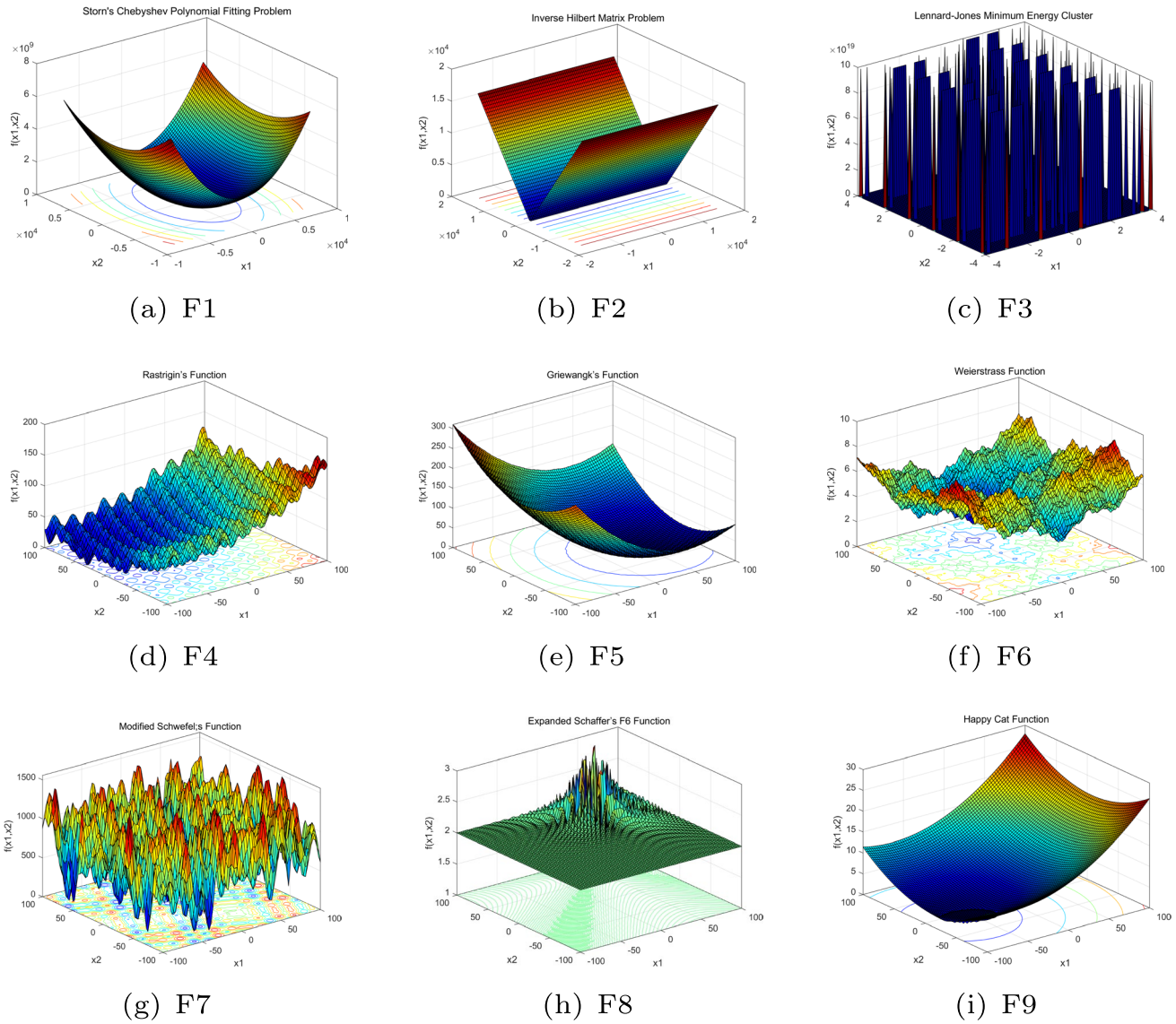
Subject to:

$$\begin{aligned} &g_1(\vec{z}) = \tau(\vec{z}) - \tau_{\max} \leqslant 0, \\ &g_2(\vec{z}) = \sigma(\vec{z}) - \sigma_{\max} \leqslant 0, \\ &g_3(\vec{z}) = \delta(\vec{z}) - \delta_{\max} \leqslant 0, \\ &g_4(\vec{z}) = z_1 - z_4 \leqslant 0, \\ &g_5(\vec{z}) = P - P_c(\vec{z}) \leqslant 0, \\ &g_6(\vec{z}) = 0.125 - z_1 \leqslant 0, \\ &g_7(\vec{z}) = 1.10471 z_1^2 + 0.04811 z_3 z_4 (14.0 + z_2) - 5.0 \leqslant 0, \end{aligned} \tag{15}$$

where, $0.05 \leqslant z_1 \leqslant 2.00$, $0.25 \leqslant z_2 \leqslant 1.30$, $2.00 \leqslant z_3 \leqslant 15.0$.

The solutions obtained from the proposed MSDBO algorithm is compared with those obtained from other advanced algorithms in the literature. These algorithms include Equilibrium Optimizer (EO) [59], Teaching-Learning-based Optimization (TLBO) [60], Whale Optimization Algorithm (WOA), Grey Wolf Optimizer (GWO), Kepler Optimization Algorithm (KOA) [61], and Dung Beetle Algorithm (DBO). The comparison results are shown in Table 7.

The proposed MSDBO algorithm achieves the optimal solution at $z^* = (0.3173, 1.9908, 9.0367, 0.2067)$, with a corresponding fitness value of $f(z^*) = 1.6524$. This solution outperforms all other algorithms considered in this study. The comparison results show that the proposed MSDBO algorithm converges faster and provides a better solution than other algorithms.



(a) F1 (b) F2 (c) F3

(d) F4 (e) F5 (f) F6

(g) F7 (h) F8 (i) F9

Fig. 8 3D function plot on the CEC 2019 benchmarks

Figure 14 (a) shows the convergence analysis of the best solutions obtained from the proposed MSDBO algorithm and other algorithms. The plot indicates that the MSDBO algorithm converges faster and achieves a better solution compared to the other algorithms. This result suggests that the proposed MSDBO algorithm performs competitively in solving the welding beam design problem.

5.3 Pressure vessel design problem

This problem was first proposed to minimize the total cost of a pressure vessel, which includes material cost, shaping cost, and welding cost. Cylindrical containers are widely used in various industries, including chemical, oil and gas, and power generation. The container has a half-spherical cover at both ends, which adds to the complexity of the problem. There are four variables in this problem: the thickness of the shell (T_s), the thickness of the head (T_h), the inner radius (R), and the length of the cylindrical section without considering the head (L). Figure 11 illustrates the pressure vessel, which is covered by half-spherical covers at both ends.

Among these four design variables, R and L are continuous variables, while T_s and T_h are integer multiples of 0.0625 inches. Adjusting these variables can help to minimize the total cost of the pressure vessel while ensuring that it meets specific constraints.

Mathematically, the pressure vessel design problem can be formulated as follows.

Consider:

$$\vec{z} = [z_1 z_2 z_3 z_4] = [T_s T_h R L] \tag{16}$$

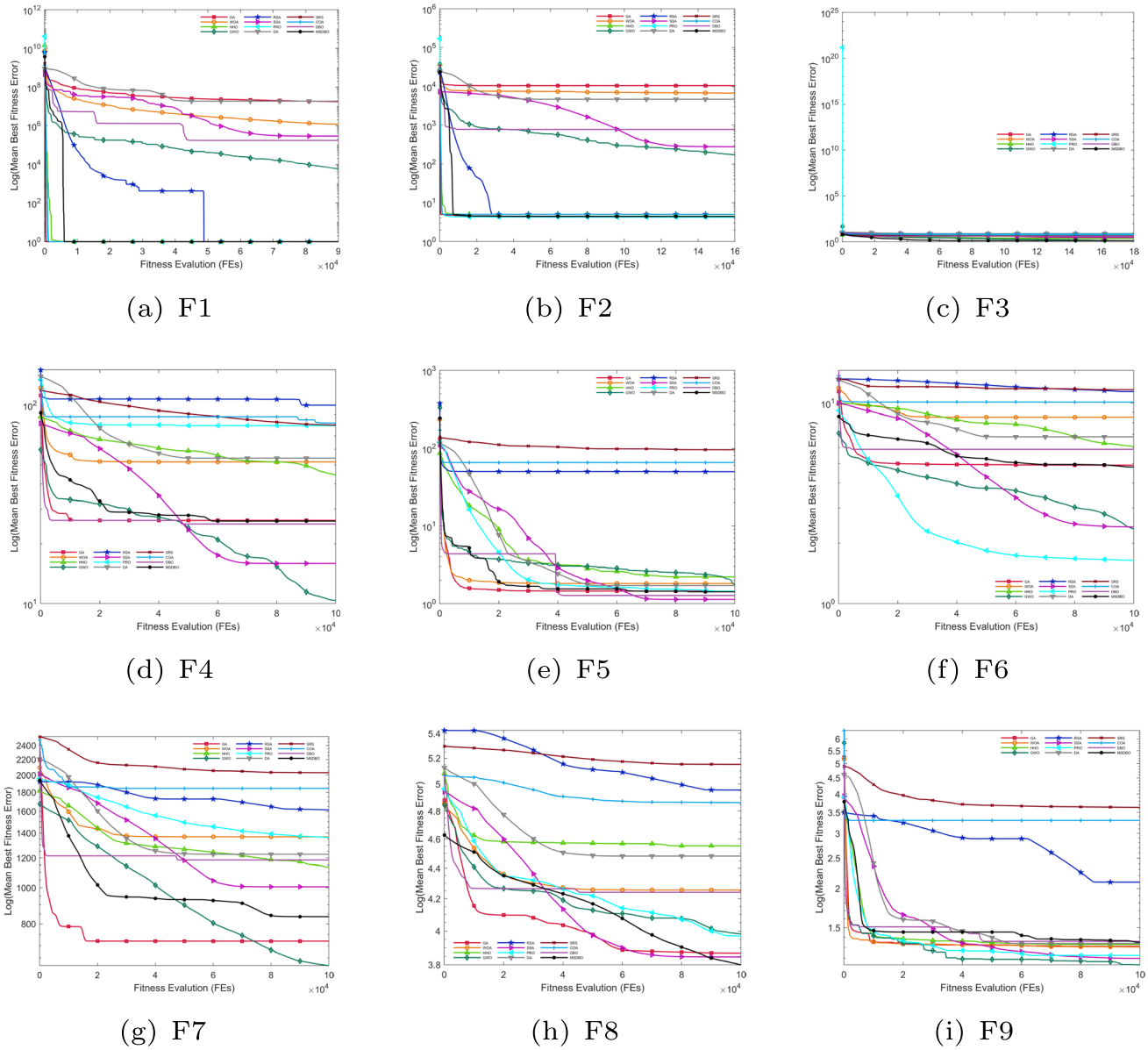


Fig. 9 Convergence curves of different algorithms obtained on the CEC 2019 benchmarks

Minimize:

$$f(\vec{z}) = 0.6224 z_1 z_3 z_4 + 1.7781 z_2 z_3^2 + 3.1661 z_1^2 z_4 + 19.84 z_1^2 z_3 \tag{17}$$

Subject to:

$$\begin{aligned} g_1(\vec{z}) &= -z_1 + 0.0193 z_3 \leqslant 0, \\ g_2(\vec{z}) &= -z_3 + 0.00954 z_3 \leqslant 0, \\ g_3(\vec{z}) &= -\pi z_3^2 z_4 - \frac{4}{3}\pi z_3^2 + 1,296,000 \leqslant 0, \\ g_4(\vec{z}) &= z_4 - 240 \leqslant 0, \end{aligned} \tag{18}$$

where, $0 \leqslant z_1 \leqslant 99$, $0 \leqslant z_2 \leqslant 99$, $0 \leqslant z_3 \leqslant 200$, $0 \leqslant z_4 \leqslant 200$.

Table 8 compares the optimal solutions obtained from the proposed MSDBO algorithm with those obtained from other advanced algorithms. The purpose of this comparison is to evaluate the effectiveness of the proposed algorithm in finding the optimal design of the pressure vessel with minimal cost. The optimal solution achieved by the proposed MSDBO algorithm is at $z^* = (0.7782, 0.3847, 40.3200, 200.0000)$, with a corresponding fitness value of $f(z^*) = 5885.3360$. This solution outperforms all other algorithms considered in this study.

The comparison results demonstrate that the proposed MSDBO algorithm is effective in finding the optimal solution with minimal cost for the pressure vessel design problem. The optimal solution obtained by the MSDBO

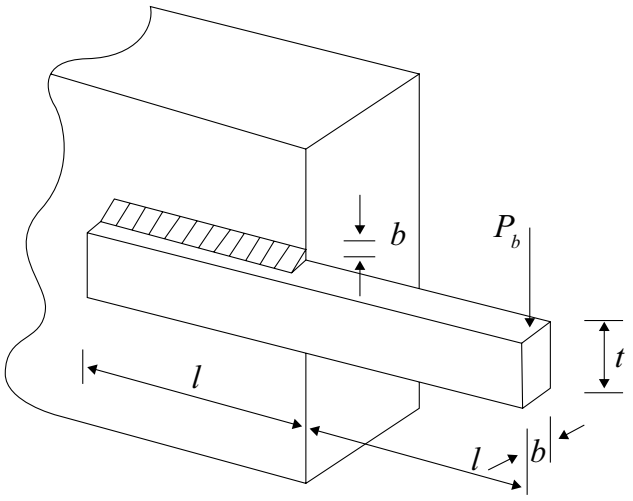


Fig. 10 Welded beam design problem

Table 7 Comparison results for welded beam design problem

Algorithm	*h*	*l*	*t*	b	Optimum cost
EO	0.1471	5.4907	10.0000	0.2177	2.1729
TLBO	0.2047	3.5363	9.0043	0.2100	1.7592
WOA	0.2056	3.4721	9.0409	0.2057	1.7255
GWO	0.1974	3.3151	10.0000	0.2014	1.8204
KOA	0.2057	3.7540	9.0370	0.2014	1.7270
DBO	0.2870	2.2332	9.0365	0.2057	1.6552
MSDBO	0.3173	1.9908	9.0367	0.2067	1.6524

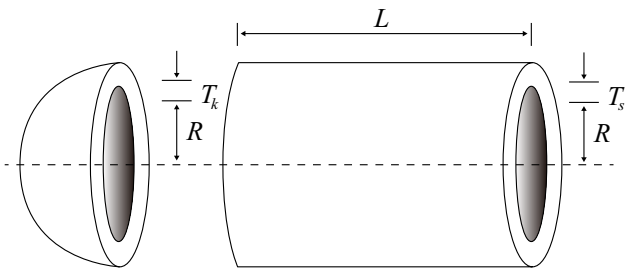


Fig. 11 Pressure vessel design problem

Table 8 Comparison results for pressure vessel design problem

	T_s	T_h	*R*	*L*	Optimum cost
EO	1.2584	0.0001	65.2010	10.1060	7591.1870
TLBO	0.8458	0.4181	43.8220	156.3900	6011.1980
WOA	1.1549	0.0000	59.7820	35.7220	7830.3550
GWO	1.2587	0.0000	65.2250	10.0310	7593.0700
KOA	0.7782	0.3847	40.3200	200.0000	5885.3420
DBO	0.8544	0.4051	43.6260	159.6100	6163.0690
MSDBO	0.7782	0.3847	40.3200	200.0000	5885.3360

algorithm has the lowest cost compared to the solutions obtained by other algorithms. This finding is significant for industries that rely on cylindrical containers to store and transport hazardous materials.

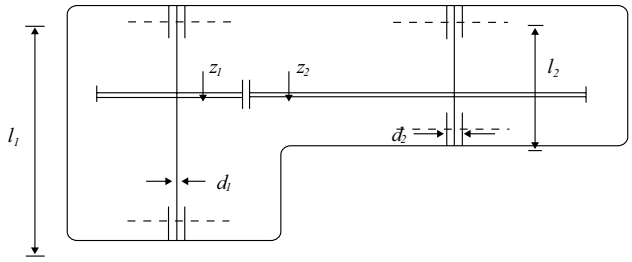


Fig. 12 Speed reducer design problem

Figure 14 (b) shows the convergence curve of the proposed MSDBO algorithm and other competitive algorithms. The plot illustrates that the MSDBO algorithm converges faster and achieves a better optimal solution than other algorithms. This result suggests that the proposed MSDBO algorithm performs competitively in solving the pressure vessel design problem.

5.4 Speed reducer design problem

The speed reducer design problem is a challenging benchmark problem in mechanical engineering, as it involves the optimization of seven design variables, as shown in Fig. 12 [62]. The problem's main objective is to minimize the weight of the reducer while satisfying various constraints, such as torque, bending strength, and contact stress [63]. The design variables to be optimized in this problem are the surface width (b), the tooth modulus (m), the number of teeth (p) on the pinion, the length (l_1) of the first shaft between bearings, the length (l_2) of the second shaft between bearings, the diameter (d_1) of the first shaft, and the diameter (d_2) of the second shaft.

The mathematical representation of the reducer design problem can be formulated as follows.

Consider:

$$\vec{z} = [z_1 z_2 z_3 z_4 z_5 z_6 z_7] = [bmpl_1 l_2 d_1 d_2] \tag{19}$$

Minimize:

$$\begin{aligned} f(\vec{z}) = {} & 0.7854 z_1 z_2^2 \left(3.3333 z_3^2 + 14.9334 z_3 - 43.0934\right) \\ & - 1.508 z_1 \left(z_6^2 + z_7^2\right) \\ & + 7.4777 \left(z_6^2 + z_7^2\right) + 0.7854 \left(z_4 z_6^2 + z_5 z_7^2\right) \end{aligned} \tag{20}$$

Subject to:

Table 9 Comparison results for speed reducer design problem

	b	m	z	l_1	l_2	d_1	d_2	Optimum cost
EO	3.4972	0.7000	17.0000	7.3253	7.7157	3.3506	5.2867	2994.7680
TLBO	3.5000	0.7000	17.0000	7.3000	7.7153	3.3505	5.2867	2994.4240
WOA	3.2860	0.7000	17.0000	8.2628	7.4952	3.3525	5.2866	3017.2030
GWO	3.4633	0.7000	17.0000	7.5213	8.2732	3.3511	5.2880	3010.9640
KOA	3.5000	0.7000	17.0000	7.3000	7.7153	3.3505	5.2867	2994.4240
DBO	3.5000	0.7000	17.0000	7.3000	7.7153	3.3505	5.2867	2994.4240
MSDBO	3.5000	0.7000	17.0000	7.3000	7.7153	3.3505	5.2867	2994.4240

$$
\begin{aligned}
g_1(\vec{z}) &= \frac{27}{z_1 z_2^2 z_3} - 1 \leqslant 0,\\
g_2(\vec{z}) &= \frac{397.5}{z_1 z_2^2 z_3^2} - 1 \leqslant 0,\\
g_3(\vec{z}) &= \frac{1.93 z_4^3}{z_2 z_6^4 z_3} - 1 \leqslant 0,\\
g_4(\vec{z}) &= \frac{1.93 z_5^3}{z_2 z_7^4 z_3} - 1 \leqslant 0,\\
g_5(\vec{z}) &= \frac{\left[\left(745\left(z_4/z_2 z_3\right)\right)^2 + 16.9 \times 10^6\right]^{1/2}}{110 z_6^3} - 1 \leqslant 0,\\
g_6(\vec{z}) &= \frac{\left[\left(745\left(z_5/z_2 z_3\right)\right)^2 + 157.5 \times 10^6\right]^{1/2}}{85 z_7^3} - 1 \leqslant 0,\\
g_7(\vec{z}) &= \frac{z_2 z_3}{40} - 1 \leqslant 0,\\
g_8(\vec{z}) &= \frac{5 z_2}{z_1} - 1 \leqslant 0,\\
g_9(\vec{z}) &= \frac{z_1}{12 z_2} - 1 \leqslant 0'\\
g_{10}(\vec{z}) &= \frac{1.5 z_6 + 1.9}{z_4} - 1 \leqslant 0,\\
g_{11}(\vec{z}) &= \frac{1.1 z_7 + 1.9}{z_5} - 1 \leqslant 0,
\end{aligned}
\tag{21}
$$

where, $2.6 \leqslant z_1 \leqslant 3.6$, $0.7 \leqslant z_z \leqslant 0.8$, $17 \leqslant z_3 \leqslant 28$, $7.3 \leqslant z_4 \leqslant 8.3$, $7.3 \leqslant z_5 \leqslant 8.3$, $2.9 \leqslant z_6 \leqslant 3.9$, $5.0 \leqslant z_7 \leqslant 5.5$.

Table 9 provides a comparison of the best solutions obtained with various optimization algorithms. The proposed MSDBO algorithm provides the best solution at z^*=(3.5000, 0.7000, 17.0000, 7.3000, 7.7153, 3.3505, 5.2687), where the corresponding fitness value is equal to $f(z^*)$=2994.4240. The results indicate that the MSDBO algorithm outperforms other competing algorithms. Figure 14 (c) shows that the MSDBO algorithm converges to the optimal solution of the reducer problem.

5.5 Tubular column design problem

The tubular column design problem is a critical engineering problem that aims to design a uniform tubular column with the lowest possible cost to handle a compression load of $P = 2500kgf$, as shown in Fig. 13. The optimization goal of the problem is to minimize the cost of the column while satisfying various constraints related to stress, buckling, and design variables.

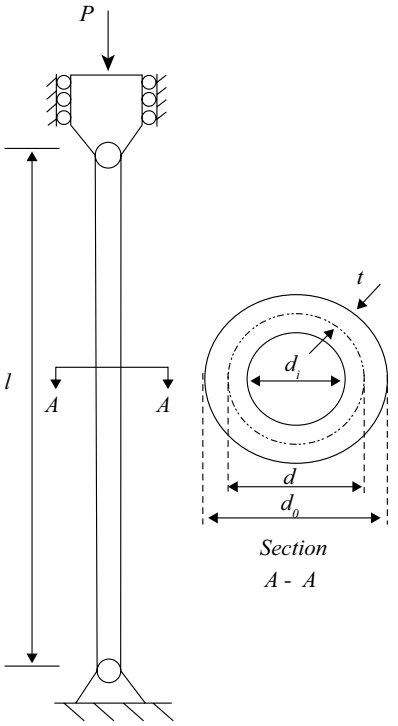
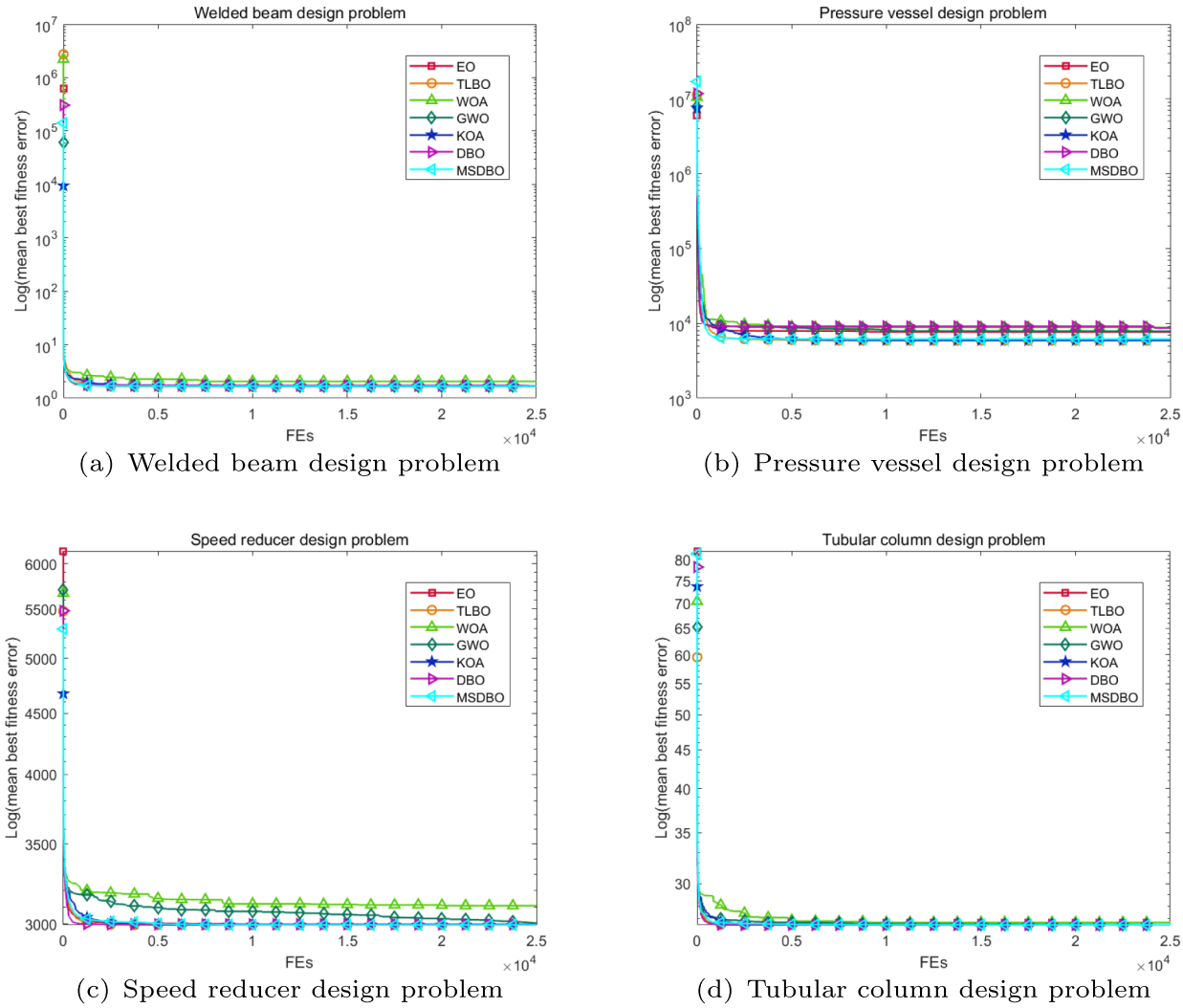


Fig. 13 Tubular column design problem

The material used in the column has an elastic modulus (E) of $0.85 \times 10^6 kgf/cm^2$, a yield stress ($\sigma_y$) of $500kgf/cm^2$, and a density (ρ) of $0.0025kgf/cm^2$. The length (L) of the column is fixed at 250cm. The stress in the column should be less than the yield stress (constraint g_1) and the buckling stress (constraint g_2). To ensure uniformity, the average diameter of the column is limited to 2 to 14cm (constraints g_3 and g_4), while thicknesses greater than 0.2 to 0.8cm are commercially unavailable (constraints g_5 and g_6).

The cost of the column is the sum of construction costs and material costs, and it is assumed to be the target function. The construction costs depend on the diameter and thickness of the column, while the material costs depend on the volume of the column. The target function needs to be minimized subject to the constraints described above.

Here is the optimization model of the problem.



(a) Welded beam design problem (b) Pressure vessel design problem

(c) Speed reducer design problem (d) Tubular column design problem

Fig. 14 Convergence curves of different algorithms on engineering design problems

Table 10 Comparison results for tubular column design problem

	d	*t*	Optimum Cost
EO	5.4522	0.2916	26.4864
TLBO	5.4522	0.2916	26.4864
WOA	5.4469	0.2925	26.5059
GWO	5.4519	0.2917	26.4879
KOA	5.4522	0.2916	26.4864
DBO	5.4522	0.2916	26.4864
MSDBO	5.4522	0.2916	26.4864

Consider:

$$\vec{z} = [z_1 z_2] = [d\ t] \tag{22}$$

Minimize:

$$f(\vec{z}) = 9.8 z_1 z_2 + 2 z_1 \tag{23}$$

Subject to:

$$\begin{aligned}
g_1(\vec{z}) &= \frac{P}{\pi z_1 z_2 \sigma_y} - 1 \leqslant 0, \\
g_2(\vec{z}) &= \frac{8PL^2}{\pi^3 E z_1 z_2 (z_1^2 + z_2^2)} - 1 \leqslant 0, \\
g_3(\vec{z}) &= \frac{2.0}{z_1} - 1 \leqslant 0, \\
g_4(\vec{z}) &= \frac{z_1}{14} - 1 \leqslant 0, \\
g_5(\vec{z}) &= \frac{0.2}{z_2} - 1 \leqslant 0, \\
g_6(\vec{z}) &= \frac{t}{0.8} - 1 \leqslant 0,
\end{aligned} \tag{24}$$

The performance comparison values and convergence curves of the EO, TLBO, WOA, GWO, KOA, DBO, and MSDBO algorithms for the tube column design problem

are shown in Fig. 14 (d), respectively. In addition, Table 10 provides the decision variables obtained from the best solution of each method across 30 runs. It can be seen that the MSDBO algorithm performs better than other algorithms at the optimal solution and can effectively solve the tube column design problem.

6 MSDBO for parameters identification of photovoltaic models

Due to energy crises, environmental pollution, and climate change, there has been a significant increase in the demand for alternative renewable energy sources [64]. Solar energy, as a secure and clean renewable energy source, has recently gained widespread attention. Solar energy is primarily applied in photovoltaic power generation, as photovoltaic systems can directly convert solar energy into electricity, and as a result, they have been adopted globally. However, due to the dependency on weather and environmental factors, particularly influenced by temperature and solar radiation, utilizing solar photovoltaic systems for power generation has become a significant challenge. In order to enhance the efficiency of photovoltaic power generation, it's necessary to extract accurate photovoltaic model parameters based on measured voltage-current data. Consequently, methods for efficiently extracting PV model parameters have become particularly crucial. Currently, the predominant approaches for solving the extraction of PV model parameters are analytical methods and deterministic methods. Analytical methods are straightforward to implement and can quickly provide solutions to problems, but they require certain assumptions that might lead to inaccuracies in the extracted parameters. Deterministic methods, such as the Newton–Raphson method [65], are highly sensitive to initial values. Moreover, deterministic methods impose strict requirements on the objective function, it must be continuous, differentiable, and convex.

In recent years, optimization algorithms have garnered increasing attention. Many researchers have attempted to use intelligent optimization algorithms for extracting parameters of photovoltaic (PV) models, such as Genetic Algorithms (GA), Whale Optimization Algorithm (WOA), Harris Hawks Optimization (HHO), and Slap Swarm Algorithm (SSA). While these endeavors have yielded promising outcomes, the effectiveness of the aforementioned algorithms is influenced by their algorithm-specific parameters or those introduced during their application. The challenge of determining optimal parameter values tailored to a specific or novel optimization problem remains, as improper parameter tuning can lead to escalated computational demands or convergence to local optima. This intricacy underscores the need for methodologies that offer automated parameter adaptation to obviate these pitfalls. Additionally, these algorithms typically require substantial computational resources for parameter extraction in PV models.

Therefore, the MSDBO is proposed, which performs parameter extraction on various PV models and is compared with results obtained from existing intelligent optimization algorithms. Experimental outcomes indicate that the proposed algorithm exhibits significant superiority in the parameter extraction of PV models.

6.1 Problem formulation

In existing literature, various photovoltaic (PV) models have been proposed to elucidate the current–voltage behaviors of solar cells and PV modules. In practical applications, the single diode model and the double diode model stand out as the most frequently employed options. This section outlines these models along with their corresponding objective functions.

6.1.1 Single diode model

According to the equivalent circuit of the single diode model, as shown in Fig. 15 (a), its output current I as Eq. (25).

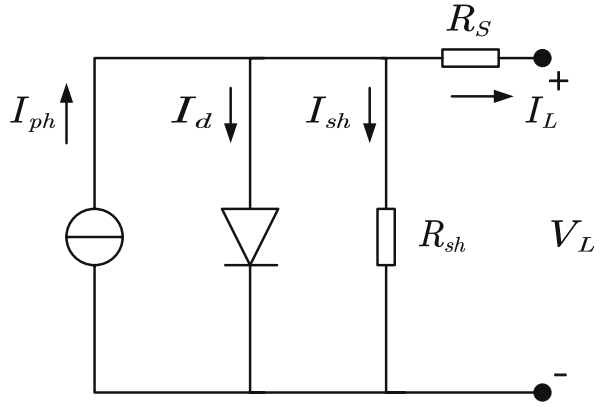
$$I = I_{ph} - I_{sd} \times \left[\exp\left(\frac{V_L + I_L \times R_S}{nV_t}\right) - 1\right] - \frac{V_L + I_L \times R_S}{R_{sh}} \tag{25}$$

where I_{ph} is the photocurrent generated by actual illumination, in Amperes (A), and I_{sd} is the reverse saturation current of the diode, in microamperes ($?A$). R_S and R_{sh} are the equivalent series and shunt resistances of the photovoltaic panel, and n is the diode ideality factor. $V_t = kT/q$ where k is the Boltzmann constant ($1.38 \times 10^{-23} J/K$), and q is the elementary charge of an electron ($1.6 \times 10^{-19} C$). T is the ambient temperature in Kelvin (K).

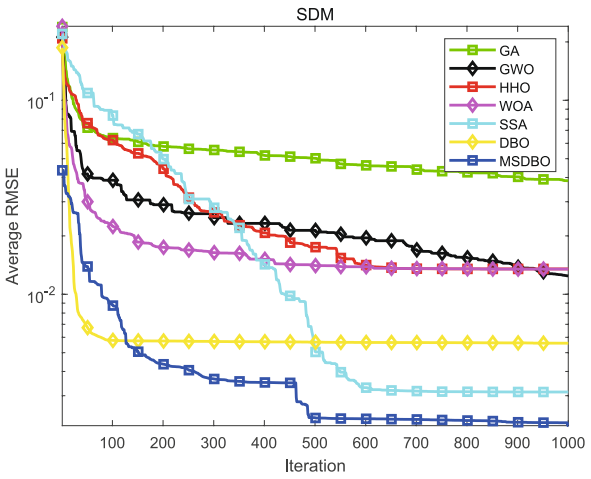
6.1.2 Double diode model

According to the equivalent circuit of the double diode model, as shown in Fig. 15 (c), its output current I as Eq. (26).

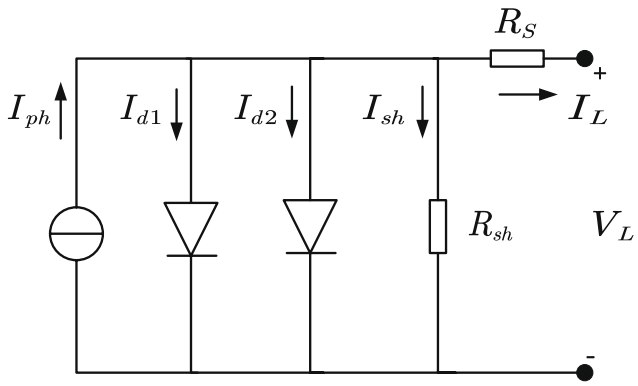
$$\begin{aligned} I = I_{ph} &- I_{sd1} \times \left[\exp\left(\frac{V_L + I_L \times R_S}{n_1 \times V_t}\right) - 1\right] \\ &- I_{sd2} \times \left[\exp\left(\frac{V_L + I_L \times R_S}{n_2 V_t} - 1\right) - \frac{V + I_L \times R_S}{R_{sh}}\right] \end{aligned} \tag{26}$$



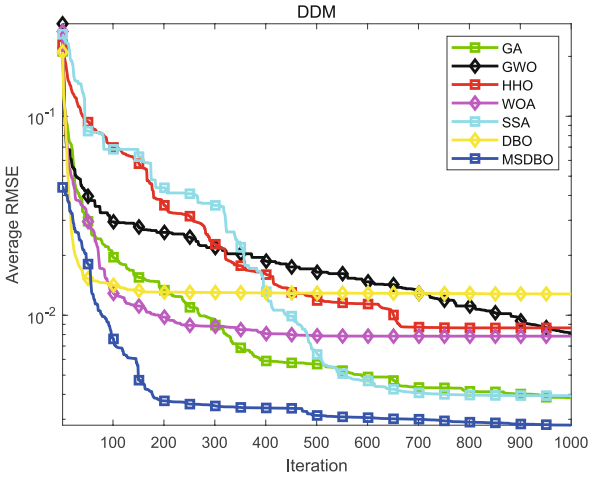
(a) Schematics for single diode.



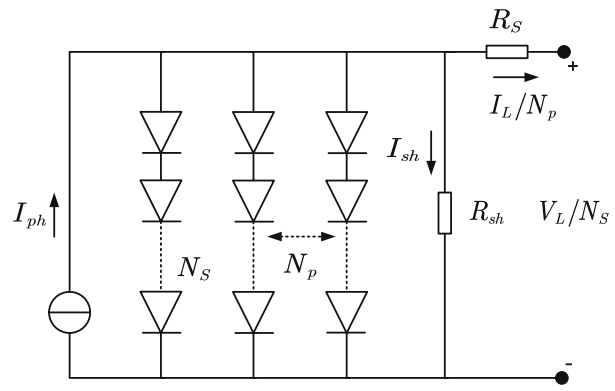
(b) Convergence curves of SDM.



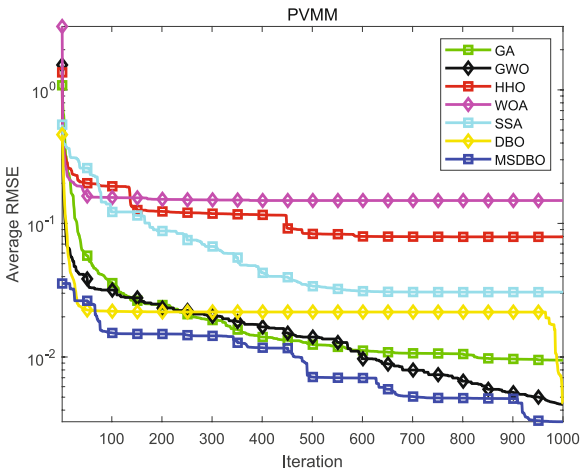
(c) Schematics for double diode.



(d) Convergence curves of DDM.



(e) Schematics for PV module.



(f) Convergence cueves of PVMM.

Fig. 15 Schematics and Convergence profiles of PV models

Table 11 Parameters range for the single and double diode models, and PV module model

Paramter	Single diode/double diobe		PV module	
	Lower bound	Upper bound	Lower bound	Upper bound
$I_{ph}\,(A)$	0	1	0	1
$I_{sd}, I_{sd1}, I_{sd2}\,(\mu A)$	0	1	0	50
$R_S\,(\Omega)$	0	0.5	0	2
$R_{sh}\,(\Omega)$	0	100	0	2000
n, n_1, n_2	1	2	1	50

where I_{sd1} and I_{sd2} are the reverse saturation currents of the two diodes, in microamperes (μA), and n_1 and n_2 are the ideality factors of the two diodes, respectively.

6.1.3 PV module model

In Fig. 15 (e), the single diode PV module model with interconnected solar cells is depicted. The output current expression is as follows.

$$I_L/N_p = I_{ph} - I_{sd} \times \left[\exp\left(\frac{V_L/N_S + I_L \times R_S/N_p}{V_t}\right) - 1\right] - \frac{V_L/N_S + I_L \times R_S/N_p}{R_{sh}} \tag{27}$$

where N_p denotes the number of solar cells in parallel, and N_S indicates the number of solar cells in series.

6.1.4 Objective function

Directly solving for the parameters of the PV model is extremely challenging. In reference [66], the parameter extraction problem is transformed into a nonlinear optimization problem, and an optimization objective function can be formulated as shown in Eq. (28).

$$RMSE = \sqrt{\frac{1}{N}\sum_{k=1}^{N} f\left(V_k, I_k, \vec{x}\right)} \tag{28}$$

where $\vec{x}$ contains the parameters to be extracted for each model, and N is the number of V-I data points. $f\left(V_k, I_k, \vec{x}\right) = I_{sim} - I_k$, which represents the difference between the simulated current I_{sim} and the experimental current I_k. Root Mean Square Error (RMSE) is employed as an evaluation metric. RMSE signifies the sum of squared differences between all simulated and experimental currents. A smaller RMSE indicates more accurate extracted parameters.

6.2 Experimental results and analysis

The data for this evaluation is obtained from a source referenced as [65]. Specifically, the data comes from a 57*mm* diameter commercial RTC France silicon solar cell operating under a condition of $1000W/m^2$ at $33°C$, and a solar module known as Photowatt-PWP201 composed of 36 polycrystalline silicon cells in series, operating under $1000W/m^2$ at $45°C$. These datasets are considered benchmark experimental data for testing techniques designed to extract parameters from PV models. Previous studies [67, 68] have also utilized this dataset for similar purposes. To ensure a fair comparison and consistency with prior research, the section establishes lower and upper bounds for each parameter involved in the PV models. These bounds are presented in Table 11 and are kept consistent with the bounds used in previous literature.

To demonstrate the effectiveness of the MSDBO algorithm, it is compared against several advanced algorithms, namely GA, GWO, HHO, WOA, SSA, and DBO. To ensure a fair and consistent evaluation, all of these algorithms are subjected to the same conditions. Each algorithm is set to run for a maximum of 1000 iterations in every individual attempt to solve a given problem. Furthermore, to minimize the impact of statistical variations, each algorithm is independently tested 30 times for each specific problem.

6.2.1 Results on the single diode model

The outcomes of the comparison for the single diode model (SDM) are showcased through Table 12, featuring the estimated parameters and corresponding Root Mean Square Error (RMSE) values. Notably, the most superior RMSE value achieved among all the algorithms under

Table 12 Comparison results for the single diode model

	$I_{ph}\,(A)$	$I_{sd}\,(\mu A)$	$R_S\,(\Omega)$	$R_{sh}\,(\Omega)$	n	RMSE
GA	0.7635	5.6924e−07	0.0345	65.4173	1.5399	2.5998e−03
GWO	0.7599	3.4139e−07	0.0360	65.4591	1.4866	1.1653e−03
HHO	0.7592	3.0364e−07	0.0357	64.3013	1.4749	2.1872e−03
WOA	0.7606	3.8740e−07	0.0356	59.9890	1.4996	1.0479e−03
SSA	0.7593	2.7749e−07	0.0373	76.1971	1.4656	1.4819e−03
DBO	0.7607	3.7671e−07	0.0357	57.4455	1.4968	1.0324e−03
MSDBO	0.6811	6.9850e−07	0.4856	37.8139	1.1636	**2.3112e−06**

Table 13 Comparison results for the double diode model

	$I_{ph}\,(A)$	$I_{sd1}\,(\mu A)$	$R_S\,(\Omega)$	$R_{sh}\,(\Omega)$	n_1	$I_{sd2}\,(\mu A)$	n_2	RMSE
GA	0.7625	3.98e−07	0.0327	51.3501	1.7300	4.2198e−07	1.5260	2.31e−03
GWO	0.7590	2.15e−07	0.0384	74.1039	1.5089	4.7737e−08	1.3684	1.83e−03
HHO	0.7593	2.10e−07	0.0376	76.1159	1.5012	7.9409e−08	1.4190	1.53e−03
WOA	0.7597	2.38e−07	0.0367	61.2422	1.4553	4.9341e−08	1.6241	1.63e−03
SSA	0.7609	1.32e−07	0.0345	60.8730	1.5819	3.7477e−07	1.5146	1.31e−03
DBO	0.7595	4.73e−07	0.0350	100.0000	1.5204	1.0000e−09	2.0000	1.44e−03
MSDBO	0.7608	2.07e−07	0.0369	53.5201	1.4444	5.0779e−07	1.8855	**9.90e−04**

Table 14 Comparison results for the PV module model

	$I_{ph}\,(A)$	$I_{sd}\,(\mu A)$	$R_S\,(\Omega)$	$R_{sh}\,(\Omega)$	n	RMSE
GA	1.0299	4.8487e−06	1.1559	1041.0857	49.9664	2.9611e−03
GWO	1.0284	4.9178e−06	1.1647	1613.0557	50.0000	2.6107e−03
HHO	1.0214	1.7328e−07	1.5675	1134.2689	39.2633	1.4785e−02
WOA	1.0304	3.1456e−06	1.2202	983.2622	48.2512	2.5532e−03
SSA	1.0328	2.9193e−06	1.2163	728.9405	47.9830	2.5082e−03
DBO	1.0288	4.4190e−06	1.1764	1399.1755	49.5707	2.5158e−03
MSDBO	1.0307	3.3672e−06	1.2045	941.5590	48.5150	**2.4269e−03**

Table 15 Statistical results of RMSE of different algorithms for three models

Model	Algorithm	RMSE			
		Min	Mean	Max	Std
SDM	GA	2.5998e−03	3.8461e−02	1.1806e−01	4.9012e−02
	GWO	1.1653e−03	1.2437e−02	3.8156e−02	1.4434e−02
	HHO	2.1872e−03	1.3513e−02	4.5514e−02	1.3915e−02
	WOA	1.0479e−03	1.3489e−02	4.4712e−02	1.6206e−02
	SSA	1.4819e−03	3.1347e−03	**6.1010e−03**	1.6771e−03
	DBO	1.0324e−03	5.5913e−03	3.8151e−02	1.1454e−02
	MSDBO	**2.3112e−06**	**2.1024e−03**	6.7964e−03	2.1680e−03
DDM	GA	2.3119e−03	3.8470e−03	5.3167e−03	9.3030e−04
	GWO	1.8358e−03	8.1431e−03	3.9392e−02	1.1535e−02
	HHO	1.5347e−03	8.6396e−03	2.7025e−02	7.2732e−03
	WOA	1.6377e−03	7.8489e−03	4.0782e−02	1.2286e−02
	SSA	1.3186e−03	3.9442e−03	9.3486e−03	2.4293e−03
	DBO	1.4436e−03	1.2782e−02	3.9488e−02	1.7410e−02
	MSDBO	**9.9021e−04**	**2.8008e−03**	**1.1560e−02**	3.1517e−03
PVM	GA	2.9611e−03	9.4646e−03	3.2894e−02	8.6238e−03
	GWO	2.6107e−03	4.4095e−03	1.0193e−02	2.3930e−03
	HHO	1.4785e−02	7.9353e−02	2.7970e−01	8.0732e−02
	WOA	2.5532e−03	1.4847e−01	2.7570e−01	1.3564e−01
	SSA	2.5082e−03	3.0600e−02	2.8130e−01	8.8086e−02
	DBO	2.5158e−03	4.5269e−03	1.6719e−02	4.4266e−03
	MSDBO	**2.4269e−03**	**3.2754e−03**	**7.7310e−03**	1.6327e−03

consideration is emphasized in bold text. As depicted in Table 12, the results demonstrate that MSDBO attains the lowest RMSE value (2.3112e−06) when juxtaposed with the selected algorithms. Following MSDBO in descending order are SSA, DBO, GWO, WOA, HHO, and GA, each exhibiting progressively higher RMSE values.

To reinforce the credibility of the outcomes, the convergence profiles, as illustrated in Fig. 15 (b), delineate the mean performance of RMSE across the 30 autonomous runs. Evidently, these graphs affirm that MSDBO boasts a swifter rate of convergence in comparison to the other algorithms under scrutiny.

6.2.2 Results on the double diode model

In the context of the double diode model (DDM), the task involves identifying seven distinct parameters. The estimated parameter values alongside their corresponding Root Mean Square Error (RMSE) metrics, derived from various algorithms, are itemized in Table 13. Notably, MSDBO once again emerges as the frontrunner, securing the most favorable RMSE value (9.9021e−04) within the spectrum of considered algorithms. Following suit, SSA captures the second most advantageous RMSE value. The convergence curve of MSDBO and its comparison algorithms is shown in Fig. 15 (d), indicating that MSDBO performs competitively.

6.2.3 Results on the PV module model

In the context of the PV module model (PVMM), the endeavor entails identifying five specific parameters. The ascertained parameter values alongside their respective Root Mean Square Error (RMSE) measures are showcased through Table 14. Evidently, among the spectrum of algorithms considered, MSDBO emerges triumphant yet again, securing the most favorable RMSE value (0.0024269). In the ranking that ensues, SSA follows suit with the second best RMSE value, closely trailed by DBO in third place. The validation of MSDBO's superiority is further substantiated by the convergence curve depicted in Fig. 15 (f).

The aggregated statistical outcomes encompassing all compared algorithms across 30 independent iterations are succinctly presented in Table 15. The Mean RMSE encapsulates the average precision, while SD elucidates the standard deviation of RMSE, thereby illuminating the robustness of parameter estimation. In the case of each model, the preeminent outcomes amidst the seven algorithms is distinctly highlighted.

Upon analyzing the amalgamated data, it becomes apparent that MSDBO consistently outperforms all alternative algorithms across the single and double diode models. Furthermore, in the case of the PV module model, MSDBO maintains a competitive position, achieving the best results in terms of both mean RMSE and standard deviation. This highlights its strong performance in precision and reliability.

7 Conclusion and future works

In this paper, a multi-strategy boosted dung beetle optimizer is proposed, which combines dynamic population size variation, forced ranking, and a novel boundary control strategy. In MSDBO, the dynamic population is introduced to obtain a balance between exploration and exploitation, improving the convergence speed of the algorithm. Then, a forced ranking strategy is used to improve the quality of the population avoid search stagnation of the optimal solution during optimization. In addition, a novel boundary control strategy is introduced into the updating process to better apply the information of excellent individuals and increase the convergence speed of the algorithm. The performance of MSDBO is validated on the CEC 2014 and CEC 2019 test suites. The experimental results show that MSDBO can obtain the minimum average rank (rank=3.82) of the CEC 2014 suite and obtain the minimum average rank (rank=3.85) of the CEC 2019 suite. The experiments show that MSDBO has better convergence performance and global search capability for most tested problems. In addition, the engineering optimization performance of MSDBO is validated on eight specific engineering design cases (including four functions from CEC 2011 and four real-world engineering problems) and the Parameters identification of photovoltaic models. From the experiments, first, it is clear that MSDBO obtains the best average value among the 4 engineering problems of the engineering optimization suite. Second, the proposed MSDBO algorithm demonstrates competitive performance in the parameter identification of photovoltaic models. The experiments show the applicability and potential of MSDBO in solving different types of applications.

However, while the proposed MSDBO algorithm demonstrates competitive performance in solving a variety of optimization problems, it has several limitations that need to be addressed to further enhance its effectiveness and applicability. The key limitations include:

(1) Sensitivity to parameter tuning, as the optimal values for parameters such as p, p_1, and p_2 may vary significantly across different problems, requiring manual adjustment for each scenario.
(2) Room for improvement in convergence speed and solution accuracy, particularly for highly complex or multimodal problems where the algorithm may struggle to find global optima efficiently.
(3) Limited applicability to higher-dimensional and discrete optimization problems, as the current implementation is primarily designed for continuous optimization and may not handle discrete variables or high-dimensional search spaces effectively.
(4) Challenges in scalability and computational efficiency, as the algorithm's performance may degrade when applied to large-scale problems with high computational demands.

To address these limitations, future work should focus on developing adaptive parameter tuning mechanisms, exploring hybrid approaches to enhance convergence, extending MSDBO to handle higher-dimensional and discrete

problems, improving scalability through parallelization or surrogate modeling, and validating its performance on a broader range of real-world applications. These efforts will significantly enhance the capabilities of MSDBO, making it a more versatile and powerful tool for solving complex optimization problems and expanding its practical impact in addressing real-world challenges.

Acknowledgements This work is supported by the National Natural Science Foundation of China (62006144), the major fundamental research project of Shandong, China (No. ZR2019ZD03), the Taishan Scholar Project of Shandong, China (No. ts20190924).

Author Contributions X.L. and Q.Z.: Conceptualization, Methodology, Software, Investigation, Writing – Original Draft. Q.Z., J.L., and H.Z.: Writing – Review & Editing. Q.Z.: Supervision, Project Administration.

Data Availability Data will be made available on request.

Declarations

Conflict of interest The authors declare no Conflict of interest.

References

1. Pierezan J, Coelho LDS (2018) Coyote optimization algorithm: a new metaheuristic for global optimization problems. In: 2018 IEEE congress on evolutionary computation (CEC), pp. 1–8 . IEEE
2. Zhan Z-H, Shi L, Tan KC, Zhang J (2022) A survey on evolutionary computation for complex continuous optimization. Artif Intell Rev 55:1–52
3. Mahdavi S, Shiri ME, Rahnamayan S (2015) Metaheuristics in large-scale global continues optimization: a survey. Inf Sci 295:407–428. https://doi.org/10.1016/j.ins.2014.10.042
4. Xiong G, Zhang J, Shi D, He Y (2018) Parameter extraction of solar photovoltaic models using an improved whale optimization algorithm. Energy Convers Manage 174:388–405. https://doi.org/10.1016/j.enconman.2018.08.053
5. Molina D, Poyatos J, Ser JD, García S, Hussain A, Herrera F (2020) Comprehensive taxonomies of nature-and bio-inspired optimization: inspiration versus algorithmic behavior, critical analysis recommendations. Cogn Comput 12:897–939
6. Singh S, Singh H, Mittal N, Hussien AG, Sroubek F (2022) A feature level image fusion for night-vision context enhancement using arithmetic optimization algorithm based image segmentation. Expert Syst Appl 209:118272
7. Blanco-Montenegro I, De Ritis R, Chiappini M (2007) Imaging and modelling the subsurface structure of volcanic calderas with high-resolution aeromagnetic data at vulcano (aeolian islands, italy). Bull Volcanol 69:643–659
8. Fathi H, AlSalman H, Gumaei A, Manhrawy II, Hussien AG, El-Kafrawy P, et al (2021) An efficient cancer classification model using microarray and high-dimensional data. Comput Intell Neurosci**2021**
9. Hussien AG, Amin M (2022) A self-adaptive Harris Hawks optimization algorithm with opposition-based learning and chaotic local search strategy for global optimization and feature selection. Int J Mach Learn Cybern 13(2):309–336. https://doi.org/10.1007/s13042-021-01326-4
10. Aguiar MJR, Alves TDR, Honório LM, Junior IC, Vidal VF (2021) Performance evaluation of bundle adjustment with population based optimization algorithms applied to panoramic image stitching. Sensors 21(15):5054
11. Sahoo SK, Houssein EH, Premkumar M, Saha AK, Emam MM (2023) Self-adaptive moth flame optimizer combined with crossover operator and fibonacci search strategy for covid-19 ct image segmentation. Expert Syst Appl 227:120367
12. Sahoo SK, Saha AK, Nama S, Masdari M (2023) An improved moth flame optimization algorithm based on modified dynamic opposite learning strategy. Artif Intell Rev 56(4):2811–2869
13. Sahoo SK, Saha AK (2022) A hybrid moth flame optimization algorithm for global optimization. J Bionic Eng 19(5):1522–1543
14. Sahoo SK, Reang S, Saha AK, Chakraborty S (2024) F-woa: an improved whale optimization algorithm based on fibonacci search principle for global optimization. Handbook of whale optimization algorithm. Elsevier, pp 217–233
15. Sahoo SK, Saha AK, Houssein EH, Premkumar M, Reang S, Emam MM (2024) An arithmetic and geometric mean-based multi-objective moth-flame optimization algorithm. Clust Comput 27:1–35
16. Sahoo SK, Premkumar M, Saha AK, Houssein EH, Wanjari S, Emam MM (2024) Multi-objective quasi-reflection learning and weight strategy-based moth flame optimization algorithm. Neural Comput Appl 36(8):4229–4261
17. Sahoo SK, Sharma S, Saha AK (2023) A novel variant of moth flame optimizer for higher dimensional optimization problems. J Bionic Eng 20(5):2389–2415
18. Sahoo SK, Saha AK, Sharma S, Mirjalili S, Chakraborty S (2022) An enhanced moth flame optimization with mutualism scheme for function optimization. Soft Computing, 1–28
19. Zhan Z-H, Li J-Y, Zhang J (2022) Evolutionary deep learning: a survey. Neurocomputing 483:42–58
20. Xue J, Shen B (2023) Dung beetle optimizer: a new meta-heuristic algorithm for global optimization. J Supercomput 79(7):7305–7336. https://doi.org/10.1007/s11227-022-04959-6
21. Adegboye OR, Feda AK (2025) Improved exponential distribution optimizer: enhancing global numerical optimization problem solving and optimizing machine learning paramseters. Clust Comput 28(2):128
22. Adegboye OR, Feda AK, Ojekemi OS, Agyekum EB, Elattar EE, Kamel S (2024) Refinement of dynamic hunting leadership algorithm for enhanced numerical optimization. IEEE Access
23. Adegboye OR, Ülker ED, Feda AK, Agyekum EB, Mbasso WF, Kamel S (2024) Enhanced multi-layer perceptron for co2 emission prediction with worst moth disrupted moth fly optimization (wmfo). Heliyon **10**(11)
24. Adegboye OR, Deniz Ülker E (2023) Hybrid artificial electric field employing cuckoo search algorithm with refraction learning for engineering optimization problems. Sci Rep 13(1):4098
25. Liang JJ, Qu BY, Suganthan PN (2013) Problem definitions and evaluation criteria for the cec 2014 special session and competition on single objective real-parameter numerical optimization. Computational Intelligence Laboratory, Zhengzhou University, Zhengzhou China and Technical Report, Nanyang Technological University, Singapore **635**(2)
26. Price KV, Awad NH, Ali MZ, Suganthan, P.N.: The, (2019) 100-digit challenge on real-parameter. Analysis of results, single objective optimization
27. Li J, Li J, Gao K, Duan P (2024) A hybrid graph-based imitation learning method for a realistic distributed hybrid flow shop

with family setup time. IEEE Transactions on Systems, Man, and Cybernetics: Systems
28. Holland JH (1992) Genetic algorithms. Sci Am 267(1):66–73
29. Price KV (1996) Differential evolution: a fast and simple numerical optimizer. In: Proceedings of North American fuzzy information processing, pp. 524–527 . IEEE
30. Kennedy J, Eberhart R (1995) Particle swarm optimization. In: Proceedings of ICNN'95-international Conference on Neural Networks, vol. 4, pp. 1942–1948 . ieee
31. Karaboga D, Basturk B (2007) A powerful and efficient algorithm for numerical function optimization: artificial bee colony (ABC) algorithm. J Global Optim 39(3):459–471. https://doi.org/10.1007/s10898-007-9149-x
32. Mirjalili S, Lewis A (2016) The whale optimization algorithm. Adv Eng Softw 95:51–67. https://doi.org/10.1016/j.advengsoft.2016.01.008
33. Hamadneh T, Batiha B, Werner F, Montazeri Z, Dehghani M, Bektemyssova G, Eguchi K (2024) Fossa optimization algorithm: a new bio-inspired metaheuristic algorithm for engineering applications. Int J Intell Eng Syst 17:1038–1045
34. Hamadneh T, Kaabneh K, Alssayed O, Eguchi K, Gochhait S, Leonova I, Dehghani M(2024) Addax optimization algorithm: a novel nature-inspired optimizer for solving engineering applications. Int J Intell Eng Syst,**17**(3)
35. Jia H, Peng X, Lang C (2021) Remora optimization algorithm. Expert Syst Appl 185:115665. https://doi.org/10.1016/j.eswa.2021.115665
36. 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
37. 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
38. Abualigah L, Elaziz MA, Sumari P, Geem ZW, Gandomi AH (2022) Reptile Search Algorithm (RSA): a nature-inspired meta-heuristic optimizer. Expert Syst Appl 191:116158. https://doi.org/10.1016/j.eswa.2021.116158
39. Meraihi Y, Ramdane-Cherif A, Acheli D, Mahseur M (2020) Dragonfly algorithm: a comprehensive review and applications. Neural Comput Appl 32(21):16625–16646
40. Dehghani M, Montazeri Z, Trojovská E, Trojovskỳ P (2023) Coati optimization algorithm: a new bio-inspired metaheuristic algorithm for solving optimization problems. Knowl-Based Syst 259:110011
41. Bertsimas D, Tsitsiklis J (1993) Simulated annealing. Stat Sci 8(1):10–15
42. Dehghani M, Samet H (2020) Momentum search algorithm: a new meta-heuristic optimization algorithm inspired by momentum conservation law. SN Appl Sci 2(10):1720
43. Dehghani M, Montazeri Z, Dehghani A, Seifi A (2017) Spring search algorithm: A new meta-heuristic optimization algorithm inspired by hooke's law. In: 2017 IEEE 4th international conference on knowledge-based engineering and innovation (KBEI), pp. 0210–0214 . IEEE
44. Rashedi E, Nezamabadi-Pour H, Saryazdi S (2009) Gsa: a gravitational search algorithm. Inf Sci 179(13):2232–2248
45. Goodarzimehr V, Shojaee S, Hamzehei-Javaran S, Talatahari S (2022) Special relativity search: a novel metaheuristic method based on special relativity physics. Knowl-Based Syst 257:109484
46. Sadollah A, Eskandar H, Bahreininejad A, Kim JH (2015) Water cycle algorithm for solving multi-objective optimization problems. Soft Comput 19:2587–2603
47. Rao RV, Savsani VJ, Vakharia DP (2011) Teaching-learning-based optimization: a novel method for constrained mechanical design optimization problems. Comput Aided Des 43(3):303–315
48. Zhang Q, Gao H, Zhan Z-H, Li J, Zhang H (2023) Growth optimizer: a powerful metaheuristic algorithm for solving continuous and discrete global optimization problems. Knowl-Based Syst 261:110206
49. Hamadneh T, Batiha B, Alsayyed O, Bektemyssova G, Montazeri Z, Dehghani M, Eguchi K (2024) On the application of potter optimization algorithm for solving supply chain management application. Int J Intell Eng Syst**17**(5)
50. Alomari S, Kaabneh K, AbuFalahah I, Gochhait S, Leonova I, Montazeri Z, Dehghani M, Eguchi K (2024) Carpet weaver optimization: A novel simple and effective human-inspired metaheuristic algorithm. Int J Intell Eng Syst**17**(4)
51. Kaabneh K, AbuFalahah I, Eguchi K, Gochhait S, Leonova I, Montazeri Z, Dehghani M, et al (2024) Dollmaker optimization algorithm: a novel human-inspired optimizer for solving optimization problems. Int J Intell Eng Syst**17**(3)
52. Moosavi SHS, Bardsiri VK (2019) Poor and rich optimization algorithm: a new human-based and multi populations algorithm. Eng Appl Artif Intell 86:165–181
53. Hamadneh T, Kaabneh K, AlSayed O, Bektemyssova G, Montazeri Z, Dehghani M, Eguchi K (2024) Sculptor optimization algorithm: a new human-inspired metaheuristic algorithm for solving optimization problems. Int J Intell Eng Syst**17**(4)
54. Dacke M, Baird E, El Jundi B, Warrant EJ, Byrne M (2021) How dung beetles steer straight. Annu Rev Entomol 66:243–256
55. Yin Z, Zinn-Björkman L (2020) Simulating rolling paths and reorientation behavior of ball-rolling dung beetles. J Theor Biol 486:110106
56. Yan S, Yang P, Zhu D, Zheng W, Wu F., et al (2021) Improved sparrow search algorithm based on iterative local search. Comput Intell Neurosci**2021**
57. Das S, Suganthan PN (2010) Problem definitions and evaluation criteria for cec 2011 competition on testing evolutionary algorithms on real world optimization problems. Jadavpur University, Nanyang Technological University, Kolkata, 341–359
58. Smith AE, Coit DW, Baeck T, Fogel D, Michalewicz Z (1997) Penalty functions Handbook of evolutionary computation 97(1), 5
59. Faramarzi A, Heidarinejad M, Stephens B, Mirjalili S (2020) Equilibrium optimizer: a novel optimization algorithm. Knowl-Based Syst 191:105190. https://doi.org/10.1016/j.knosys.2019.105190
60. Rao RV, Savsani VJ, Vakharia DP (2011) Teaching–learning-based optimization: a novel method for constrained mechanical design optimization problems. Comput Aided Des 43(3):303–315. https://doi.org/10.1016/j.cad.2010.12.015
61. Abdel-Basset M, Mohamed R, Azeem SAA, Jameel M, Abouhawwash M (2023). Kepler optimization algorithm A new metaheuristic algorithm inspired by Kepler's laws of planetary motion. https://doi.org/10.1016/j.knosys.2023.110454
62. Gandomi AH, Yang X-S (2011) Benchmark problems in structural optimization. Comput Optim Methods Algorithms 356:259–281
63. Mezura-Montes E, Coello CAC (2005) Useful infeasible solutions in engineering optimization with evolutionary algorithms. In: MICAI 2005: Advances in Artificial Intelligence: 4th Mexican International Conference on Artificial Intelligence, Monterrey, Mexico, November 14-18, 2005. Proceedings 4, pp. 652–662 . Springer
64. Muhsen DH, Ghazali AB, Khatib T, Abed IA (2015) Parameters extraction of double diode photovoltaic module s model based on hybrid evolutionary algorithm. Energy Convers Manage 105:552–561

65. Easwarakhanthan T, Bottin J, Bouhouch I, Boutrit C (1986) Nonlinear minimization algorithm for determining the solar cell parameters with microcomputers. Int J Solar Energy 4(1):1–12
66. Jiang LL, Maskell DL, Patra JC (2013) Parameter estimation of solar cells and modules using an improved adaptive differential evolution algorithm. Appl Energy 112:185–193
67. Chen Z, Wu L, Lin P, Wu Y, Cheng S (2016) Parameters identification of photovoltaic models using hybrid adaptive nelder-mead simplex algorithm based on eagle strategy. Appl Energy 182:47–57
68. Niu Q, Zhang H, Li K (2014) An improved tlbo with elite strategy for parameters identification of pem fuel cell and solar cell models. Int J Hydrogen Energy 39(8):3837–3854