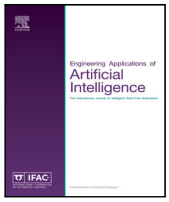
Engineering Applications of Artificial Intelligence 137 (2024) 109202

Contents lists available at ScienceDirect

Engineering Applications of Artificial Intelligence

journal homepage: www.elsevier.com/locate/engappai



Research paper

# Alpha evolution: An efficient evolutionary algorithm with evolution path adaptation and matrix generation☆

Hao Gao, Qingke Zhang *

*School of Information Science and Engineering, Shandong Normal University, Jinan 250358, China*

ARTICLE INFO

*Keywords:*
Evolutionary computation
Non-metaphor algorithm
Alpha operator
Numerical optimization
Real-world applications

ABSTRACT

Metaheuristics involve information extraction and utilization processes to generate more promising solutions. However, the background of excessive metaphor has led to ambiguity in the computational process. To solve this problem, this paper proposes a novel evolutionary algorithm called alpha evolution (AE). It updates the solution using the alpha operator with the adaptive base vector and the random and adaptive step sizes. First, sample candidate solutions to construct the evolution matrix. Estimate the population state through diagonal or weighted operations of the evolution matrix. To enhance the correlation of estimates for each generation, two evolution paths accumulate the estimate results and achieve the base vector adaptation. Second, the composite differential operation constructs the adaptive step size to estimate the problem gradient, which is used to accelerate the convergence of AE. Finally, the attenuation factor alpha adaptively adjusts the random step size generated based on search space to balance exploration and exploitation. AE was comprehensively verified regarding its search bias, invariance, scalability, parameter sensitivity, search behavior, qualitative indicators, exploration and exploitation, convergence, statistics, and complexity. In numerical simulation, AE was compared with 106 algorithms on the CEC'17 benchmark announced at the 2017 congress on evolutionary computation (CEC). Furthermore, AE was applied to solve multiple sequence alignment and engineering design problems. The evidence shows that AE is competitive in exploration and exploitation, convergence speed and accuracy, avoiding local optima, applicability, and reliability. The source code of AE is publicly available at https://github.com/tsingke/AlphaEvolution.

## 1. Introduction

In the real world, there are many highly nonconvex, nonlinear, and multimodal optimization problems (Ma et al., 2023). Exact algorithms require traversing all possible solutions in the problem space effectively and guaranteeing the discovery of the globally optimal solution, but this may result in expensive time costs (Rajwar et al., 2023; Khanduja and Bhushan, 2021). However, intelligence optimization algorithms often generate approximately optimal solutions that meet the requirements within a specified time, and they are especially valuable in solving black-box problems (Deng and Liu, 2024; Sörensen, 2015). Due to their simplicity, flexibility, and avoidance of local minima, they have gained wide application in the recent past (Johnvictor et al., 2022; Eiben and Smith, 2015). Their common application fields include feature selection (Kwakye et al., 2024), engineering optimization (Barua and Merabet, 2024), soft testing (Khoshniat et al., 2024), path planning (Cui et al., 2024), scheduling (Pérez et al., 2023), image processing (Sameh et al., 2024), and network communication (Benbraika et al., 2024).

One of the sources of inspiration for optimization algorithm design is nature (Banzhaf et al., 2006), which has resulted in the creation of crucial paradigms in optimization algorithms, including evolutionary algorithms and swarm intelligence (Das and Suganthan, 2010). Over time, the classification of algorithms based on different sources of inspiration has roughly expanded to include physics-based and human-based algorithms (Mohamed et al., 2020; Singh and Kumar, 2021). On the contrary, metaphor-less algorithms do not emphasize the source of inspiration. Traditional evolutionary algorithms have emerged based on the biological evolution mechanisms of crossover, mutation, selection, etc. (Miikkulainen and Forrest, 2021). Classical evolutionary algorithms are presented in the first part of Table 1. Swarm intelligence refers to the collective intelligent behavior exhibited by self-organizing and decentralized systems (Ab Wahab et al., 2015; Slowik and Kwasnicka, 2017). Several swarm intelligence algorithms are presented in the second part of Table 1. Physics-based algorithms draw inspiration

☆ This work is supported by the National Natural Science Foundation of China (No. 62006144).
* Corresponding author.
*E-mail addresses:* GaooHaoo@hotmail.com (H. Gao), tsingke@sdnu.edu.cn (Q. Zhang).

https://doi.org/10.1016/j.engappai.2024.109202
Received 22 November 2023; Received in revised form 27 May 2024; Accepted 23 August 2024
Available online 30 August 2024

**Nomenclature**

| | |
|---|---|
| $N$ | Population size |
| $D$ | Probelm dimension |
| $lb$ | The lower bound |
| $lb$ | The upper bound |
| $X_i$ | The $i$th candidate solution |
| $E_i$ | The $i$th evolved solution |
| $W_i$ | The $i$th sampled better solution |
| $L_i$ | The $i$th sampled worse solution |
| $t$ | The current iteration |
| $FEs$ | Fitness evaluations |
| $MaxFEs$ | Maximum fitness evaluations |
| $P$ | The base vector |
| $\alpha$ | The attenuation factor |
| $\theta$ | The control parameter |
| $\Delta r_i$ | The $i$th random step size |
| $\omega$ | The non-negative weight |
| $c_a$ | The learning rate of $P_a$ |
| $c_b$ | The learning rate of $P_b$ |
| $P_a$ | The evolution path $a$ |
| $P_b$ | The evolution path $b$ |
| $\mathbf{R_1}$ | The random real matrix |
| $\mathbf{R_2}$ | The random real matrix |
| $\mathbf{A}$ | The with-replacement sampling matrix |
| $\mathbf{B}$ | The without-replacement sampling matrix |
| $\mathbf{S}$ | The random integer matrix |

primarily from quantum theory, electrostatics, electromagnetism, Newton's laws of gravitation, and the laws of motion (Biswas et al., 2013; Can and Alataş, 2015). Several algorithms based on these principles are listed in the third part of Table 1. Human-based algorithms are inspired by how humans think and behave (Hashim et al., 2023; Mohamed et al., 2020). Relevant algorithmic information for this category is listed in the fourth part of Table 1. The most distinctive algorithms are the metaphor-less ones, which are not derived from any specific source of inspiration. The last part of Table 1 provides information regarding several metaphor-less algorithms. Further improvements to all the discussed algorithms are called variant algorithms and are aimed at further enhancing the performance of the original algorithm (Abbaszadeh Shahri et al., 2022a). Regardless of the classification to which the algorithms belong, they require iterative evolution on the computer, and they can evolve surprising and interesting structures (Forrest, 1993). Population-based optimization algorithms can be classified in the way described above, and together with single solution-based optimization algorithms, they constitute stochastic search techniques. Optimization techniques can be divided into application and classical methods at a more macroscopic classification level. Specifically, applied methods consist of heuristic algorithms and approximation algorithms, while heuristic algorithms include stochastic search techniques and deterministic techniques (Abbaszadeh Shahri et al., 2022b). This classification structure contributes to a clearer understanding of the relationships and hierarchy between different optimization techniques. However, no single algorithm is universally superior. Therefore, the selection and design of algorithms need to be considered according to the needs and characteristics of the problem (Han et al., 2024).

To achieve high-quality evolution, it is essential to strike a balance between the algorithmic diversification (exploration) and intensification (exploitation) capabilities (Brahim et al., 2024; Del Ser et al., 2019). Exploration refers to the ability of the algorithm to discover different solutions scattered in the search space. In contrast, exploitation emphasizes the enhancement search process in promising regions to obtain higher-quality solutions (Črepinšek et al., 2013). Therefore, excessive focus on exploration will sacrifice the convergence speed of the algorithm, while undue emphasis on exploitation will increase the probability of entrapment into the local optimum (Yang et al., 2014). Although many ideas and concepts may appear contradictory, striking a reasonable balance between exploration and exploitation is crucial for ensuring optimal algorithmic performance (Morales-Castañeda et al., 2020).

Metaheuristic algorithms use search operators to generate more promising solutions to problems, which often involves how to extract and utilize information effectively. However, this process seems to be blurred and masked by metaphors in the background of excessive metaphors. From a mathematical perspective, the base vector in the search operator usually determines the search's starting point. The base vector is directly sampled from the solution set without considering adaptation. Moreover, they are also prone to ignoring the impact of historical information on the search process. They usually employ multiple operators to approximate the optimal solution, which may make search processes complicated and not easily understood by users. Further, these algorithms still expose an imbalance between exploration and exploitation in the optimization process. The lack of comprehensive and rigorous validation experiments and reliable work in real-world scenarios may still be problems they face.

Based on the above discussion, the motivations behind this study can be distilled into the following points:

1. How to effectively extract and utilize information hidden within candidate solutions and achieve the base vector adaptation are challenges and the first motivation for this work.
2. The complexity of the optimization process leads to multiple operators whose capabilities counteract each other or act equivalently. Therefore, integrating multiple operations for extracting and utilizing information to construct an operator is the second motivation for this work.
3. Ensuring the simplified implementation and efficient performance of the algorithm are necessary (Piotrowski and Napiorkowski, 2018; Yang et al., 2022). Therefore, how to rationally take both into consideration becomes the third motivation for this work.
4. Metaphor-based algorithms have been controversial (Velasco et al., 2024; Vermetten et al., 2024; Sörensen, 2015; Johnvictor et al., 2022). Therefore, the ultimate motivation lies in how to describe the algorithm's work from a non-metaphorical perspective.

To address these problems, this paper proposes an efficient optimization algorithm called alpha evolution (AE). A distinctive operator was designed in AE to drive optimization by incorporating multiple operations of extracting and utilizing information. To achieve the adaptation process of the base vector in the operator, two evolution paths are used to establish correlations for the starting position information, thus allowing previous experience information to be carried over from generation to generation. Furthermore, the composite differential operation of multiple solutions makes the algorithm's search more efficient and stable. Further, the exploration and exploitation capabilities of the proposed algorithm are balanced by the adaptation adjustment of $\alpha$. The main highlights and contributions of AE can be summarized as follows:

1. A novel optimization algorithm named AE is proposed, which uses only one alpha operator to update solutions. This operator realizes the base vector adaptation and integrates various techniques of extracting and utilizing evolutionary information.
2. Unlike operators in other algorithms, the alpha operator can be used for either exploration or exploitation. Therefore, AE does not need additional search operators or components with search capabilities.

**Table 1**
A brief review of optimization algorithms.

| Classification | Algorithms | Inspiration | Proposed time | Citations |
|---|---|---|---|---|
| Evolutionary algorithms | Evolutionary programming (EP) (Fogel, 1998) | The finite state machine | 1966 | 5886 |
| | Evolution strategy (ES) (Rechenberg, 1965) | The mechanisms of biological evolution | 1960 | 1373 |
| | Genetic algorithm (GA) (Sampson, 1976) | The Darwinian evolution | 1975 | 146 |
| | Memetic algorithm (MA) (Moscato et al., 1989) | The Darwinian evolution and Dawkins meme | 1989 | 2622 |
| | Genetic programming (GP) (Koza, 1994) | The mechanisms of biological evolution | 1994 | 24 838 |
| | Cultural algorithm (CA) (Reynolds, 1994) | The evolution process of human culture | 1994 | 1331 |
| | Differential evolution (DE) (Storn and Price, 1997) | The mechanism of biological evolution | 1997 | 33 171 |
| | Differential search (DS) (Civicioglu, 2012) | The random motion of the organism | 2012 | 511 |
| Swarm intelligence | Ant colony optimization (ACO) (Dorigo, 1992) | The foraging behavior of ants | 1992 | 5974 |
| | Particle swarm optimization (PSO) (Kennedy and Eberhart, 1995) | The foraging behavior of birds | 1995 | 80 167 |
| | Harmony search (HS) (Geem et al., 2001) | The mechanism of improvisation performance | 2001 | 7072 |
| | Shuffled frog-leaping algorithm (SFLA) (Eusuff et al., 2006) | The foraging behavior of frogs | 2006 | 1381 |
| | Artificial bee colony (ABC) (Karaboga and Basturk, 2007) | The foraging behavior of honeybees | 2007 | 7959 |
| | Biogeography-based optimization (BBO) (Simon, 2008) | The migration mechanism of biological | 2008 | 4217 |
| | Firefly algorithm (FA) (Yang, 2009) | The behavior mechanism of fireflies | 2009 | 4944 |
| | Cuckoo search (CS) (Yang and Deb, 2009) | The obligate brood parasitic behaviour of the cuckoo | 2009 | 7778 |
| | Flower pollination algorithm (FPA) (Yang, 2012) | The pollination process of flowers | 2012 | 2470 |
| | Grey wolf optimizer (GWO) (Mirjalili et al., 2014) | The hierarchical mechanism and the hunting behavior of wolves | 2014 | 12 173 |
| | Moth-flame optimization (MFO) (Mirjalili, 2015) | The navigation mechanism of moth | 2015 | 3301 |
| | Whale optimization algorithm (WOA) (Mirjalili and Lewis, 2016) | The bubble-net hunting strategy of whales | 2016 | 8293 |
| | Salp swarm algorithm (SSA) (Mirjalili et al., 2017) | The movement and foraging behavior of salp swarm | 2017 | 3446 |
| | Coyote optimization algorithm (COA) (Pierezan and Coelho, 2018) | The social behavior of coyotes | 2018 | 433 |
| | Harris hawks optimization (HHO) (Heidari et al., 2019) | The cooperation and foraging behavior of Harris's hawks | 2019 | 3121 |
| | Butterfly optimization algorithm (BOA) (Arora and Singh, 2019) | The foraging and mating behavior of butterflies | 2019 | 998 |
| | Supply–demand-based optimization (SDO) (Zhao et al., 2019b) | The mechanism of market supply and demand | 2019 | 114 |
| | Manta ray foraging optimization (MRFO) (Zhao et al., 2020b) | The foraging behavior of manta rays | 2020 | 557 |
| | Marine predators algorithm (MPA) (Faramarzi et al., 2020a) | The survival evolutionary mechanism of marine animals | 2020 | 1132 |
| | Aquila optimizer (AO) (Abualigah et al., 2021b) | The foraging behavior of Aquila | 2021 | 1049 |
| | Red fox optimization (RFO) (Połap and Woźniak, 2021) | The living and hunting habits of red foxes | 2021 | 186 |
| | Firebug swarm optimization (FSO) (Noel et al., 2021) | The swarm breeding behavior of fireflies | 2021 | 12 |
| | Remora optimization algorithm (ROA) (Jia et al., 2021) | The parasitic behavior of remoras | 2021 | 120 |
| | White shark optimizer (WSO) (Braik et al., 2022a) | The movement and foraging behavior of white sharks | 2022 | 101 |
| | Honey badger algorithm (HBA) (Hashim et al., 2022) | The foraging behavior of honey badgers | 2022 | 343 |
| | Dwarf mongoose optimization algorithm (DMOA) (Agushaka et al., 2022) | The foraging behavior of dwarf mongooses | 2022 | 279 |
| | Snake optimizer (SO) (Hashim and Hussien, 2022) | The foraging and reproductive behavior of snakes | 2022 | 158 |
| | Prairie dog optimization algorithm (PDO) (Ezugwu et al., 2022) | The survival behavior of prairie dogs | 2022 | 113 |
| | Reptile search algorithm (RSA) (Abualigah et al., 2022) | The foraging behavior of crocodiles | 2022 | 537 |
| | Ebola optimization search algorithm (EOSA) (Oyelade et al., 2022) | The mechanism of transmission of Ebola virus | 2022 | 195 |
| | Poplar optimization algorithm (POA) (Chen et al., 2022) | The sexual and asexual propagation mechanisms of poplars | 2022 | 16 |
| | Beluga whale optimization (BWO) (Zhong et al., 2022) | The behaviors of beluga whales | 2022 | 59 |
| | Chimp optimization algorithm (ChOA) (Khishe and Mosavi, 2020) | The intelligence and sexual motivation of chimps | 2022 | 537 |

(*continued on next page*)

**Table 1** (*continued*).

| Classification | Algorithms | Inspiration | Proposed time | Citations |
|---|---|---|---|---|
| | Spider wasp optimizer (SWO) (Abdel-Basset et al., 2023e) | The hunting, nesting, and mating behaviors of the female spider wasps | 2023 | 5 |
| | Nutcracker optimizer algorithm (NOA) (Abdel-Basset et al., 2023d) | The behaviors of Clark's nutcrackers | 2023 | 13 |
| Physics-based algorithms | Simulated annealing (SA) (Kirkpatrick et al., 1983) | The principle of solid annealing | 1983 | 55 949 |
| | Gravitational search algorithm (GSA) (Rashedi et al., 2009) | The law of gravity and Newton's second law | 2009 | 6929 |
| | Lightning search algorithm (LSA) (Shareef et al., 2015) | The natural phenomenon of lightning | 2015 | 346 |
| | Stochastic fractal search (SFS) (Salimi, 2015) | The fractal concept | 2015 | 517 |
| | Sine cosine algorithm (SCA) (Mirjalili, 2016) | The sine and cosine functions | 2016 | 3502 |
| | Multi-verse optimizer (MVO) (Mirjalili et al., 2016) | The cosmological concepts | 2016 | 1982 |
| | Electromagnetic field optimization (EFO) (Abedinpourshotorban et al., 2016) | The behavior of electromagnets with different polarities | 2016 | 251 |
| | Atom search optimization (ASO) (Zhao et al., 2019a) | The motion mechanism of atoms | 2019 | 405 |
| | Artificial electric field algorithm (AEFA) (Yadav et al., 2019) | The Coulomb's law of electrostatic force | 2019 | 184 |
| | Artificial ecosystem-based optimization (AEO) (Zhao et al., 2020a) | The production, consumption, and decomposition behaviors of organisms | 2019 | 246 |
| | Equilibrium optimizer (EO) (Faramarzi et al., 2020b) | The model of control volume mass balance | 2020 | 1190 |
| | Arithmetic optimization algorithm (AOA) (Abualigah et al., 2021a) | The distribution behavior of the main arithmetic operators in mathematics | 2021 | 1297 |
| | Homonuclear molecules optimization (HMO) (Mahdavi-Meymand and Zounemat-Kermani, 2022) | The Bohr atomic model and the structure of homonuclear molecules | 2022 | 3 |
| | Kepler optimization algorithm (KOA) (Abdel-Basset et al., 2023c) | The Kepler's laws of planetary motion | 2023 | 9 |
| | Snow ablation optimizer (SAO) (Deng and Liu, 2023) | The sublimation and melting behavior of snow | 2023 | 1 |
| | RIME (Su et al., 2023) | The soft-rime and hard-rime growth process of rime-ice | 2023 | 27 |
| Human-based algorithms | Brain storm optimization (BSO) (Shi, 2011) | The brainstorming process of humans | 2011 | 707 |
| | Teaching-learning-based optimization (TLBO) (Rao et al., 2011) | The process of classroom teaching | 2011 | 4140 |
| | Heap-based optimizer (HBO) (Askari et al., 2020a) | The company hierarchy | 2020 | 200 |
| | Coronavirus herd immunity optimizer (CHIO) (Al-Betar et al., 2021) | The mechanism of herd immunity | 2020 | 172 |
| | Gaining sharing knowledge (GSK) (Mohamed et al., 2020) | The process of gaining and sharing knowledge during the human life span | 2020 | 256 |
| | Human felicity algorithm (HFA) (Veysari et al., 2022) | The human pursuit of happiness | 2021 | 19 |
| | Search and rescue optimization algorithm (SAR) (Shabani et al., 2020) | The human exploration behavior in search and rescue operations | 2021 | 107 |
| | Growth optimizer (GO) (Zhang et al., 2023) | The learning and reflection mechanisms in human growth | 2022 | 12 |
| | Fans optimizer (FO) (Wang et al., 2023b) | The Fans economy fused in the entertainment domain | 2023 | 1 |
| Metaphor-less algorithms | Jaya (Rao, 2016) | No source of inspiration | 2016 | 1922 |
| | Best-worst-play algorithm (BWP) (Singh et al., 2020) | No source of inspiration | 2020 | 5 |
| | Rao algorithm (Rao) (Rao, 2020) | No source of inspiration | 2020 | 229 |
| | Runge kutta optimizer (RUN) (Ahmadianfar et al., 2021) | No source of inspiration | 2021 | 446 |
| | Weighted mean of vector (INFO) (Ahmadianfar et al., 2022) | No source of inspiration | 2022 | 232 |

3. AE has no special hyperparameters, and its code implementation is relatively compact. Therefore, it is easy for researchers to understand and use.
4. Metaphor-based algorithms may over-rely on metaphors. However, the AE algorithm does not consider metaphor as an innovation, and its mathematical model is also directly presented.
5. AE can fast convergence, avoid local optima, and obtain high-quality solutions in multiple sequence alignment and engineering design. Therefore, it can provide underlying algorithmic support for the fields of bioinformatics and engineering.

The rest of the paper is organized as follows. The proposed AE is introduced in Section 2. The experimental results and discussion are shown and analyzed in Section 3. Sections 4 and 5 respectively describe the applications of AE for multiple sequence alignment and engineering design problems. The conclusion and future work are presented in Section 6.

## 2. Alpha evolution (AE)

### 2.1. Research gap

In terms of algorithm theory, although similar studies have made significant contributions to computational intelligence, their research still focuses on a metaphorical perspective. However, while it is easy for metaphor-based algorithms to see metaphors as the main innovation, this can easily obscure the nature of the algorithm and make it more difficult to be understood directly (Brahim et al., 2024). It is crucial to extract and utilize evolutionary information in algorithmic iteration effectively. In fact, this issue is unrelated to metaphors but is closely associated with the operators that generate candidate solutions. The search operators of metaheuristic algorithms usually comprise base vectors and step sizes from a macro perspective. However, base vectors in their operators usually do not track and record the progress of the population but rather focus too much on the progress of a single solution (Qing, 2008). Macroscopically estimating base vectors can reduce

the impact of local optimal information on the algorithm's convergence performance, but previous studies have focused on increasing the expectation of promising information in base vectors (de Anda-Suárez et al., 2022). However, the population can easily get stuck in this trap if a promising region is locally optimal. They rarely consider unpromising information and fail to consider that fitness is not the only criterion distinguishing good from bad information. Unlike them, AE aims to reduce expectations for promising information in the base vector and achieve complete adaptation of the base vector. The evolution path (a sequence of successive steps) is also introduced into the calculation of the AE's base vector to enhance the correlation of continuous steps. Therefore, two evolution paths that record evolutionary information are designed to guide the algorithm toward a promising region. The well-known covariance matrix adaptive evolution strategy has also introduced it and adaptively updated the covariance matrix (Hansen et al., 2003). To achieve the movement of the solution in the search space, the algorithm needs to generate a step size and add it to the base vector. Although it has been commonly studied to estimate the gradient of a problem using differential vectors or random vectors as step sizes, the way AE computes and controls these two vectors still distinguishes it from other similar techniques.

In terms of research and verification, algorithm researchers tend to overlook the structural biases of algorithms, such as the algorithm not working in spaces far from 0 (Castelli et al., 2022) or the problem of algorithm forcing searches in certain regions (Kononova et al., 2015). When a researcher finally adapts the algorithm to the application at hand, only to find that the algorithm does not perform as well as claimed, the researcher is likely to be frustrated. In other words, algorithms that introduce structural bias increase user bias in the computational intelligence field (Vermetten et al., 2022). Furthermore, the algorithm proposer did not provide readers with a comprehensive perspective on validating the algorithm. Although a study has showcased and discussed the currently published numerous metaheuristic algorithms (Ma et al., 2023), it still lacks a comprehensive and detailed comparison.

### 2.2. Algorithm model

This section provides a complete mathematical model of the AE algorithm for solving minimization problems. The nomenclature is listed below to present and explain the AE algorithm model more clearly. Moreover, these concepts are expressed directly without the need for metaphors.

#### 2.2.1. Initialization

The population initialization strategy generates a set of candidate solutions based on the problem space to provide the evolutionary algorithm with an initial guess about the promising information (Xu et al., 2021). In a $D$-dimensional search space, the candidate solution $X_i$, often metaphorically referred to as the individual, is typically initialized uniformly using Eq. (1):

$$X_i = lb + (ub - lb) \cdot \mathrm{rand}\,(0, 1, [1, D])\,,\ i = 1, 2, \ldots, N \tag{1}$$

where $X_i$ denotes the $i$th candidate solution, while $lb$ and $ub$ refer to the lower and upper bounds of the search space, respectively. Moreover, rand is defined as a random number generator. $\mathrm{rand}(0, 1, [1, D])$ generates a vector of $D$ columns containing random numbers uniformly distributed in the range $[0, 1]$. Moreover, $N$ represents the number of solutions, usually likened to the population size.

Indeed, $X_i$ is a $D$-dimensional vector, represented by $X_i = (X_{i,1}, X_{i,2}, \ldots, X_{i,j}, \ldots, X_{i,D})$, where $X_{i,j}$ denotes the $j$th decision variable of the solution. The objective function $f(X_i)$ is used to evaluate the quality of $X_i$, which usually returns a real value. Herein, this value is likened to fitness in biology. For minimization problems, the optimization objective of the algorithm is usually to minimize the objective function.

#### 2.2.2. Alpha operator

It is necessary to improve the candidate solutions to find the optimal solution to the problem, which is a crucial step in the optimization algorithm (Yang et al., 2024). In this work, the set of candidate solutions is defined as the candidate matrix $\mathbf{X}$, while the evolution matrix $\mathbf{E}$ is a subset of the candidate matrix. Note that the population is usually a metaphor for the solution set, which is essentially a matrix. In the evolution process, instead of directly evolving each solution $X_i$ in the current candidate matrix $\mathbf{X}$, a sampling with replacement operation is performed on $\mathbf{X}$ to obtain the evolution matrix $\mathbf{E}$ to be evolved. Fig. 1 and Eq. (2) describe this relationship:

$$\mathbf{E} \overset{N}{\rightarrow} \mathbf{X}, \mathbf{E} \subseteq \mathbf{X} \tag{2}$$

where the symbol "→" represents the operation that performs sampling with replacement, which will be executed $N$ times. The candidate matrix $\mathbf{X}$ is responsible for providing solutions, while the evolution matrix $\mathbf{E}$ is responsible for exploring new solutions. Note that when $E_i$ successfully evolves, the corresponding solution in $\mathbf{X}$ will be replaced based on the index.

For optimization algorithms, the performance heavily depends on the search operator during the iterative process (Wang et al., 2023a). Therefore, an excellent search operator plays a crucial role in guiding the algorithm to discover solutions of higher quality. Unlike other algorithms, the proposed algorithm relies on only one operator (the alpha operator) for efficient search. In this operator, multiple steps of extracting and utilizing evolutionary information now occur simultaneously in one operator. The mathematical model of the operator is given in Eq. (3):

$$E_i^{t+1} = P + \alpha \Delta r_i + \theta \cdot \left(W_i + E_i^t - P - L_i\right) \tag{3}$$

- $E_i$, the $i$th evolved solution in the evolution matrix $\mathbf{E}$.
- $t$, the current iteration.
- $P$, the base vector, which determines the evolutionary starting position.
- $\alpha$, the attenuation factor, which controls algorithmic exploration and exploitation.
- $\Delta r_i$, the $i$th random step size.
- $\theta$, the control parameter, which controls the differential vector (self-adaptive step size).
- $W_i$ and $L_i$, the sampled solutions from $\mathbf{X}$, satisfying $f(W_i) \leq f(E_i) \leq f(L_i)$.

Next, this operator is divided into three components, namely $P$, $\alpha \Delta r_i$, and $\theta \cdot \left(W_i + E_i^t - P - L_i\right)$, and describe them in the following order. Note that the three components generate the base vector, the random step size, and the self-adaptive step size, respectively.

**For the adaptive base vector** $P$, the component plays a vital role in the updating of solutions because it determines evolutionary starting point, which is initially calculated in two ways according to Eq. (4):

$$P = \begin{cases} \mathrm{diagonal}\,(\mathbf{A})\,, & if\ \mathrm{rand}(0, 1) < 0.5 \\ \omega \mathbf{B}, & otherwise \end{cases} \tag{4}$$

where diagonal represents a function that obtains diagonal elements of a matrix. $\mathbf{A}$ is a $D$th order square matrix, which is obtained by sampling with replacement $D$ times the candidate solutions from $\mathbf{X}$. And $\mathbf{B}$ is a matrix with $K$ rows and $D$ columns, which is obtained by sampling without replacement $K$ times ($K = \lceil N \times \mathrm{rand}(0, 1) \rceil$ and $1 \leq K \leq N$) the candidate solutions from $\mathbf{X}$. Fig. 2 shows how to perform diagonal($\mathbf{A}$) and $\omega\mathbf{B}$. Moreover, $\omega_{i:K}$ represents the weight of the $i$th solution in $\mathbf{B}$, which is calculated by Eq. (5):

$$\omega_{i:K} = \frac{f(X_{i:K})}{\sum_{i=1}^{K} f(X_{i:K})} \tag{5}$$

where $i:K$ represents the $i$th solution out of the sampled $K$ solutions, and $f(X_{i:K})$ returns the objective function value of $X_{i:K}$.

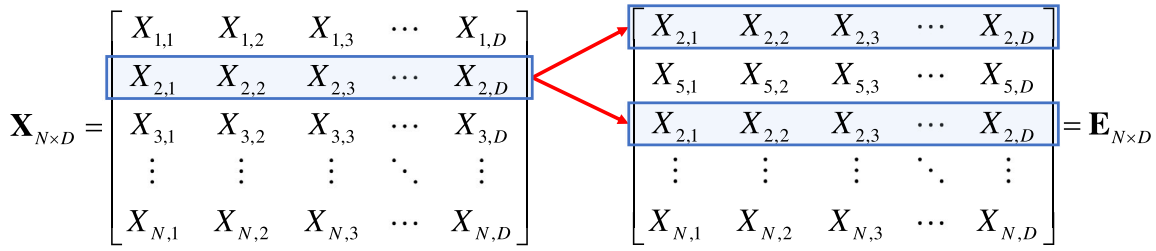


**Fig. 1.** The evolution matrix **E** is obtained by sampling with replacement from the candidate matrix **X**.

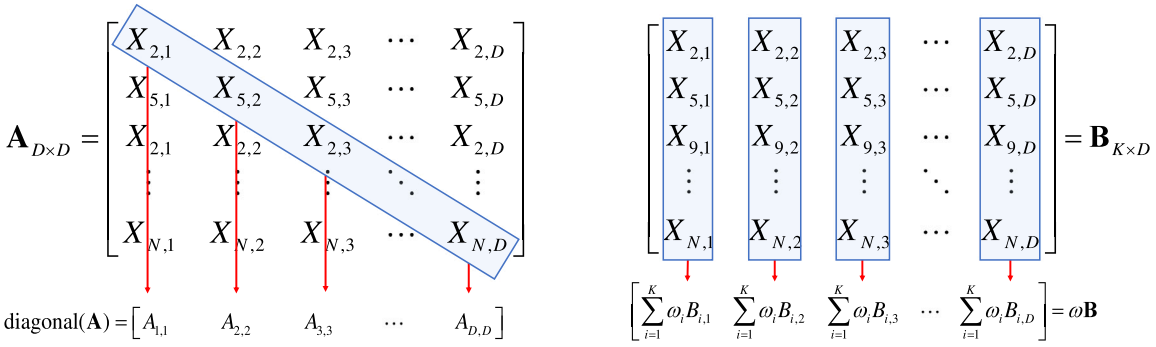


**Fig. 2.** The operation principles of diagonal(**A**) and $\omega\mathbf{B}$.

However, $P$ produced by **A** or **B** itself does not participate in evolution, which results in no connection among each generation of $P$, causing the loss of evolutionary information. Therefore, an evolution path concept is introduced to increase the correlation between successive iteration steps (Hansen and Ostermeier, 1996), that is, to respectively construct two independent evolution paths $P_a$ and $P_b$ for $P$ produced by **A** and **B**. Certainly, one might question why a single independent evolutionary path is not constructed instead. This is because the methods by which **A** and **B** generate $P$ are entirely distinct. If both are employed in constructing a single evolution path, it would significantly disrupt the accumulation of path information, leading to a reduced correlation between continuous iterative steps. Therefore, it is necessary to improve Eq. (4). Based on the above description, two different evolution paths are constructed for $P$ using Eq. (6):

$$P = \begin{cases} c_a P_a^t + (1-c_a) \times \text{diagonal}(\mathbf{A}) = P_a^{t+1}, & if\ \text{rand}(0,1) < 0.5 \\ c_b P_b^t + (1-c_b) \times \omega\mathbf{B} = P_b^{t+1}, & otherwise \end{cases} \tag{6}$$

where $c_a$ and $c_b$ represent the learning rates of $P_a$ and $P_b$, respectively. They are both calculated as $1 - FEs/MaxFEs$. The learning rates are designed to continuously enhance the impact of current information and weaken the influence of historical information, which will complete the transformation of the algorithm from exploration to exploitation. Moreover, the initial paths $P_a^0$ and $P_b^0$ will be randomly generated based on the effective search space.

**For the random step size** $\alpha\Delta r_i$, the component provides a global exploration function. The attenuation factor $\alpha$ is a nonlinear decreasing value that is closely related to the perturbation matrix $\boldsymbol{\Delta}\mathbf{r}$. Its attenuation process is described in Fig. 3 and calculated using Eq. (7):

$$\alpha = e^{\left(\ln\left(\frac{MaxFEs-FEs}{MaxFEs}\right)-\left(\frac{4FEs}{MaxFEs}\right)^2\right)} \tag{7}$$

where $FEs$ represents fitness evaluations (the current number of calls to the objective function), while $MaxFEs$ represents maximum fitness evaluations (the maximum number of calls to the objective function). Furthermore, $e$ represents the exponential function, while ln represents the natural logarithm function. Additionally, $\boldsymbol{\Delta}\mathbf{r}$ is defined as the perturbation that gradually fades under the influence of $\alpha$ to achieve

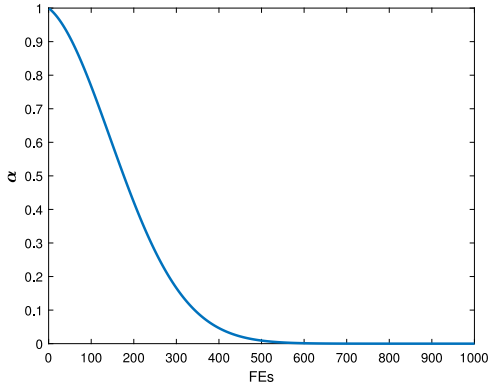



**Fig. 3.** The change of $\alpha$ during 1000 fitness evaluations processes.

the transition of the algorithm from exploration to exploitation. It is calculated by Eq. (8):

$$\boldsymbol{\Delta}\mathbf{r} = (ub - lb) \cdot \left(2\mathbf{R}_1 \cdot \mathbf{R}_2 - \mathbf{R}_2\right) \cdot \mathbf{S} \tag{8}$$

where $\mathbf{R}_1$ and $\mathbf{R}_2$ represent random real matrices generated by $\text{rand}(0, 1, [N, D])$, and they are used to generate perturbations, while **S** represents a random integer matrix containing only 0 or 1 generated by $\text{randi}(0, 1, [N, D])$, which trades off perturbations based on dimensions. Note that $\boldsymbol{\Delta}\mathbf{r}$ is a matrix with $N$ rows and $D$ columns, and $\Delta r_i$ represents the $i$th row vector within $\boldsymbol{\Delta}\mathbf{r}$.

**For the self-adaptive step size** $\theta \cdot \left(W_i + E_i^t - P - L_i\right)$, the component contributes a local exploitation function, where $W_i$ and $L_i$ represent solutions sampled from **X**, they satisfy $f(W_i) \le f(E_i) \le f(L_i)$. Moreover, $\theta$ is an uncertainty control vector, and its uncertainty is due to each of its components can be the same or different. If the values of each dimension are the same, use $\text{rand}(0, 2)$ to determine them. If the values of each dimension are different, use $\text{rand}(0, 1)$ to determine them separately. Note that the probability of these two scenarios occurring is equal.

#### 2.2.3. Boundary constraint

The boundary constraint method ensures the algorithm effectively searches within the search space, particularly for constrained optimization problems. Common boundary constraint methods such as clipping, random, reflection, periodic, and halving the distance (Kadavy et al., 2022). In the AE algorithm, halving the distance method is employed as described by Eq. (9):

$$E_{i,j} = \begin{cases} \frac{E_{i,j}+ub}{2}, & if\ E_{i,j} > ub \\ \frac{E_{i,j}+lb}{2}, & if\ E_{i,j} < lb \\ E_{i,j}, & otherwise \end{cases} \tag{9}$$

where $ub$ and $lb$ are the upper and lower bounds of the search space, respectively. Note that $E_i$ is responsible for generating better solutions, which is why it is constrained, not $X_i$.

#### 2.2.4. Selection strategy

The selection strategies are methods to add relevant solutions to the next generation set. Common selection strategies include greedy selection, roulette-wheel selection, tournament selection, truncation selection, etc (Bäck et al., 2018). Among them, greedy selection puts successfully evolved solutions into the next generation through a simple method without additional operations. Therefore, this selection strategy is introduced into the AE algorithm, and its model can be presented by Eq. (10):

$$X_k^{t+1} = \begin{cases} E_i^{t+1}, & if\ f\left(E_i^{t+1}\right) <= f\left(E_i^t\right) \\ X_k^t, & otherwise \end{cases} \tag{10}$$

where the $i$th solution in $\mathbf{E}$ actually corresponds to the $k$th solution in $\mathbf{X}$.

**Algorithm 1:** AE

**Input:** $N$, $D$, $ub$, $lb$, $FEs = 0$, $MaxFEs$
**Output:** Global optimal solution: $X_{best}$

Initialize the candidate matrix $\mathbf{X}$ using Eq. (1) and evaluate it
$FEs = FEs + N$
**while** $FEs < MaxFEs$ **do**
  $\mathbf{E} \overset{N}{\rightarrow} \mathbf{X}$
  $ind = \text{sort}(f(\mathbf{X}))$
  $\mathbf{R}_1 = \text{rand}(0, 1, [N, D])$
  $\mathbf{R}_2 = \text{rand}(0, 1, [N, D])$
  $\mathbf{S} = \text{randi}(0, 1, [N, D])$
  $\Delta \mathbf{r} = (ub - lb) \cdot \left(2\mathbf{R}_1 \cdot \mathbf{R}_2 - \mathbf{R}_2\right) \cdot \mathbf{S}$
  $\alpha = \exp(\ln(1 - FEs/MaxFEs) - (4FEs/MaxFEs)^2)$
  **for** $i = 1 : N$ **do**
    **if** $\text{rand}(0, 1) < 0.5$ **then**
      $\mathbf{A} = \mathbf{X}(\text{randi}(1, N, [D, 1]), :)$
      $c_a = 1 - FEs/MaxFEs$
      $P_a^{t+1} = c_a P_a^t + (1 - c_a) \times \text{diagonal}(\mathbf{A})$
      $P = P_a^{t+1}$
    **else**
      $K = \lceil N \times \text{rand}(0, 1) \rceil$
      $I_1 = \text{randperm}(N, K)$
      $\mathbf{B} = \mathbf{X}(I_1, :)$
      $\omega = f(I_1)/\text{sum}(f(I_1))$
      $c_b = 1 - FEs/MaxFEs$
      $P_b^{t+1} = c_b P_b^t + (1 - c_b) \times \omega \mathbf{B}$
      $P = P_b^{t+1}$
    **end**
    $W_i = X(ind(\text{randi}([1, \text{length}(1 : \text{find}(k == ind))])), :)$
    $L_i = X(ind(\text{randi}([\text{length}(1 : \text{find}(k == ind)), N])), :)$
    $I_2 = \text{round}(\text{rand}(0, 1))$
    $\theta = I_2 \times \text{ones}(1, D) \cdot \text{rand}(0, 2) + (1 - I_2) \times \text{rand}(0, 1, [1, D])$
    $E_i^{t+1} = P + \alpha \Delta r_i + \theta \cdot \left(W_i + E_i^t - P - L_i\right)$
    Boundary constraints on variables using Eq. (9)
    Select $X_k$ according to Eq. (10)
    Record $X_{best}$
    $FEs = FEs + 1$
  **end**
**end**

To show a more detailed presentation of the computational process of AE, the pseudo-code (most of which is source code) for AE is presented in **Algorithm 1**. This version regards maximum fitness evaluations as the termination criterion and is used to solve minimization problems. Furthermore, Fig. 4 presents the flowchart of the proposed algorithm.

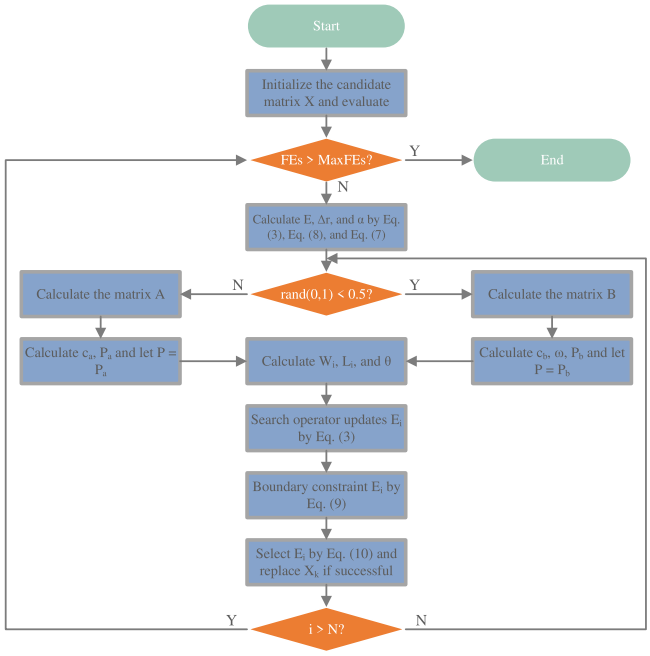



**Fig. 4.** The flowchart of the AE algorithm.

### 2.3. Algorithm theory

According to the global convergence theorem (Solis and Wets, 1981; Rudolph, 1996), the convergence of AE was proved through the following two lemmas. $f : \mathbb{R} \rightarrow \mathbb{R}^n$ is assumed to be the objective function of a problem. $\Omega$ represents the set of all feasible solutions in the problem space.

**Lemma 1.** *An algorithm is denoted as $Al$, $G$ represents an optimal solution in a finite set $Z$, and $X_g$ represents an optimal solution found by the algorithm. if $f(Al(G, X_g)) \leq f(G)$ and $X_g \in Z$, then $f(Alg(G, X_g)) \leq f(X_g)$. In other words, if the algorithm improves the tentative optimal solution in solution set $Z$, and if the optimal solution obtained by the algorithm is also in solution set $Z$, then the optimal solution obtained by the algorithm itself can also be improved. Therefore, the algorithm can achieve convergence.*

The greedy selection mechanism of the AE algorithm enables one-on-one comparison between offspring and parents, thus maintaining the diversity of the algorithm. Furthermore, this choice will only lead the solution $X_i$ in a better direction. Eq. (11) describes the comparison between the currently discovered optimal solution $X_g$ and the newly generated solution $X_i^{t+1}$. It is evident that whenever a better solution is found, $X_g$ will be replaced by $X_i^{t+1}$, and the AE algorithm will converge with a probability of 1. Furthermore, the change in fitness of each solution is monotonic, i.e., $f(X_i^{t+1}) <= f(X_i^t)$. Due to its greedy selection mechanism, the evolution of the solution does not regress; therefore, the AE algorithm can be proven to converge asymptotically.

$$Al\left(X_i^{t+1}, X_g\right) = \begin{cases} X_g, & if\ f\left(X_g < f\left(X_i^{t+1}\right)\right) \\ X_i^{t+1}, & otherwise \end{cases} \tag{11}$$

**Lemma 2.** *After initializing at any feasible point, an algorithm must probabilistically generate a set $Z$ ($Z \subseteq \Omega$) that includes a finite sequence of*

*points. Convergence occurs if $Y$ ($Y \subseteq Z$) is obtained by sufficiently densely sampling the entire feasible set $Z$, and the sampling operation is conducted in a non-zero measure neighborhood $v(Y)$ ($v(Y) > 0$) of any other feasible point. Herein, the mathematical expression for algorithm convergence is Eq. (12):*

$$\prod_{k=0}^{\infty}\left[1-\mathbb{P}_k(Y)\right]=0 \tag{12}$$

*where $\mathbb{P}_k(Y)$ represents the probability that fulfills the global convergence criterion.*

**Proof.** All candidate solutions searched by AE during the iteration process can easily form a set $Z$. Because any solution $X_i^t$ within $Z$ is constrained by Eq. (9), thereby it must satisfy $lb \le X_i^t \le ub$. Further, it can be obtained that $Z \subseteq \Omega$. The operation process of Eq. (3) can be understood as the alpha operator sampling from $Z$ to obtain $X_i^t$, then establishing a non-zero sampling space near $X_i^t$ and performing sampling to obtain $E_i^t$. The above process can be considered as each operation of the alpha operator sampling the neighborhood of $Z$, and from the entire operation process of the algorithm, this sampling process must be dense. If any $E_i^t$ of the $t$th iteration is included in set $Y$, it must satisfy that $\bigcup_{i=1}^{N} E_i^t = Y \subseteq Z$. When $v(Y) > 0$, $\mathbb{P}_k(Y) = \sum \lim_{k=1}^{N} \mathbb{P}_k^t(Y) = 1$. Therefore, it can be obtained that $\prod_{k=0}^{\infty}\left[1-\mathbb{P}_k(Y)\right]=0$.

Of course, the convergence of AE can also be analyzed from the simplest and most extreme perspective. When $N \to +\infty$, the first iteration of AE can weakly converge with a probability of 1. This is because the population was too large and scattered throughout the problem space during the first iteration. In this case, it is possible to immediately find a sub-optimal or global optimal solution to the problem. In addition, when $N$ is finite but $t \to +\infty$, according to the global search convergence theorem, Eq. (13) can be obtained:

$$\lim_{t\to+\infty}\mathbb{P}\left(X_i^t = X_{best}\right)=1 \tag{13}$$

where $\mathbb{P}\left(X_i^t = X_{best}\right)$ represents that $X_i^t$ is the probability of the global optimal $X_{best}$.

### 2.4. Algorithm management

The mathematical model of the proposed AE algorithm can only be applied to solve single-objective minimization optimization problems in this section. Additional adjustments to the mathematical model of AE need to be made according to the problem when AE is applied to other problems. In essence, AE is still a population-based algorithm. Therefore, the number of candidate solutions $N$ in the mathematical model should be set according to the recommendation of this paper; otherwise, it will change the algorithm's performance. In addition, from Eq. (1), Eq. (8), and Eq. (9), AE needs to search in the given feasible domain $[lb, ub]$, and users who are unable to determine the feasible domain of the problem will directly affect the effectiveness of the model. Unlike deterministic optimization techniques, AE is essentially a stochastic search technique, and stochasticity makes the results of each run potentially different. Therefore, users need to choose appropriate algorithms based on problem characteristics and actual requirements. If AE is considered appropriate for the current problem, special attention should be directed toward Eq. (3), which is the core component of AE. Any adjustments made to the core may have implications for the outcome.

## 3. Experimental studies

In the experiments, AE was first subjected to search bias and invariance analysis, scalability analysis, parameter sensitivity analysis, search behavior analysis, qualitative analysis, and exploration and exploitation analysis. Furthermore, AE and 106 algorithms (including 100 metaheuristic algorithms and 6 algorithm variants) conducted numerical optimization experiments. Finally, the computational complexity of AE was analyzed.

**Table 2**
The experimental setup of the well-known algorithms.

| Algorithm | Parameter settings | Citations |
|---|---|---|
| Grey wolf optimizer (GWO) | $\alpha \in [0,2]$ | 12173 |
| Whale optimization algorithm (WOA) | $a_1 \in [0,2], a_2 \in [0,2], b = 1$ | 8293 |
| Sine cosine algorithm (SCA) | $a = 2$ | 3502 |
| Salp swarm algorithm (SSA) | $c_1 \in [0,2]$ | 3446 |
| Arithmetic optimization algorithm (AOA) | $MOP \in [0.2,1], \alpha = 5, \mu = 0.499$ | 1297 |

### 3.1. Search bias and invariance analysis

Search bias is an emerging problem in the field of optimization. A study has revealed that around half of the 90 algorithms proposed in the past three decades have exhibited this problem. Especially, the number of methods introducing search bias has rapidly increased in the past five years (Kudela, 2023). Optimization algorithms very easily introduce an unfair search bias, meaning they tend to move toward the center of the search space or converge toward zero when solving problems with symmetry properties centered at the origin. Additionally, according to a critical discussion (Castelli et al., 2022), introducing boundary value calculations in the search operator may lead to solutions being constantly clipped to the boundary, rendering the algorithm ineffective. This phenomenon becomes more severe as the search space moves further away from zero.

Taking into account the aforementioned problems, this experiment aims to demonstrate that the proposed AE algorithm does not exhibit bias toward the origin, nor is it affected by search space shifting or scaling. We constructed four different research cases based on the standard sphere function. All algorithm details and parameter configurations are presented in Table 2. In particular, the parameter settings of these algorithms are derived from the articles in which they are proposed. They were independently executed 30 times for each case, with $N = 40$, $D = 2$, and $MaxFEs = D \times 10,000$ for each run. The details of the four cases are as follows:

- In the case 1, the standard sphere function is used with an effective search space of $[-10^2, 10^2]$. Typically, algorithms with a search bias in this case converge quickly to the global optimum.
- In the case 2, a shift operation is performed on the search space of case 1, shifting its effective search space to $[10^6, 10^6 + 2 \times 10^2]$ away from 0. Typically, in this case, algorithms that tend to converge toward 0 will fail, especially those that exhibit abnormal convergence.
- In the case 3, a shift operation is performed on the center of case 2 (the center is the location of the global optimal solution). Typically, algorithms that tend to search toward the center of the space will fail in this case.
- In the case 4, the effective search space is scaled for case 3 to $[10^6, 10^6 + 1]$. In this case, introducing an algorithm that calculates boundary values will continuously constrain the solution, rendering it unable to work.

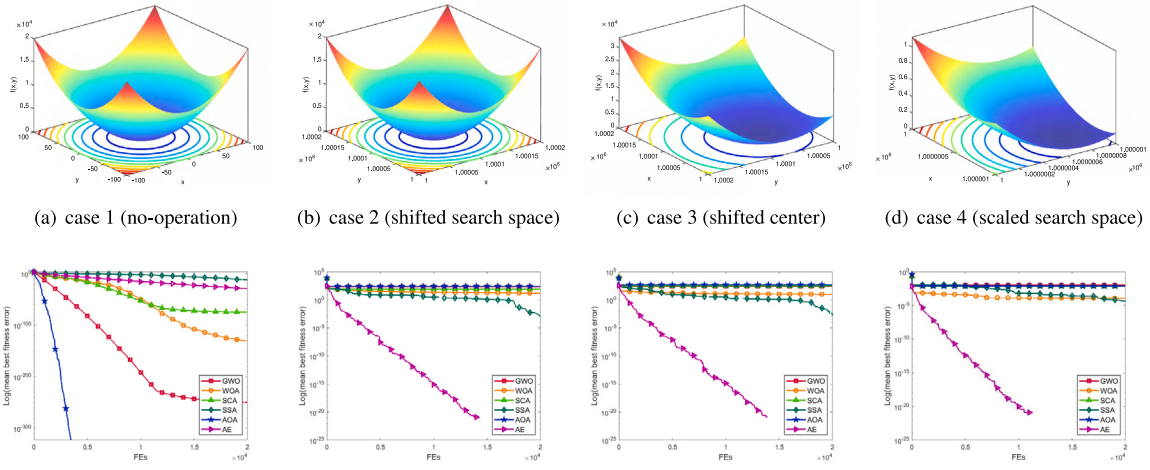
Table 3 and Fig. 5 present the results of the algorithms involved in four case studies. It is evident that most algorithms converge quickly on the standard sphere function, especially AOA. However, when the search space is shifted, the search performance of all algorithms except for AE deteriorates significantly. Furthermore, as the global optimum is constantly shifted, SSA appears to have a specific convergence trend. However, when the search space is scaled, this trend disappears in SSA. Instead, AE demonstrates competitive search behavior. Based on the aforementioned experiments and discussions, the conclusion is drawn as follows: (1) AE has no search bias toward the origin. (2) AE has the shift invariance for search space. (3) AE has the scale invariance for search space.

**Table 3**
The experimental results of AE and the well-known algorithms in the four cases.

| Case | GWO | WOA | SCA | SSA | AOA | AE |
|---|---|---|---|---|---|---|
| case 1 | 5.63E−241 ± 0.00E+00 | 5.81E−125 ± 1.84E−124 | 1.49E−12 ± 2.19E−12 | 8.15E−13 ± 8.40E−13 | **0.00E+00 ± 0.00E+00** | 2.03E−65 ± 3.40E−65 |
| case 2 | 2.06E+02 ± 2.61E+02 | 1.50E+01 ± 1.20E+01 | 7.87E+01 ± 9.03E+01 | 9.21E−04 ± 1.18E−03 | 2.46E+02 ± 3.24E+02 | **0.00E+00 ± 0.00E+00** |
| case 3 | 2.45E+02 ± 2.23E+02 | 9.37E+00 ± 6.97E+00 | 2.93E+02 ± 2.36E+02 | 1.42E−03 ± 2.16E−03 | 4.54E+02 ± 5.09E+02 | **0.00E+00 ± 0.00E+00** |
| case 4 | 9.25E−03 ± 7.25E−03 | 9.24E−05 ± 1.34E−04 | 4.14E−05 ± 3.73E−05 | 6.85E−03 ± 1.01E−02 | 8.57E−03 ± 6.81E−03 | **0.00E+00 ± 0.00E+00** |

**Table 4**
The scalability experiment results on the shifted sphere function from the CEC'05 test suite.

| Dimension | GWO | WOA | SCA | SSA | AOA | AE |
|---|---|---|---|---|---|---|
| $D = 2$ | 3.20E−05 ± 2.18E−05 | 8.68E−09 ± 1.35E−08 | 6.26E−02 ± 8.09E−02 | 1.65E−12 ± 1.25E−12 | 8.34E−06 ± 8.22E−06 | **0.00E+00 ± 0.00E+00** |
| $D = 10$ | 2.36E+01 ± 3.80E+01 | 2.80E−02 ± 1.27E−02 | 4.73E+02 ± 1.52E+02 | 3.48E−10 ± 1.36E−10 | 2.92E+03 ± 1.52E+03 | **0.00E+00 ± 0.00E+00** |
| $D = 30$ | 9.67E+02 ± 6.25E+02 | 3.14E−01 ± 8.02E−02 | 1.16E+04 ± 2.08E+03 | 4.10E−09 ± 5.91E−10 | 5.52E+04 ± 6.00E+03 | **0.00E+00 ± 0.00E+00** |
| $D = 50$ | 5.95E+03 ± 1.73E+03 | 1.44E+00 ± 2.40E−01 | 3.83E+04 ± 3.90E+03 | 1.06E−08 ± 1.68E−09 | 1.11E+05 ± 1.06E+04 | **2.27E−14 ± 3.11E−14** |
| $D = 100$ | 3.35E+04 ± 6.61E+03 | 1.10E+01 ± 6.87E+00 | 1.37E+05 ± 5.34E+03 | 4.35E−08 ± 2.46E−09 | 2.70E+05 ± 7.71E+03 | **5.68E−14 ± 1.12E−14** |
| $D = 1000$ | 1.21E+06 ± 1.65E+05 | 5.81E+04 ± 1.83E+03 | 1.66E+06 ± 6.74E+05 | 6.15E−05 ± 4.21E−04 | 3.23E+06 ± 1.35E+05 | **2.84E−11 ± 1.28E−11** |




(a) case 1 (no-operation) (b) case 2 (shifted search space) (c) case 3 (shifted center) (d) case 4 (scaled search space)

(e) Convergence curve of case 1 (f) Convergence curve of case 2 (g) Convergence curve of case 3 (h) Convergence curve of case 4

**Fig. 5.** The 3D views of the four cases and convergence curves obtained by the algorithms.

### 3.2. Scalability analysis

Scalability assessment is used to study the impact of dimensionality changes on algorithm performance. In this section, the standard shifted sphere function from the CEC'05 test suite is used, instead of the cases in the previous section. As conclusions have already been drawn from the two-dimensional four cases, it can be inferred that the well-known algorithms cannot converge better on the four cases as the dimensionality increases. The algorithm selection remains unchanged in this section. Please refer to Table 2 for further details. For each specific condition, every algorithm employs the following parameter settings: $run = 30$, $N = 40$, $D = 2/10/30/50/100/1000$, and $MaxFEs = D \times 10,000$. One exception is that for $D = 1000$, $MaxFEs$ is set to $3 \times 10^6$ based on the large-scale benchmark criterion.

Table 4 and Fig. 6 show the results of different dimensions. It is clear that AE achieves the best performance in the six different dimensions, and its convergence behavior is hardly affected by changes in dimension. Undeniably, SSA also performs excellently, but as described in the previous section, this performance deteriorates with the shifting or scaling of the effective search space. Moreover, the operation of shifting the center severely limits the performance of the remaining algorithms. Based on the aforementioned experiments and discussions, the conclusion is drawn as follows (Fig. 11 provides supporting evidence for this inference): AE has the scale invariance for dimension.

**Table 5**
The parameter sensitivity results based on the Friedman test.

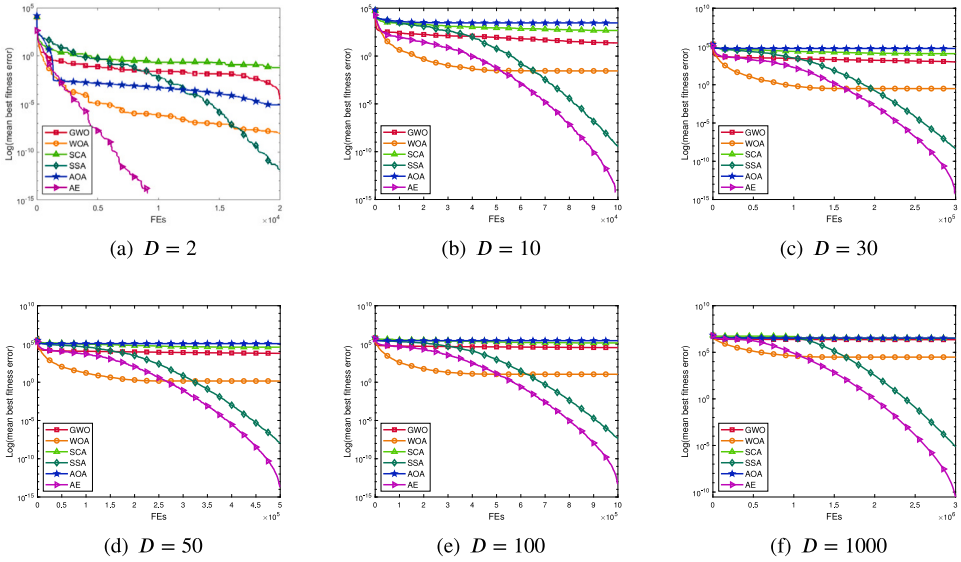
| Parameter | $D = 10$ | $D = 30$ | $D = 50$ | $D = 100$ | Mean rank |
|---|---|---|---|---|---|
| $N = 10$ | 3.30 | 3.03 | 3.33 | 3.13 | 3.1975 |
| $N = 40$ | 2.07 | 2.33 | 2.17 | 2.43 | 2.2500 |
| $N = 70$ | 2.10 | 2.50 | 2.30 | 2.20 | 2.2750 |
| $N = 100$ | 2.53 | 2.13 | 2.20 | 2.23 | 2.2725 |

### 3.3. Parameter sensitivity analysis

The number of candidate solutions $N$ is the only hyperparameter of AE. To analyze the impact of this parameter on AE's performance, different values ($N = 10/40/70/100$) are selected and tested them with 30 benchmark functions from the CEC' 17 test suite. The experiments were carried out with $D = 10/30/50/100$ and $MaxFEs = 10,000 \times D$, and the results were obtained through 30 independent executions. The statistical ranking was calculated by the Friedman test (Wilcoxon, 1992).

From Table 5, it can be seen that when $N$ is 10, the average performance of AE is relatively poor. However, as the parameter value increased, the average performance of AE seemed to no longer change. This indicates that the proposed algorithm is no longer sensitive to the setting of the parameter $N$ when the parameter value controllably increases.

Sensitivity analysis methods are widely used in artificial intelligence, such as feature selection and updating neural networks (Asheghi




(a) $D = 2$ (b) $D = 10$ (c) $D = 30$

(d) $D = 50$ (e) $D = 100$ (f) $D = 1000$

**Fig. 6.** The convergence curves of AE and the well-known algorithms on the shifted sphere function in different dimensions.

**Table 6**
The impact of varying parameter values on average fitness error and standard deviation.

| No. | Parameter | $D = 10$ | $D = 30$ | $D = 50$ | $D = 100$ |
|---|---|---|---|---|---|
| CEC'17 F1 | $N = 10$ | 1.88E+03 ± 2.06E+03 | 4.99E+03 ± 4.94E+03 | 1.06E+04 ± 1.10E+04 | 5.85E+03 ± 8.29E+03 |
| | $N = 40$ | 6.96E+01 ± 5.72E+01 | 1.27E+03 ± 9.16E+02 | 1.96E+03 ± 1.12E+03 | 3.70E+03 ± 3.79E+03 |
| | $N = 70$ | 3.32E+02 ± 5.44E+02 | 1.64E+03 ± 1.85E+03 | 6.60E+03 ± 4.40E+03 | 4.39E+03 ± 3.75E+03 |
| | $N = 100$ | 2.76E+02 ± 5.09E+02 | 1.82E+03 ± 1.88E+02 | 5.84E+03 ± 5.95E+03 | 3.87E+03 ± 3.32E+03 |
| CEC'17 F10 | $N = 10$ | 3.45E+02 ± 2.06E+02 | 1.96E+03 ± 6.97E+02 | 5.39E+03 ± 1.11E+03 | 9.97E+03 ± 2.60E+03 |
| | $N = 40$ | 1.58E+02 ± 1.79E+02 | 2.77E+03 ± 5.55E+02 | 4.66E+03 ± 1.05E+03 | 9.51E+03 ± 9.35E+02 |
| | $N = 70$ | 2.35E+02 ± 1.59E+02 | 2.18E+03 ± 4.48E+02 | 5.01E+03 ± 9.00E+02 | 1.13E+04 ± 1.27E+03 |
| | $N = 100$ | 3.16E+02 ± 3.45E+02 | 3.74E+03 ± 1.99E+03 | 5.04E+03 ± 7.58E+02 | 1.28E+04 ± 6.40E+02 |
| CEC'17 F20 | $N = 10$ | 7.10E+00 ± 9.29E+00 | 1.00E+02 ± 1.08E+02 | 4.89E+02 ± 1.88E+02 | 1.72E+03 ± 9.58E+02 |
| | $N = 40$ | 6.96E+00 ± 9.90E+00 | 7.91E+01 ± 6.60E+01 | 1.60E+02 ± 2.58E+01 | 1.06E+03 ± 3.66E+02 |
| | $N = 70$ | 1.11E+01 ± 1.05E+01 | 1.01E+02 ± 6.43E+01 | 3.66E+02 ± 1.86E+02 | 1.73E+03 ± 3.65E+02 |
| | $N = 100$ | 2.67E+00 ± 6.33E+00 | 1.39E+02 ± 5.08E+01 | 1.87E+02 ± 9.57E+01 | 1.94E+03 ± 1.22E+02 |
| CEC'17 F30 | $N = 10$ | 7.05E+02 ± 1.23E+02 | 8.54E+03 ± 4.30E+03 | 1.09E+06 ± 2.86E+05 | 7.68E+03 ± 2.31E+03 |
| | $N = 40$ | 4.53E+02 ± 6.82E+01 | 3.36E+03 ± 1.12E+03 | 6.01E+05 ± 1.36E+04 | 3.33E+03 ± 2.18E+02 |
| | $N = 70$ | 4.95E+02 ± 1.94E+01 | 3.00E+03 ± 7.52E+02 | 7.20E+05 ± 1.99E+05 | 3.34E+03 ± 3.42E+02 |
| | $N = 100$ | 4.97E+02 ± 5.98E+01 | 4.16E+03 ± 2.05E+03 | 6.44E+05 ± 4.57E+04 | 3.71E+03 ± 7.22E+02 |

et al., 2020). Therefore, another parameter sensitivity analysis method is also used in this experiment. Four different structural problems (unimodal function CEC'17 F1, multimodal function CEC'17 F10, hybrid function CEC'17 F20, and composition function CEC'17 F30) were selected to test the impact of the parameter $N$ changes on AE's performance, and the test results are shown in Table 6.

From Table 6, it can be seen that the results obtained by AE are relatively excellent when $N = 40$. Furthermore, in most cases, changes in parameter $N$ do not significantly affect the results. However, this change can still lead to unstable phenomena on a few issues.

### 3.4. Search behavior analysis

Search behavior refers to the search strategies and actions taken by an algorithm to find the optimal solution. Different intelligent optimization algorithms may exhibit different search behaviors. In this section, three indicators are used to comprehensively analyze the search behavior of AE: 1-dimensional search history, 2-dimensional search history, and 3-dimensional search history. The proposed algorithm is applied to solve the sphere function, the schwefel 2.22 function, the ackley function, and the levy function. For each function, AE employs 80 candidate solutions and performs 100 iterations. Samples are collected at the 1st, 30th, 60th, and 100th generations to obtain the corresponding indicators.

The first metric in Fig. 7 is the 1D search history. The objective function values of all solutions and their corresponding 1D positions are displayed on the vertical contour of the 2D function. At the beginning of the iterations, the positions of the solutions are relatively dispersed. However, as the iterations progress, the quality of all solutions improves, and they converge toward the global minimum. This behavior indicates that AE retains the advantages of population-based algorithms and demonstrates its convergence capability.

The second metric in Fig. 7 is the 2D search history, where the contour lines transition from red to blue, indicating a decrease in the objective function values. At the beginning of the iterations, the individuals are uniformly distributed in the search space. As the iterations progress, all solutions converge toward the global optimal region, illustrating the algorithm's transition from global exploration to local exploitation. In particular, the search trajectory of a specific solution is also shown, which displays its historical path throughout the iteration process. It is evident that no matter how the path initially deviates from the global optimal region, it eventually points toward the global optimum. On the one hand, this deviation in the path helps prevent premature convergence for AE, and on the other hand, it increases the chances of finding more promising regions.

The third metric in Fig. 7 is the 3D search history. Analyzing this metric helps to understand the complexity of the problem structure. Similar to the previous metrics, the color of the contour line is closely

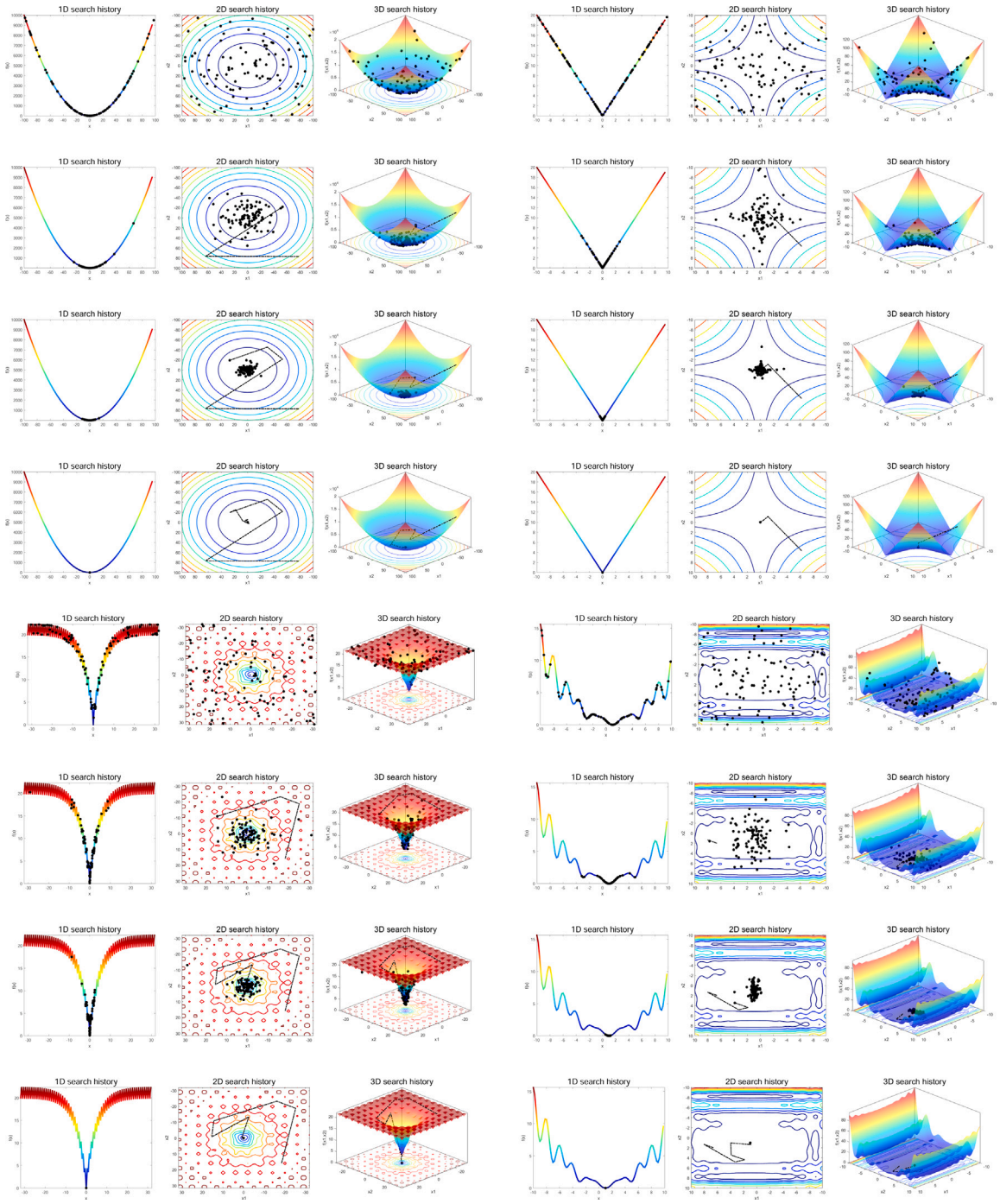



**Fig. 7.** The 1D, 2D, and 3D search histories of the AE algorithm on four classical functions.

related to the objective function value. Unlike the paths depicted in the previous metric, this metric introduces the concept of height to the paths. It is clear that regardless of the deviations in the paths, their height decreases, indicating a tendency for the solutions to converge toward the optimal solution. Regardless of the initial position of the solutions in the problem space, they eventually converge to the global optimal position.

### 3.5. Qualitative analysis

In this section, five qualitative metrics are used to study AE's search patterns and behaviors. These metrics include the search history, the search trajectory, the optimization history, the average fitness history, and the trajectory in the 1st dimension. All metrics were obtained through testing on CEC benchmark functions. Moreover, the experimental settings are $N = 40$, $D = 2$, and $MaxFEs = D \times 10,000$.

In the first column of Fig. 8, the 3D view of the functions is shown. It can be observed that these problems have distinct topological structures and are all challenging. The second column of Fig. 8 represents the search trajectory, illustrating the search history path of a particular solution. The path is highlighted in the two-dimensional search space to better demonstrate the search behavior and trends of the solution. The paths on the CEC'17 F3 and the CEC'14 F16 indicate the behavior of solutions rapidly approaching the global optimum regions. On other problems, the paths cover a larger range and appear slightly more complex. This pattern demonstrates that the AE algorithm exhibits good

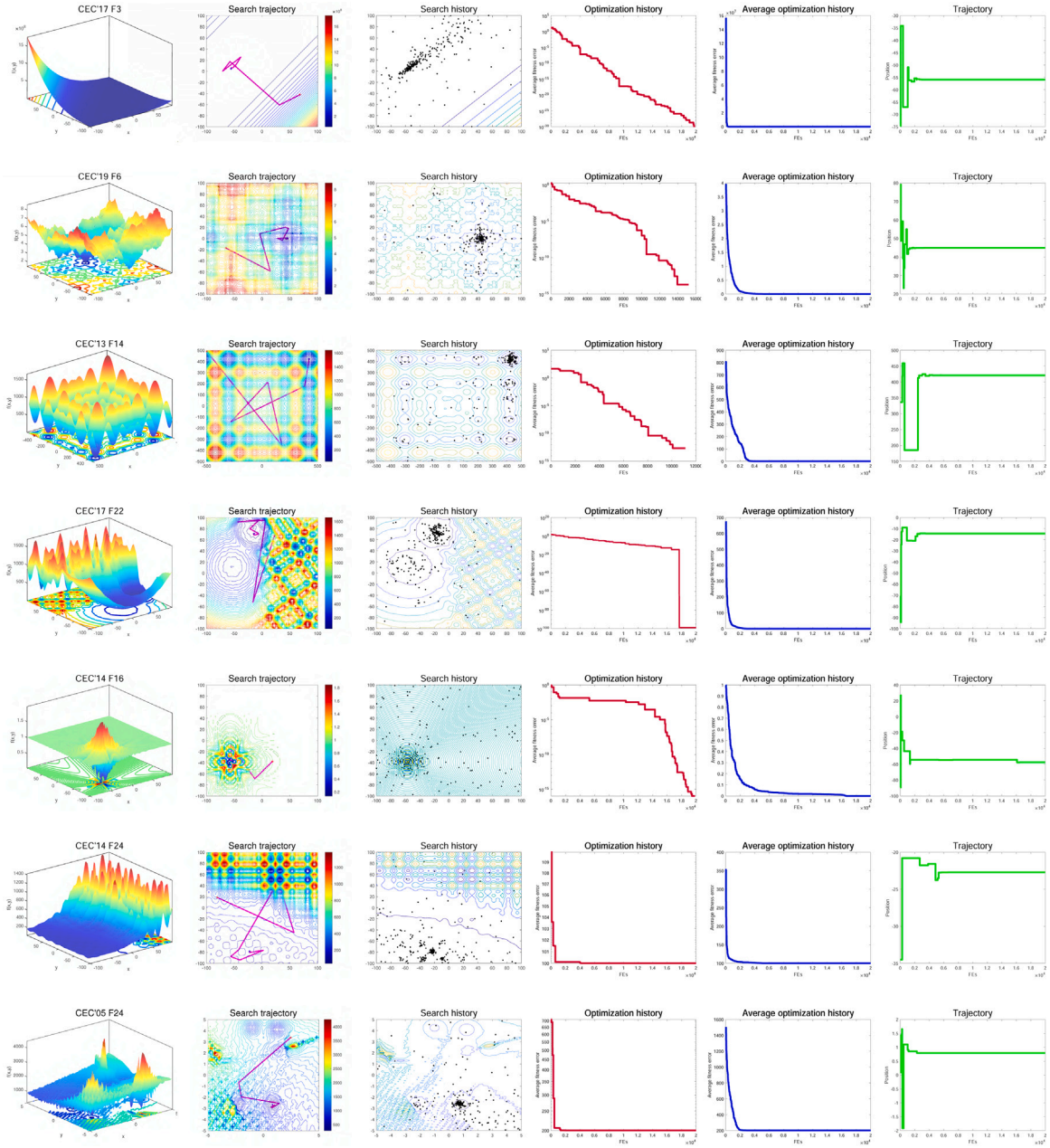



**Fig. 8.** The qualitative metrics: search history, search trajectory, optimization history, average fitness history, and trajectory in 1st dimension.

global exploration behavior. With iterations, these paths eventually converge to the global optimal region. In particular, on the CEC'14 F24, the trajectory demonstrates how a solution, initially trapped in a local optimum region, manages to escape and explore a more promising area.

The third column of Fig. 8 represents the search history, including all solutions' historical updated positions. New positions are continually generated in the search space to better present how solutions explore and exploit the space. At the beginning of the iterations, the solutions are distributed across various positions in the search space for global exploration. As the iterations progress, these solutions gradually converge to the global optimal position for exploitation. Particularly, on the CEC'14 F24, AE's multi-point exploitation behavior is demonstrated. This indicates that the proposed algorithm can effectively balance its exploration and exploitation capabilities.

The fourth column of Fig. 8 represents the optimization history; it displays the error change between the algorithm's current optimal solution and the problem's theoretical optimal solution. It is evident that all curves are decreasing, indicating that the proposed algorithm exhibits good convergence. Different optimization history characteristics exist for different types of functions. For the CEC'17 F3, the CEC'19 F6, the CEC'13 F14, the CEC'17 F22, and the CEC'14 F16, the convergence curves gradually decrease to prevent premature convergence and maintain a balanced search. On the other hand, for other functions, the convergence curves seem to decrease more rapidly. However, when combined with the first two indicators, it can be observed that AE already found the global optimum solutions in the early iterations. This further leads to convergence curves that remain unchanged as the iterations progress.

The fifth column of Fig. 8 represents the average fitness history of all solutions, which differs from the optimization history in that it describes the average quality change of all solutions. It is evident that all curves exhibit similar trends, smoothly and steadily decreasing, indicating that all solutions are improving. Furthermore, there is a close similarity between the average fitness history and the optimization

history. This similarity suggests that whether it is the current best solution or all candidate solutions, they all progress toward better positions.

The sixth column of Fig. 8 represents the trajectory in 1st dimension, depicting the positional changes of a specific solution along the first dimension. In the initial search process of the algorithm, all curves exhibit significant and drastic oscillations. These sudden changes indicate the exploratory behavior of the algorithm. In the subsequent iterations, the curves gradually stabilize, indicating a transition from global exploration to local exploitation by the algorithm.

### 3.6. Exploration and exploitation analysis

In population-based optimization algorithms, exploration and exploitation are often hidden (Cai et al., 2022). To visually present the exploration and exploitation behavior of the proposed algorithm and conduct analysis, dimension-wise diversity measurement (Cheng et al., 2014; Morales-Castañeda et al., 2020) is considered. According to this method, the diversity metric $Div$ is defined as Eqs. (14) and (15):

$$Div_j = \frac{1}{N}\sum_{i=1}^{N}\left|median_j - X_{i,j}\right| \tag{14}$$

$$Div^t = \frac{1}{D}\sum_{j=1}^{D} Div_j \tag{15}$$

where $median_j$ represents the median of dimension $j$ in $\mathbf{X}$, and $Div_j$ represents the diversity of dimension $j$ in $\mathbf{X}$. After the iteration process is completed, the percentages of exploration and exploitation invested by the algorithm are calculated in the $t$th iteration using Eqs. (16) and (17):

$$Exploration^t = \frac{Div^t}{Div_{\max}} \tag{16}$$

$$Exploitation^t = \frac{\left|Div_{\max} - Div^t\right|}{Div_{\max}} \tag{17}$$

where $Div_{max}$ is the largest $Div$ in the entire iteration process.

In this experiment, the unimodal function CEC'17 F1, the multimodal function CEC'17 F10, the hybrid function CEC'17 F20, and the composition function CEC'17 F30 from the CEC'17 test suite were considered to evaluate the exploration and exploitation behavior of the AE algorithm. Furthermore, four variations in problem dimensions were considered to investigate the influence of dimension scalability on this behavior. The algorithmic termination criterion $MaxFEs$ for each problem was set as $D\times 10,000$, and $N$ was set to 40. Fig. 9 illustrates the experimental results, where the $x$-axis represents $FEs$, and the $y$-axis represents the percentage of exploration or exploitation invested by the AE algorithm.

From Fig. 9, it can be observed that on the CEC'17 F1, the proposed algorithm primarily engages in exploitation behavior, which dominates the later iterations of the algorithm. Conversely, exploration behavior is less prominent. Furthermore, for this particular function, the algorithmic exploration and exploitation behaviors are less affected by changes in dimensionality. This characteristic becomes more pronounced on the CEC'17 F10 and the CEC'17 F20. Due to the complex topology of these two problems, exploration behavior is rapidly attenuated in the mid-iterations, possibly due to the presence of numerous local optima that impact the search behavior. Exploitation behavior predominates as all solutions converge to local optimal zones. The slow changes in exploration and exploitation on the CEC'17 F30 for $D = 10$ show that the solutions continuously investigate several local optima regions. This also suggests that the algorithm does not suffer from search stagnation. For other dimensions, the search pattern seems similar. In conclusion, the proposed algorithm demonstrated a good balance between exploration and exploitation behaviors, with less influence from changes in dimensionality. This is achieved by adjusting the parameter $\alpha$, which is crucial in balancing its search capabilities.

### 3.7. Compare AE with other 100 optimization algorithms

To test and validate the optimization performance of the proposed algorithm, this section conducts a comprehensive evaluation of AE on the challenging CEC'17 test suite. AE is compared with 100 intelligent optimization algorithms for various problem dimensions ($D = 30$, 50, and 100). These algorithms include classical and novel algorithms widely validated through experiments or applications.

#### 3.7.1. Description of the benchmark functions

Search bias can give algorithms an unfair advantage, which has led to critical discussions (Castelli et al., 2022; Pickard et al., 2016; Niu et al., 2019). In the CEC' 17 test suite (Wu et al., 2017), all functions are shifted and rotated, rendering search bias ineffective (Gao et al., 2024). These characteristics harm the prior knowledge of the problem, making it more difficult to find promising gradients or optimal solutions, thus making artificial problems closer to real problems (Caraffini and Neri, 2019). Therefore, 30 benchmark functions from the CEC' 17 test suite are used to evaluate relevant algorithms. These functions are divided into unimodal functions (CEC'17 F1–F3), multimodal functions (CEC'17 F4–F10), hybrid functions (CEC'17 F11–F20), and composition functions (CEC'17 F21–F30). The effective search ranges for these test problems are defined as $[-100, 100]$.

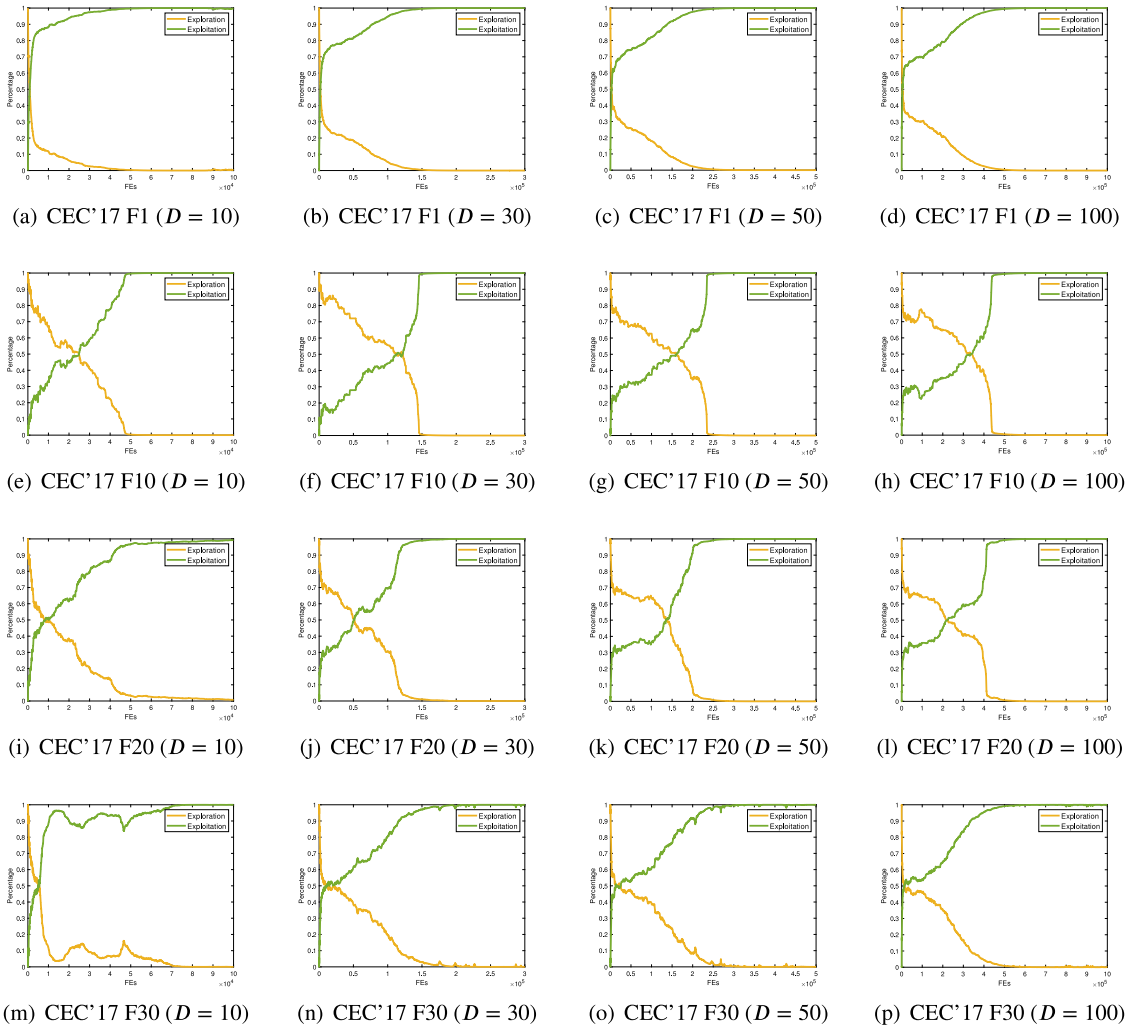
Fig. 10 illustrates the 3D views of selected functions from the CEC'17 test suite. Due to their distinct characteristics, excessively superior performance on one problem may be nullified by the characteristics of another problem. Furthermore, since hybrid functions divide variables into different groups and optimize different sub-functions using them, it is not feasible to depict their 3D views using two-dimensional variables.

#### 3.7.2. Experimental setup and involved algorithms

In the numerical optimization experiment, the number of competitors for AE was set to 100 to evaluate its performance comprehensively. The unified parameter settings for all algorithms are $N = 40$, $D = 30/50/100$. For each algorithm, it is run independently 30 times on each problem. The maximum fitness evaluations ($MaxFEs = 10,000 \times D$) were used as the termination criterion for the algorithms. Table 7 presents the parameter settings for the 100 algorithms involved in the experiments. Moreover, these metaphorical algorithms from different sources of inspiration or metaphor-less algorithms are proven to be efficient, so they pose challenges for AE.

#### 3.7.3. Convergence results and analysis

This section analyzes convergence using four representative functions, including the multimodal functions CEC'17 F5 and CEC'17 F8, the hybrid function CEC'17 F17, and the composition function CEC'17 F24. From Fig. 11, it can be observed that due to the presence of numerous local optima on the CEC'17 F5 and the CEC'17 F8, many algorithms fail to converge early in the iterations. However, AE under the control of parameter $\alpha$ avoids falling into local optima, enabling rapid convergence during the mid-iterations and demonstrating advantages. Certainly, AEFA and CMA-ES also demonstrate excellent optimization capabilities for these two problems, but they appear to exhibit premature convergence behavior. On the hybrid function CEC'17 F17, AE does not exhibit significant advantages at $D = 30$. However, as the dimensionality increases, AE shows better convergence. For other algorithms, at $D = 100$, the increased exploration space of the problem leads to a significant divergence among the algorithms. For the composition function CEC'17 F24, most algorithms yield results within the same order of magnitude. It is noteworthy that HMO demonstrates advantages at $D = 30$. However, its performance seems to deteriorate as the dimension increases. On the contrary, although AE falls behind HMO at $D = 30$, it surpasses HMO at $D = 50$ and $D = 100$, achieving the best convergence results.




(a) CEC'17 F1 ($D = 10$) (b) CEC'17 F1 ($D = 30$) (c) CEC'17 F1 ($D = 50$) (d) CEC'17 F1 ($D = 100$)

(e) CEC'17 F10 ($D = 10$) (f) CEC'17 F10 ($D = 30$) (g) CEC'17 F10 ($D = 50$) (h) CEC'17 F10 ($D = 100$)

(i) CEC'17 F20 ($D = 10$) (j) CEC'17 F20 ($D = 30$) (k) CEC'17 F20 ($D = 50$) (l) CEC'17 F20 ($D = 100$)

(m) CEC'17 F30 ($D = 10$) (n) CEC'17 F30 ($D = 30$) (o) CEC'17 F30 ($D = 50$) (p) CEC'17 F30 ($D = 100$)

**Fig. 9.** The average balance employed by AE on the unimodal function CEC'17 F1, the multimodal function CEC'17 F10, the hybrid function CEC'17 F20, and the composition function CEC'17 F30.

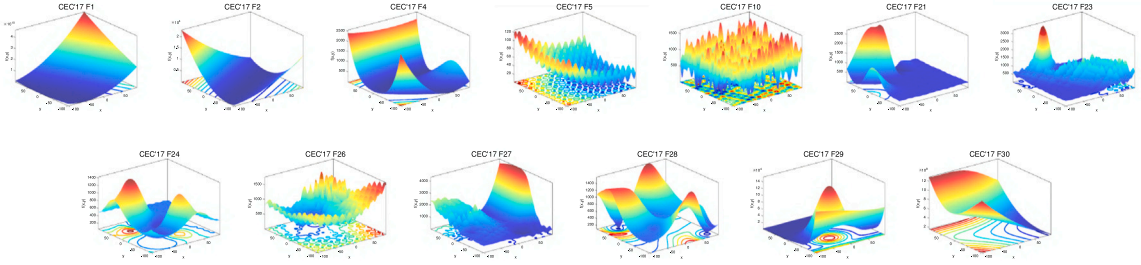



**Fig. 10.** The 3D views of some benchmark functions on the CEC'17 test suite.

Through the analysis above, the performance of AE appears to be less affected by changes in dimensionality compared to other algorithms. Furthermore, due to the effective balance of exploration and exploitation achieved by parameter $\alpha$, AE can effectively solve different problems. The evolutionary paths connect evolutionary information from different generations to locate an effective initial search point for the algorithm, which provides a particular advantage in approximating the optimal solution.

### 3.7.4. Statistical results and analysis

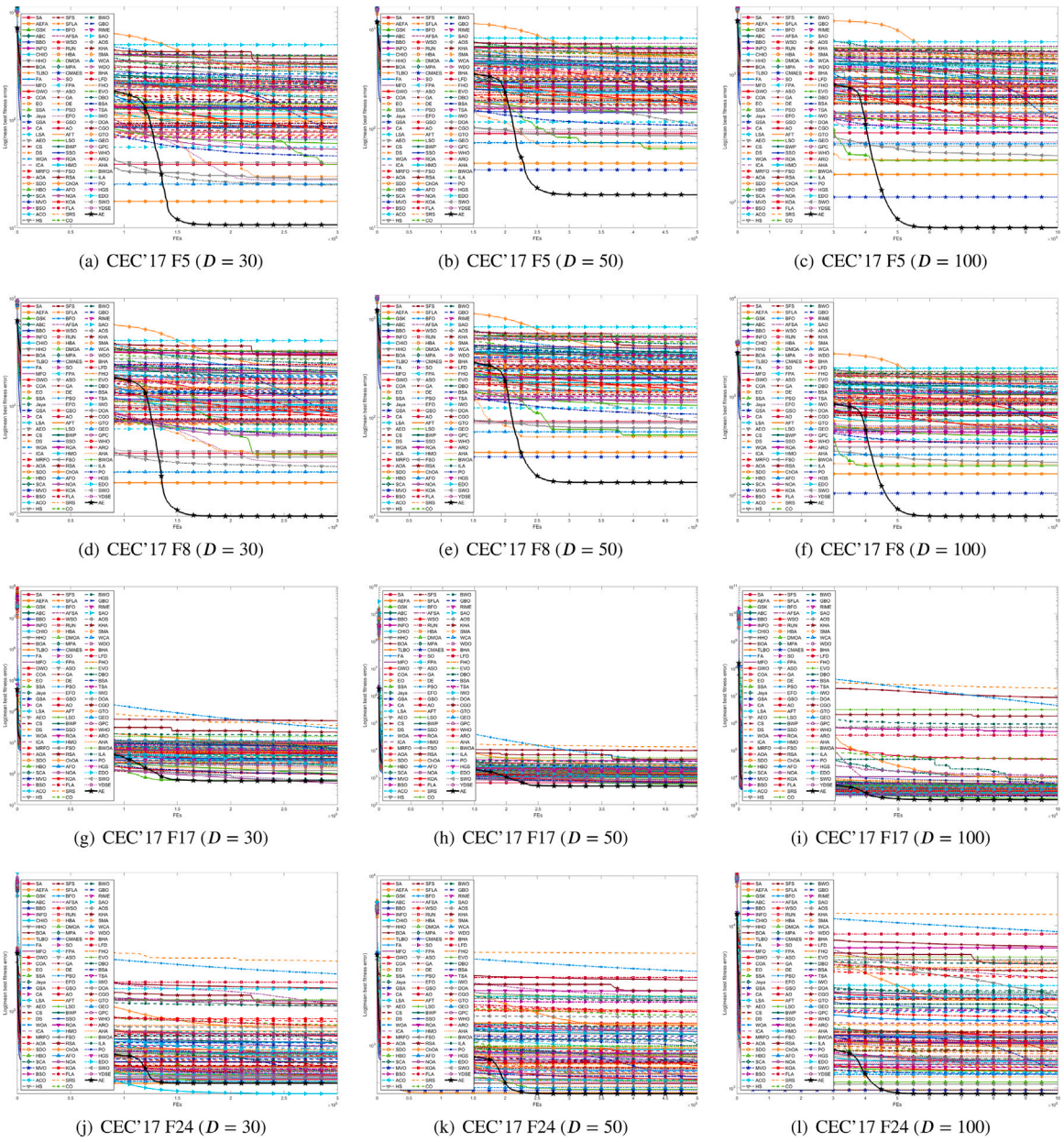
This section uses box plots, the Friedman test (Friedman, 1940, 1937), and the Wilcoxon rank sum test (Derrac et al., 2011) to statistically analyze the experimental results. Box plots are commonly used to represent the discrete distribution of data. In Fig. 12, box plots are provided for the four representative functions mentioned in the previous section. They display the result distributions of all algorithms when solving these functions, aiming to analyze the stability of the algorithms. From Fig. 12, it can be observed that most algorithms appear unstable on the CEC'17 F5 and the CEC'17 F8, possibly due to them getting trapped in different local optima. Additionally, the box plots of AE are positioned the lowest on these problems, indicating that it is more stable than other algorithms. However, box plots of AE also show that its results are more dispersed, demonstrating a certain level of controlled instability. This instability seems to diminish as dimensionality increases. On the CEC'17 F17, as the problem dimensionality increases, the data obtained by all algorithms appears to gradually concentrate,

**Table 7**
The algorithm information and parameter settings.

| No. | Algorithms | Proposed time | Parameters Settings | References |
|---|---|---|---|---|
| 1 | Genetic algorithm (GA) | 1975 | $pc = 0.8, pm = 0.2$ | Sampson (1976) |
| 2 | Simulated annealing (SA) | 1983 | $T_0 = 0.1, \alpha = 0.99, c = 5, \mu = 0.5$ | Kirkpatrick et al. (1983) |
| 3 | Ant colony optimization (ACO) | 1992 | $q = 0.5, \zeta = 1$ | Dorigo (1992) |
| 4 | Cultural algorithm (CA) | 1994 | $pAccept = 0.35, \alpha = 0.3$ | Reynolds (1994) |
| 5 | Particle swarm optimization (PSO) | 1995 | $w = 1, w_p = 0.99, c_1 = 1.5, c_2 = 2.0$ | Kennedy and Eberhart (1995) |
| 6 | Differential evolution (DE) | 1997 | $F = 0.5, CR = 0.9$ | Storn and Price (1997) |
| 7 | Harmony search (HS) | 2001 | $HMCR = 0.9, PAR = 0.1, FW_{damp} = 0.995$ | Geem et al. (2001) |
| 8 | Covariance matrix adaptation evolution strategy (CMA-ES) | 2003 | $\sigma = 0.3$ | Hansen et al. (2003) |
| 9 | Shuffled frog-leaping algorithm (SFLA) | 2006 | $Memeplexes = 5, Offsprings = 3, Step = 2, Run = 5$ | Eusuff et al. (2006) |
| 10 | Artificial bee colony (ABC) | 2007 | $a = 1$ | Karaboga and Basturk (2007) |
| 11 | Biogeography-based optimization (BBO) | 2008 | $KeepRate = 0.2, \alpha = 0.9, Mutation = 0.1$ | Simon (2008) |
| 12 | Firefly algorithm (FA) | 2009 | $\alpha = 0.2, beta = 2, \gamma = 1, DampingRatio = 0.98$ | Yang (2009) |
| 13 | Cuckoo search (CS) | 2009 | $Pa = 0.25, \alpha = 1, \beta = 1.5, \sigma_v = 1$ | Yang and Deb (2009) |
| 14 | Gravitational search algorithm (GSA) | 2009 | $G_0 = 100, \alpha = 20, Rpower = 1$ | Rashedi et al. (2009) |
| 15 | Group search optimizer (GSO) | 2009 | $np = 1, ns = 0.8, st = 50, ve = 1$ | He et al. (2009) |
| 16 | Bacterial foraging optimization (BFO) | 2010 | $c = 25, s = 4, r = 4, p = 0.5, Sr = N/2$ | Das et al. (2009) |
| 17 | Wind driven optimization (WDO) | 2010 | $RT = 3, g = 0.2, a = 0.4, C = 0.4, maxv = 0.3$ | Bayraktar et al. (2010) |
| 18 | Invasive weed optimization (IWO) | 2010 | $nPop0 = 10, Smin = 0, Smax = 5, Exponent = 2, \sigma \in [0.001, 0.05]$ | Karimkashi and Kishk (2010) |
| 19 | Brain storm optimization (BSO) | 2011 | $m = 5, p_{5a} = 0.2, p_{6b} = 0.8, p_{6bi} = 0.4, p_{6c} = 0.5, k = 20, \mu = 0, \sigma = 1$ | Shi (2011) |
| 20 | Teaching-learning-based optimization (TLBO) | 2011 | $T_F = 1or2$ | Rao et al. (2011) |
| 21 | Differential search (DS) | 2012 | $p_1 = 0.3, p_2 = 0.3$ | Civicioglu (2012) |
| 22 | Flower pollination algorithm (FPA) | 2012 | $p = 0.8$ | Yang (2012) |
| 23 | Krill herd (KH) | 2012 | $Vf = 0.02, Dmax = 0.005, Nmax = 0.01$ | Gandomi and Alavi (2012) |
| 24 | Water cycle algorithm (WCA) | 2012 | $Nsr = 4, dmax = 1E-16$ | Eskandar et al. (2012) |
| 25 | Social spider optimization (SSO) | 2013 | $fp \in [0.65, 0.9]$ | Cuevas et al. (2013) |
| 26 | Backtracking search optimization algorithm (BSA)) | 2013 | $DIM_{RATE} = 1$ | Civicioglu (2013) |
| 27 | Artificial fish swarm algorithm (AFSA) | 2014 | $T = 100, visual = 0.5, step = 2, \delta = 0.618$ | Neshat et al. (2014) |
| 28 | Imperialist competitive algorithm (ICA) | 2014 | $nEmp = 10, \alpha = 1, \beta = 1.5, pR = 0.05, \mu = 0.1, \zeta = 0.2$ | Atashpaz-Gargari and Lucas (2007) |
| 29 | Grey wolf optimizer (GWO) | 2014 | $\alpha \in [0, 2]$ | Mirjalili et al. (2014) |
| 30 | Black-hole algorithm (BHA) | 2014 | $landa \in [0, 1]$ | Bouchekara (2014) |
| 31 | Moth-flame optimization (MFO) | 2015 | $r \in [-2, -1], b = 1$ | Mirjalili (2015) |
| 32 | Lightning search algorithm (LSA) | 2015 | $maxchtime = 10$ | Shareef et al. (2015) |
| 33 | Stochastic fractal search (SFS) | 2015 | $MDN = 2, Walk = 1$ | Salimi (2015) |
| 34 | Tree-seed algorithm (TSA) | 2015 | $ST = 0.1, low_r = 0.1, up_r = 0.25$ | Kiran (2015) |
| 35 | Whale optimization algorithm (WOA) | 2016 | $b = 1, a_1 = \in [0, 2], a_2 = \in [-2, -1]$ | Mirjalili and Lewis (2016) |
| 36 | Jaya | 2016 | No special parameters | Rao (2016) |
| 37 | Sine cosine algorithm (SCA) | 2016 | $a = 2$ | Mirjalili (2016) |
| 38 | Multi-verse optimizer (MVO) | 2016 | $WEP_{Max} = 1, WEP_{Min} = 0.2$ | Mirjalili et al. (2016) |
| 39 | Electromagnetic field optimization (EFO) | 2016 | $R = 0.3, Ps = 0.2, P = 0.1, N = 0.45$ | Abedinpourshotorban et al. (2016) |
| 40 | Salp swarm algorithm (SSA) | 2017 | $c_1 \in [0, 2]$ | Mirjalili et al. (2017) |
| 41 | Coyote optimization algorithm (COA) | 2018 | $N_c = 5, N_p = 8$ | Pierezan and Coelho (2018) |
| 42 | Atom search optimization (ASO) | 2019 | $\alpha = 50, \beta = 0.2$ | Zhao et al. (2019a) |
| 43 | Artificial electric field algorithm (AEFA) | 2019 | $K_0 = 500, \alpha = 30$ | Yadav et al. (2019) |
| 44 | Harris hawks optimization (HHO) | 2019 | $E_0 \in [-1, 1], E_1 = \in [0, 2]$ | Heidari et al. (2019) |
| 45 | Butterfly optimization algorithm (BOA) | 2019 | $P = 0.8, \alpha = 0.1, c = 0.01$ | Arora and Singh (2019) |
| 46 | Artificial ecosystem-based optimization (AEO) | 2019 | $a \in [0, 1]$ | Zhao et al. (2020a) |
| 47 | Supply–demand-based optimization (SDO) | 2019 | $a_1 \in [0, 2]$ | Zhao et al. (2019b) |
| 48 | Manta ray foraging optimization (MRFO) | 2020 | $S = 2, r_1 \in [0, 1], Coef \in [0, 1]$ | Zhao et al. (2020b) |
| 49 | Gaining sharing knowledge (GSK) | 2020 | $p = 0.1, k_r = 0.9, k_f = 0.5$ | Mohamed et al. (2020) |
| 50 | Equilibrium optimizer (EO) | 2020 | $V = 1, a_1 = 2, a_2 = 1, GP = 0.5$ | Faramarzi et al. (2020b) |
| 51 | Coronavirus herd immunity optimizer (CHIO) | 2020 | $MaxAge = 100, CO = 1, BRr = 0.05$ | Al-Betar et al. (2021) |
| 52 | Heap-based optimizer (HBO) | 2020 | $cycles = \lfloor MaxIt/25 \rfloor, degree = 3$ | Askari et al. (2020a) |
| 53 | Marine predators algorithm (MPA) | 2020 | $FADs = 0.2, P = 0.5$ | Faramarzi et al. (2020a) |
| 54 | Gradient-based optimizerm (GBO) | 2020 | $pr = 0.5$ | Ahmadianfar et al. (2020) |
| 55 | Best-worst-play (BWP) | 2020 | No special parameters | Singh et al. (2020) |
| 56 | Slime mould algorithm (SMA) | 2020 | $z = 0.03$ | Li et al. (2020) |
| 57 | Lévy flight distribution (LFD) | 2020 | $threshold = 2, CSV = 0.5, \alpha_1 = 10, \alpha_2 = 5E-5, \alpha_3 = 5E-3, \partial_1 = 0.9, \partial_2 = 0.1$ | Houssein et al. (2020) |
| 58 | Black widow optimization algorithm (BWOA) | 2020 | $pp = 0.6, pm = 0.4, cr = 0.44$ | Hayyolalam and Kazem (2020) |
| 59 | Political optimizer (PO) | 2020 | $lambda = 1.0, areas = parties = 8$ | Askari et al. (2020b) |
| 60 | Chimp optimization algorithm (ChOA) | 2020 | $a = 0.5, b = 0.2, z_1 = 0.7, z_2 = 4, z_3 = 0.6, P = 0.4, u = 1.07$ | Khishe and Mosavi (2020) |
| 61 | Arithmetic optimization algorithm (AOA) | 2021 | $MOP_{Max} = 1, MOP_{Min} = 0.2, \alpha = 5, \mu = 0.499$ | Abualigah et al. (2021a) |
| 62 | Runge kutta optimizer (RUN) | 2021 | $a = 20, b = 12$ | Ahmadianfar et al. (2021) |
| 63 | Aquila optimizer (AO) | 2021 | $\alpha = 0.1, \beta = 1.5, \delta = 0.1$ | Abualigah et al. (2021b) |
| 64 | Atomic orbital search (AOS) | 2021 | $LayerNumber = 5, FotonRate = 0.1$ | Azizi (2021) |
| 65 | Dingo Optimization Algorithm (DOA) | 2021 | $P = 0.5, Q = 0.7, \beta_1 \in [-2, 2], \beta_2 \in [-1, 1], naIni = 2$ | Peraza-Vázquez et al. (2021) |
| 66 | Chaos game optimization (CGO) | 2021 | $I = 1or2, Ir = 0or1, ii \in [1, 4]$ | Talatahari and Azizi (2021) |
| 67 | Artificial gorilla troops optimizer (GTO) | 2021 | $p = 0.03, \beta = 3, w = 0.8$ | Abdollahzadeh et al. (2021) |
| 68 | Golden eagle optimizer (GEO) | 2021 | $ap \in [0.5, 2], cp \in [0.5, 1]$ | Mohammadi-Balani et al. (2021) |
| 69 | Giza pyramids construction (GPC) | 2021 | $G = 9.8, Tetha = 14, Mu \in [1, 10], pSS = 0.5$ | Harifi et al. (2021) |
| 70 | Hunger games search (HGS) | 2021 | $VC2 = 0.03$ | Yang et al. (2021a) |
| 71 | Remora optimization algorithm (ROA) | 2021 | $C = 0.1$ | Jia et al. (2021) |
| 72 | Firebug swarm optimization (FSO) | 2021 | $m = 20, f = 5, L_1 = 34, L_2 = 11, S_1 = 82/137/275, S_2 = 153/198/220$ | Noel et al. (2021) |
| 73 | Aptenodytes forsteri optimization (AFO) | 2021 | $v = 0.25, w_2 = 0.5, w_4 = w_5 = 1, pe = 0.01, gmin = 5, dec = 2, L = 10$ | Yang et al. (2021b) |
| 74 | Reptile search algorithm (RSA) | 2022 | $\alpha = 0.1, \beta = 0.005$ | Abualigah et al. (2022) |
| 75 | Weighted mean of vector (INFO) | 2022 | $I \in [2, 5]$ | Ahmadianfar et al. (2022) |
| 76 | White shark optimizer (WSO) | 2022 | $f \in [0.07, 0.75], \tau = 4.11, p \in [0.5, 1.5], a_0 = 6.25, a_1 = 100, a_2 = 0.0005$ | Braik et al. (2022a) |
| 77 | Honey badger algorithm (HBA) | 2022 | $\beta = 6, c = 2, flag = -1or1$ | Hashim et al. (2022) |
| 78 | Dwarf mongoose optimization algorithm (DMOA) | 2022 | $b = 3, Lp = 0.6, peep = 2$ | Agushaka et al. (2022) |
| 79 | Snake optimizer (SO) | 2022 | $T_1 = 0.25, T_2 = 0.6, C_1 = 0.5, C_2 = 0.05, C_3 = 2$ | Hashim and Hussien (2022) |
| 80 | Ali Baba and the forty thieves (AFT) | 2022 | $\alpha_0 = 1, \alpha_1 = 2, \beta_0 = 0.1, \beta_1 = 2$ | Braik et al. (2022b) |
| 81 | Wild horse optimizer (WHO) | 2022 | $PS = 0.2, PC = 0.13$ | Naruei and Keynia (2022) |
| 82 | Artificial rabbits optimization (ARO) | 2022 | No special parameters | Wang et al. (2022) |
| 83 | Artificial hummingbird algorithm (AHA) | 2022 | $Mr = 2$ | Zhao et al. (2022) |
| 84 | Homonuclear molecules optimization (HMO) | 2022 | $AN = 34, NH = 6, \beta = 2$ | Mahdavi-Meymand and Zounemat-Kermani (2022) |
| 85 | Special relativity search (SRS) | 2022 | $\mu = 1$ | Goodarzimehr et al. (2022) |
| 86 | Beluga whale optimization (BWO) | 2022 | $w_f \in [0.05, 0.1]$ | Zhong et al. (2022) |
| 87 | Light spectrum optimizer (LSO) | 2022 | $red = 1.3318, violet = 1.3435, Ps = 0.05, Pe = 0.6, Ph = 0.4, B = 0.05$ | Abdel-Basset et al. (2022) |
| 88 | RIME | 2023 | $W = 5$ | Su et al. (2023) |
| 89 | Snow ablation optimizer (SAO) | 2023 | $n = 0.5$ | Deng and Liu (2023) |
| 90 | Fire hawk optimizer (FHO) | 2023 | $H_{max} = 0.2$ | Azizi et al. (2023b) |
| 91 | Energy valley optimizer (EVO) | 2023 | $threshold = 3$ | Azizi et al. (2023a) |
| 92 | Dung beetle optimizer (DBO) | 2023 | $k = 0.1, \lambda = 0.1, b = 0.3, S = 0.5$ | Xue and Shen (2023) |
| 93 | IbI logics algorithm (ILA) | 2023 | $nM = 5, m_1 = m_2 = 0.33, b_1 = 0.4, b_2 = 0.6, nR = 10, cl = 10$ | Mirrashid and Naderpour (2023) |
| 94 | Exponential distribution optimizer (EDO) | 2023 | $s = 0.5$ | Abdel-Basset et al. (2023a) |
| 95 | Spider wasp optimizer (SWO) | 2023 | $TR = 0.3, Cr = 0.2, N_{min} = 20$ | Abdel-Basset et al. (2023e) |
| 96 | Fick's law algorithm (FLA) | 2023 | $C_1 = 0.5, C_2 = 2, C_3 = 0.1, C_4 = 0.2, C_5 = 2$ | Hashim et al. (2023) |
| 97 | Young's double-slit experiment optimizer (YDSE) | 2023 | $L = 1, d = 0.005, I = 0.01, \lambda = 5E-6, \delta = 0.38$ | Abdel-Basset et al. (2023b) |
| 98 | Coati optimization algorithm (CO) | 2023 | $I \in [1, 2]$ | Dehghani et al. (2023) |
| 99 | Nutcracker optimizer (NOA) | 2023 | $\alpha = 0.05, Pa2 = 0.2, Prb = 0.2$ | Abdel-Basset et al. (2023d) |
| 100 | Kepler optimization algorithm (KOA) | 2023 | $Tc = 3, M0 = 0.1, \lambda = 15$ | Abdel-Basset et al. (2023c) |

showcasing stability in solving this problem. On the CEC'17 F24, due to the complex topology of the problem, AE and most other algorithms present flatter box plots. In short, the search performance of AE will not become unstable due to dimensional changes. However, due to the different characteristics of different problems, stable AE may also be affected.

The Friedman test is employed to calculate the rankings of all algorithms. The null hypothesis assumes that the medians of different



(a) CEC'17 F5 ($D = 30$) (b) CEC'17 F5 ($D = 50$) (c) CEC'17 F5 ($D = 100$)

(d) CEC'17 F8 ($D = 30$) (e) CEC'17 F8 ($D = 50$) (f) CEC'17 F8 ($D = 100$)

(g) CEC'17 F17 ($D = 30$) (h) CEC'17 F17 ($D = 50$) (i) CEC'17 F17 ($D = 100$)

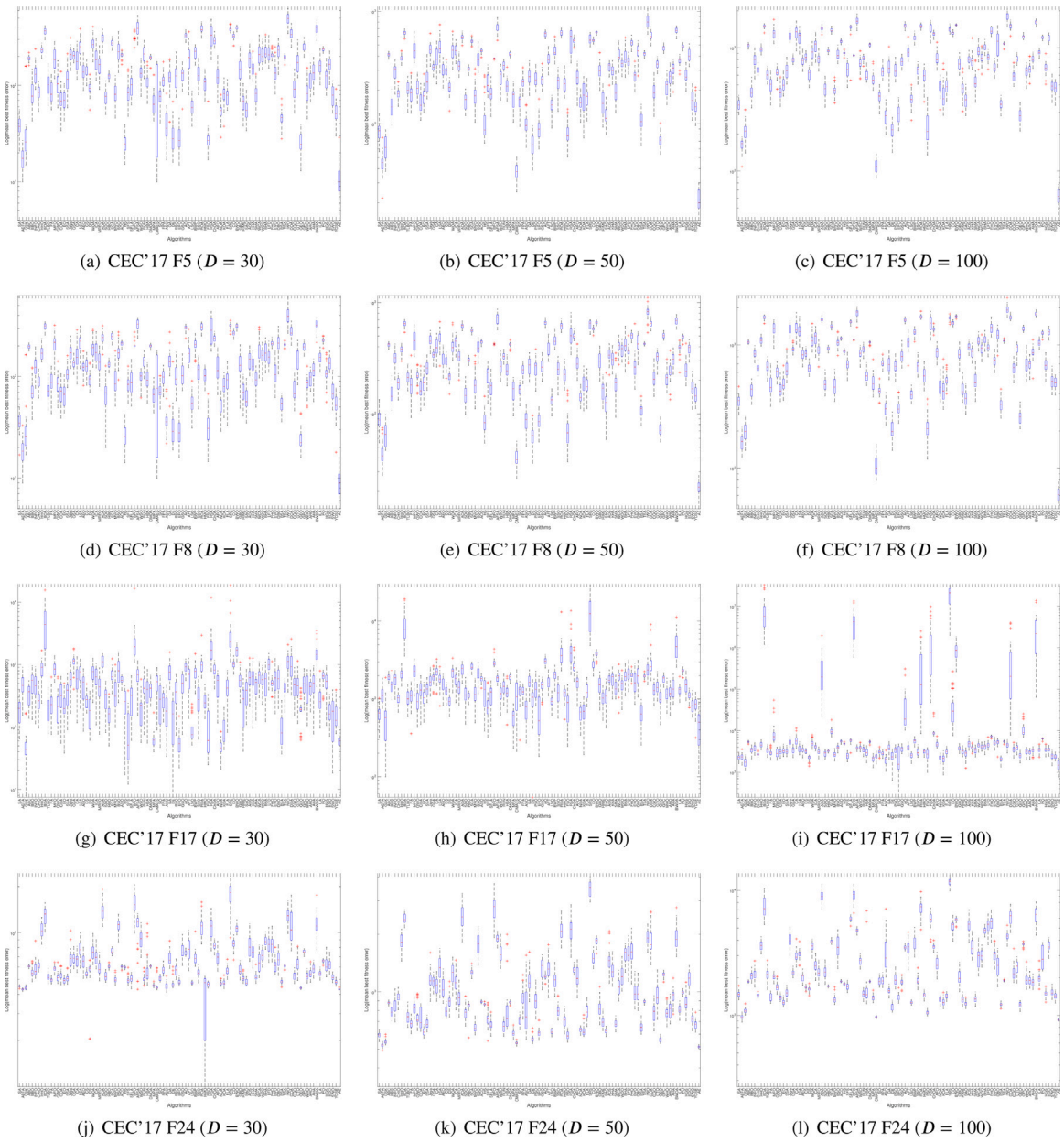
(j) CEC'17 F24 ($D = 30$) (k) CEC'17 F24 ($D = 50$) (l) CEC'17 F24 ($D = 100$)

**Fig. 11.** The convergence curves of AE and the 100 optimization algorithms on the four representative functions.

algorithms are equal, while the alternative hypothesis suggests the presence of differences in the medians among two or more algorithms. Due to the extensive volume of data obtained from the experiments, the differences in results between algorithms cannot be presented in the form of a convergence accuracy table. Therefore, the Friedman test is utilized to calculate the rankings of the algorithms across all problems, and these rankings are documented in Table 8. Moreover, Fig. 13 visualizes the results of the top 6 and the last 6 algorithms to present them better.

From Table 8 and Fig. 13, it can be observed that although AE is ranked at 11.10 at $D = 30$, there is no significant advantage in this scenario. However, as the problem dimensionality increases, AE secures the first position with statistical rankings of 7.30 and 8.27 at $D = 50$ and $D = 100$, respectively. Furthermore, the last column of the table presents the average rankings of algorithms across the three dimensions. The average rankings clearly show the efficiency of the top 6 algorithms are for solving these problems. Indeed, AE achieves the highest overall ranking, indicating its effectiveness in solving problems with different topologies and different dimensions. Thus, this further confirms the efficacy of AE's, even if it has only one core operator for approximating the optimal solution.

The Wilcoxon rank sum test is used to compare whether there is a significant difference between AE and competitors under the significance level of 0.05. The symbols "+/ ≈ /−" are derived from the Wilcoxon rank sum test and indicate the number of test problems where the competing algorithms perform better, similarly, or worse compared to AE under the significance level of 0.05. Table 9 shows the results under three problem dimensions ($D = 30/50/100$). It can be observed that when $D = 30$, GSK has the best performance among competitors, defeating AE in 16 problems. Additionally, NOA, EFO, SWO, LSO, KOA, and MPA can outperform AE in no fewer than 10 problems. However, when $D = 50$, NOA, GSK, SWO, and MPA, with the highest number of victories against AE, only defeated AE on a maximum of 7 problems. In addition, when $D = 100$, the best performing KOA among competitors is only better than AE in 7 problems. As for the other algorithms, they only manage to outperform or be similar to AE in a few problems



(a) CEC'17 F5 ($D = 30$) (b) CEC'17 F5 ($D = 50$) (c) CEC'17 F5 ($D = 100$)

(d) CEC'17 F8 ($D = 30$) (e) CEC'17 F8 ($D = 50$) (f) CEC'17 F8 ($D = 100$)

(g) CEC'17 F17 ($D = 30$) (h) CEC'17 F17 ($D = 50$) (i) CEC'17 F17 ($D = 100$)

(j) CEC'17 F24 ($D = 30$) (k) CEC'17 F24 ($D = 50$) (l) CEC'17 F24 ($D = 100$)

**Fig. 12.** The box plots of AE and the 100 optimization algorithms on the four representative functions.

and fall behind AE in most cases. Therefore, AE has been sufficiently demonstrated to be competitive.

### 3.8. Compare AE with 6 advanced adaptive algorithms

To further verify the performance of AE, six advanced adaptive algorithms were also selected. The six algorithms are the improved multi-operator differential evolution algorithm (IMODE) (Sallam et al., 2020), the gaining-sharing knowledge based algorithm with adaptive parameters hybrid with IMODE algorithm (APGSK-IMODE) (Mohamed et al., 2021), the adaptive differential evolution with optional external archive (JADE) (Zhang and Sanderson, 2009), the adaptive guided differential evolution (AGDE) (Mohamed and Mohamed, 2019), the differential evolutionary algorithm with an evolutionary state estimation method and a two-level selection mechanism (DEET) (Li and Li, 2020), and the differential mutation and novel social learning particle swarm optimization algorithm (DSPSO) (Zhang et al., 2019). The experiment was conducted on the CEC'17 test suite, with 100-dimensional testing and 30 independent executions. For detailed information on the test set, please refer to subsection 3.7. The termination criterion for algorithm execution is set to $MaxFEs = 10,000 \times D$. The parameter settings of these algorithms are derived from the papers that proposed them. Table 10 presents the mean, standard deviation, and statistical results obtained from the experimental tests.

As can be seen from Table 10, it all shows competition regardless of whether AE solves unimodal, multimodal, hybrid, or composition problems. AE outperforms the compared algorithms in a minimum of 11 problems, exhibits comparable performance in up to 10 problems, and is surpassed by the compared algorithms in a maximum of 9 problems. Although AE did not perform the best on problems such as CEC'17 F3, CEC'17 F6, and CEC'17 F9, it still achieved suboptimal results with smaller disadvantages. However, AE could not find a global optimal on CEC'17 F1 and CEC'17 F6, which is regrettable. By comparing the performance of AE with advanced algorithms for adaptive parameter control, the experiment hopes to provide meaningful insights to the reader.

**Table 8**
The statistical rankings obtained by the Friedman test.

| Placement | Algorithm | $D = 30$ | $D = 50$ | $D = 100$ | Mean rank | Placement | Algorithm | $D = 30$ | $D = 50$ | $D = 100$ | Mean rank |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | AE | 11.10 | **7.30** | **8.27** | **8.89** | 52 | HS | 36.87 | 43.07 | 55.43 | 45.12 |
| 2 | NOA | **8.30** | 10.57 | 12.33 | 10.40 | 53 | GSA | 53.73 | 48.17 | 40.30 | 47.40 |
| 3 | GSK | 9.50 | 15.17 | 16.23 | 13.63 | 54 | HGS | 51.13 | 47.50 | 43.87 | 47.50 |
| 4 | EFO | 8.40 | 18.93 | 19.07 | 15.47 | 55 | WSO | 43.77 | 49.40 | 49.83 | 47.67 |
| 5 | SWO | 13.18 | 16.33 | 16.93 | 15.48 | 56 | SFLA | 29.07 | 28.70 | 88.07 | 48.61 |
| 6 | BSA | 16.27 | 16.57 | 14.07 | 15.64 | 57 | GTO | 50.83 | 50.83 | 47.43 | 49.70 |
| 7 | LSO | 13.63 | 16.30 | 17.60 | 15.84 | 58 | ICA | 52.97 | 50.97 | 46.67 | 50.20 |
| 8 | KOA | 15.97 | 19.87 | 17.27 | 17.70 | 59 | RUN | 51.20 | 50.37 | 49.90 | 50.49 |
| 9 | MPA | 11.80 | 16.87 | 25.17 | 17.95 | 60 | DMOA | 49.47 | 51.70 | 59.43 | 53.53 |
| 10 | FSO | 20.83 | 19.30 | 18.27 | 19.47 | 61 | WCA | 56.37 | 55.27 | 49.10 | 53.58 |
| 11 | SFS | 16.57 | 22.03 | 24.20 | 20.93 | 62 | ABC | 49.73 | 51.43 | 59.97 | 53.71 |
| 12 | DE | 18.90 | 22.43 | 23.13 | 21.49 | 63 | GWO | 56.17 | 55.20 | 54.33 | 55.23 |
| 13 | YDSE | 21.10 | 22.80 | 29.87 | 24.59 | 64 | LSA | 59.63 | 55.13 | 51.87 | 55.54 |
| 14 | GEO | 26.18 | 25.00 | 22.93 | 24.70 | 65 | TSA | 52.97 | 55.03 | 62.87 | 56.96 |
| 15 | ASO | 25.33 | 25.10 | 26.10 | 25.51 | 66 | EDO | 45.63 | 58.40 | 71.73 | 58.59 |
| 16 | AEFA | 32.17 | 25.00 | 19.73 | 25.63 | 67 | AFO | 58.20 | 59.97 | 57.93 | 58.70 |
| 17 | SAO | 28.33 | 25.67 | 25.43 | 26.48 | 68 | ACO | 52.77 | 56.53 | 67.00 | 58.77 |
| 18 | EO | 28.10 | 31.03 | 27.73 | 28.95 | 69 | ILA | 56.37 | 61.10 | 63.37 | 60.28 |
| 19 | COA | 29.90 | 31.93 | 26.13 | 29.32 | 70 | BSO | 66.03 | 62.33 | 57.47 | 61.94 |
| 20 | SDO | 27.60 | 30.03 | 32.37 | 30.00 | 71 | KHA | 66.87 | 65.37 | 63.00 | 65.08 |
| 21 | DS | 25.07 | 31.27 | 34.57 | 30.30 | 72 | IWO | 72.20 | 67.90 | 59.83 | 66.64 |
| 22 | FA | 32.90 | 30.47 | 28.67 | 30.68 | 73 | WDO | 71.60 | 68.20 | 64.47 | 68.09 |
| 23 | RIME | 35.80 | 32.63 | 33.63 | 34.02 | 74 | AOS | 70.87 | 68.93 | 67.43 | 69.08 |
| 24 | SO | 36.17 | 35.13 | 32.23 | 34.51 | 75 | EVO | 70.60 | 70.20 | 68.53 | 69.78 |
| 25 | CGO | 35.07 | 36.07 | 35.97 | 35.70 | 76 | LFD | 76.03 | 71.50 | 63.63 | 70.39 |
| 26 | ARO | 36.27 | 36.13 | 35.13 | 35.84 | 77 | DBO | 70.40 | 71.77 | 69.07 | 70.41 |
| 27 | CS | 31.53 | 35.90 | 42.83 | 36.75 | 78 | BHA | 72.97 | 73.33 | 71.97 | 72.76 |
| 28 | MVO | 43.23 | 37.50 | 31.50 | 37.41 | 79 | HHO | 77.47 | 73.23 | 69.17 | 73.29 |
| 29 | GBO | 38.37 | 40.37 | 34.17 | 37.64 | 80 | CA | 73.27 | 73.07 | 79.33 | 75.22 |
| 30 | INFO | 37.30 | 40.03 | 36.47 | 37.93 | 81 | PO | 74.70 | 78.03 | 75.30 | 76.01 |
| 31 | SMA | 44.60 | 37.90 | 31.63 | 38.04 | 82 | BWP | 75.27 | 75.37 | 79.10 | 76.58 |
| 32 | HMO | 38.87 | 37.87 | 38.30 | 38.35 | 83 | Jaya | 74.10 | 76.43 | 80.63 | 77.05 |
| 33 | AEO | 40.33 | 40.13 | 35.13 | 38.53 | 84 | MFO | 77.93 | 78.33 | 77.90 | 78.05 |
| 34 | HBO | 37.63 | 36.67 | 41.37 | 38.56 | 85 | WOA | 82.97 | 78.23 | 73.63 | 78.28 |
| 35 | SA | 36.87 | 37.30 | 42.47 | 38.88 | 86 | AFSA | 81.17 | 79.07 | 77.03 | 79.09 |
| 36 | TLBO | 35.57 | 38.53 | 43.20 | 39.10 | 87 | GPC | 76.30 | 79.80 | 82.30 | 79.47 |
| 37 | AHA | 41.27 | 41.27 | 35.67 | 39.40 | 88 | FHO | 82.90 | 84.80 | 84.87 | 84.19 |
| 38 | FPA | 36.87 | 39.57 | 42.37 | 39.60 | 89 | SCA | 84.93 | 86.43 | 85.97 | 85.78 |
| 39 | CHIO | 39.83 | 37.07 | 42.23 | 39.71 | 90 | ChOA | 87.00 | 87.23 | 84.50 | 86.24 |
| 40 | PSO | 40.93 | 37.90 | 40.80 | 39.88 | 91 | CO | 87.33 | 88.47 | 88.33 | 88.04 |
| 41 | WHO | 41.03 | 39.87 | 39.57 | 40.16 | 92 | AO | 87.37 | 90.03 | 89.43 | 88.94 |
| 42 | HBA | 47.27 | 41.53 | 32.17 | 40.32 | 93 | AOA | 88.50 | 90.10 | 92.00 | 90.20 |
| 43 | MRFO | 43.30 | 42.27 | 35.73 | 40.43 | 94 | RSA | 94.77 | 93.10 | 90.13 | 92.67 |
| 44 | BBO | 44.50 | 39.87 | 38.80 | 41.06 | 95 | BFO | 96.97 | 92.57 | 88.63 | 92.72 |
| 45 | FLA | 44.73 | 39.70 | 38.87 | 41.10 | 96 | DOA | 91.77 | 94.53 | 95.50 | 93.93 |
| 46 | AFT | 43.70 | 41.93 | 39.73 | 41.79 | 97 | BWO | 93.73 | 94.37 | 93.97 | 94.02 |
| 47 | CMAES | 52.93 | 40.77 | 32.57 | 42.09 | 98 | ROA | 95.60 | 94.90 | 93.37 | 94.62 |
| 48 | SSA | 47.07 | 42.73 | 38.13 | 42.64 | 99 | BOA | 95.03 | 97.27 | 96.60 | 96.30 |
| 49 | SSO | 42.67 | 43.23 | 42.20 | 42.70 | 100 | SRS | 97.50 | 97.63 | 94.90 | 96.68 |
| 50 | GSO | 46.97 | 47.23 | 38.30 | 44.17 | 101 | BWOA | 97.20 | 97.57 | 98.47 | 97.75 |
| 51 | GA | 51.00 | 45.07 | 38.20 | 44.76 | – | – | – | – | – | – |

### 3.9. Complexity analysis

The big $O$ notation is usually used to evaluate the time complexity of algorithms (Chivers et al., 2015). Therefore, this section also uses this method to evaluate the computational complexity of AE. First, the cost of initializing and evaluating the candidate solutions is $O(N \times D)$. Second, it is necessary to sort the objective function values, and the time cost of the Quicksort algorithm used in this step is $O(N\log(N))$. Then, the maximum cost required for executing the operator in the for loop is $O(D \times D)$. The reason for this expensive overhead is caused by the diagonal operation on matrix $\mathbf{A}$. Therefore, the overall time complexity of AE is $O(N \times D) + O(N\log(N)) + O(D \times D)$.

Furthermore, another complexity computation method is considered. Four complexity-related parameters $T0$, $T1$, $T2$, and $\overline{T2}$ are considered according to the CEC' 17 problem definition and evaluation criteria. $T0$ represents the running time of the test command. $T1$ represents the time for 200,000 evaluations of the F18 function of a certain dimension $D$. $T2$ is the complete computation time for 200,000 evaluations of the same function. This step is repeated five times to obtain $\overline{T2} = \text{mean}(T2)$. Therefore, the complexity of the AE algorithm $(\overline{T2} - T1)/T0$ is calculated as shown in Table 11.

## 4. Application I : Multiple sequence alignment

Multiple sequence alignment (MSA) is a fundamental cornerstone of bioinformatics and is used to unravel the function, evolution, and structure of biological sequences (Zhang et al., 2022; Sun et al., 2012). With the prevalence of COVID-19, it also plays a crucial role in virus prevention and vaccine development. Additionally, MSA is considered one of the most challenging tasks in bioinformatics, as it is an NP-complete problem (Lee et al., 2008). This implies that the computational time required to solve this problem increases exponentially with the input size. To accomplish the multiple sequence alignment task, characters in the same column of three or more sequences need to be aligned (Sayood and Otu, 2023).

For MSA, the hidden Markov model (HMM) is a probabilistic model considered one of the most popular techniques for solving MSA (Karplus et al., 1998). However, training HMM is highly challenging, and no known deterministic algorithm guarantees the generation

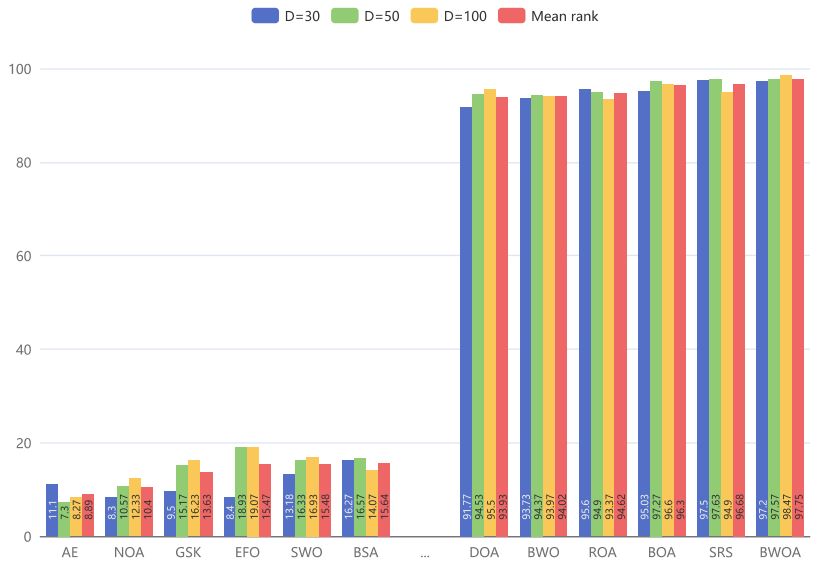



**Fig. 13.** The Friedman ranking for the top 6 and the last 6 algorithms.

of the optimal HMM within a reasonable computation time (Sun et al., 2012). In contrast, intelligent optimization algorithms offer advantages such as lower complexity, less stringent restrictions on the number and length of sequences to be aligned, and the ability to avoid local optima. Therefore, these algorithms can effectively train HMM and solve the MSA (Öztürk and Aslan, 2016). These methods aim to maximize the objective function value to obtain an optimal alignment result. Based on the above, the proposed AE algorithm is applied to the MSA problem to demonstrate its practical value.

### 4.1. Experimental setup

To effectively evaluate the quality of HMM trained by AE, the top 5 performers from numerical optimization experiments are selected as competitors to AE, namely NOA, GSK, EFO, SWO, and BSA. To assess the quality of the alignment results obtained by the algorithms, the sum-of-pairs (SOP) scoring function is used as the maximum objective function. The mathematical model of the SOP function is given by Eq. (18):

$$SOP = \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} Dist\left(l_i, l_j\right) \tag{18}$$

where $l_i$ and $l_j$ are aligned sequences, and $Dist$ is the distance metric. Note that the SOP function is a maximizing objective function for MSA problems.

The eight gene sequences utilized in the experiment were obtained from the international benchmark alignment database known as BAliBASE (http://www.drive5.com/bench/). The relevant information is presented in Table 12, which includes the sequence name, the number of sub-sequences within the sequence, the range of sub-sequence lengths, and the dimension of the solution determined based on the sequence. Since the top 6 algorithms optimize the parameters in the probabilistic model HMM, the valid search space for the problem is $[0, 1]$. Moreover, the constraint methods for all algorithms are consistent with their definitions in the relevant literature. All algorithms are independently executed 30 times for each sequence. In each run, the algorithms terminate when $MaxFEs = 4,000$ is satisfied.

### 4.2. Experimental results and analysis

The results of aligning the sequences using the corresponding algorithms are listed in Table 13. As the length and number of sub-sequences increase, aligning sequences becomes more challenging for algorithms. From Table 13, it can be seen that when aligning the four sequences, 2myr_ref1, 3pmg_ref1, 3grs_ref1, and 4enl_ref1, the AE algorithm demonstrates certain advantages. However, it performs slightly worse than the NOA algorithm when aligning the other two sequences. Additionally, the AE algorithm exhibits a shorter execution time when aligning 3pmg_ref1. Based on the standard deviation results, both NOA and AE show more stable performance than other algorithms. Moreover, BSA has certain advantages in reducing time overhead.

To better illustrate the average score of the best results obtained by the algorithms during HMM training, convergence curves achieved by the algorithms are depicted in Fig. 14. It is evident that AE exhibits the fastest convergence rate among the eight gene sequences, as it attains relatively good results in the early iterations. On the other hand, NOA, which performs slightly worse, requires more time to achieve similar outcomes. This characteristic of AE is highly advantageous when addressing real-world problems, especially for real-time decision-making and dynamically changing scenarios such as logistics scheduling, traffic planning, and stock trading.

To verify whether AE can effectively align gene sequences, Fig. 15 shows partial visualization results by AE aligning 1ped_ref1, 2myr_ref1, 3pmg_ref1, and 4nel_ref1 sequences. On the sequence identity maps, the accumulation of bases in the same column reflects the consistency of bases in that column. The size of base graphic characters is proportional to the frequency of the base appearing in that column. The maximum value on the y-axis of the sequence flag is $\log_2 4 = 2$ bits, where 4 is the number of DNA bases (Zhang et al., 2023). In the consensus, the symbol "!" indicates that the same base occupies a significant proportion at a certain position, indicating that all sequences at that position have been well aligned. The symbol "$*$" indicates that the base types at a certain position are relatively consistent, while a blank indicates that the base types at that position have very low similarity. Moreover, the symbol "." represents insertion during the alignment sequence process, thereby achieving aligning operations.

From Fig. 15, it can also be observed that many positions in the sequence have only one type of nucleotide, indicating that these positions are well aligned. This also confirms that AE can solve the

**Table 9**
The statistical results obtained by the Wilcoxon rank sum test.

| vs. AE | $D = 30$ | $D = 50$ | $D = 100$ | vs. AE | $D = 30$ | $D = 50$ | $D = 100$ |
|---|---|---|---|---|---|---|---|
| NOA | 12/6/12 | 7/6/17 | 5/3/22 | HS | 0/3/27 | 1/2/27 | 1/1/28 |
| GSK | 16/5/9 | 7/6/17 | 3/12/15 | GSA | 0/5/25 | 2/1/27 | 3/3/24 |
| EFO | 14/4/12 | 3/5/22 | 5/6/19 | HGS | 0/2/28 | 0/1/29 | 0/0/30 |
| SWO | 13/4/13 | 7/5/18 | 5/4/21 | WSO | 1/3/26 | 0/3/27 | 0/3/27 |
| BSA | 8/5/17 | 4/5/21 | 6/2/22 | SFLA | 4/4/22 | 3/3/24 | 0/0/30 |
| LSO | 11/3/16 | 6/5/19 | 6/1/23 | GTO | 1/5/24 | 0/2/28 | 0/4/26 |
| KOA | 10/4/16 | 3/7/20 | 7/3/20 | ICA | 1/3/26 | 1/4/25 | 1/7/22 |
| MPA | 11/6/13 | 7/4/19 | 2/4/24 | RUN | 0/1/29 | 0/0/30 | 0/0/30 |
| FSO | 3/3/24 | 2/4/24 | 3/4/23 | DMOA | 2/4/24 | 1/3/26 | 2/1/27 |
| SFS | 8/8/14 | 5/3/22 | 2/5/23 | WCA | 0/3/27 | 0/2/28 | 0/5/25 |
| DE | 4/10/16 | 2/5/23 | 2/4/24 | ABC | 1/5/24 | 1/2/27 | 2/1/27 |
| YDSE | 7/4/19 | 4/0/26 | 1/1/28 | GWO | 0/1/29 | 0/0/30 | 0/0/30 |
| GEO | 1/4/25 | 2/1/27 | 1/5/24 | LSA | 0/3/27 | 0/1/29 | 0/4/26 |
| ASO | 1/4/25 | 2/4/24 | 4/3/23 | TSA | 2/3/25 | 1/1/28 | 3/0/27 |
| AEFA | 5/2/23 | 3/3/24 | 6/3/21 | EDO | 5/2/23 | 4/0/26 | 1/0/29 |
| SAO | 1/6/23 | 1/2/27 | 2/4/24 | AFO | 0/1/29 | 0/0/30 | 0/0/30 |
| EO | 1/4/25 | 0/3/27 | 1/6/23 | ACO | 1/3/26 | 0/1/29 | 0/3/27 |
| COA | 4/5/21 | 2/2/26 | 3/2/25 | ILA | 2/0/28 | 0/0/30 | 0/0/30 |
| SDO | 3/4/23 | 1/2/27 | 2/3/25 | BSO | 1/1/28 | 1/0/29 | 0/2/28 |
| DS | 6/3/21 | 3/2/25 | 3/1/26 | KHA | 0/0/30 | 0/0/30 | 0/0/30 |
| FA | 1/1/28 | 0/4/26 | 0/5/25 | IWO | 0/1/29 | 0/1/29 | 1/2/27 |
| RIME | 0/2/28 | 0/1/29 | 0/1/29 | WDO | 0/1/29 | 0/0/30 | 0/0/30 |
| SO | 2/2/26 | 2/0/28 | 2/1/27 | AOS | 0/0/30 | 0/0/30 | 0/0/30 |
| CGO | 3/6/21 | 0/4/26 | 0/5/25 | EVO | 0/0/30 | 0/0/30 | 0/0/30 |
| ARO | 0/5/25 | 1/2/27 | 2/3/25 | LFD | 0/2/28 | 0/1/29 | 2/0/28 |
| CS | 8/1/21 | 3/2/25 | 0/3/27 | DBO | 0/0/30 | 0/0/30 | 0/0/30 |
| MVO | 1/0/29 | 0/0/30 | 0/2/28 | BHA | 0/0/30 | 0/0/30 | 0/0/30 |
| GBO | 2/5/23 | 0/4/26 | 2/4/24 | HHO | 0/0/30 | 0/0/30 | 0/0/30 |
| INFO | 8/1/21 | 1/4/25 | 2/5/23 | CA | 1/0/29 | 1/1/28 | 0/1/29 |
| SMA | 0/1/29 | 0/0/30 | 0/2/28 | PO | 0/0/30 | 0/0/30 | 0/1/29 |
| HMO | 1/4/25 | 1/2/27 | 0/3/27 | BWP | 1/0/29 | 1/0/29 | 0/1/29 |
| AEO | 0/6/24 | 1/3/26 | 2/4/24 | Jaya | 0/0/30 | 0/0/30 | 0/0/30 |
| HBO | 2/5/23 | 3/2/25 | 6/2/22 | MFO | 0/0/30 | 0/0/30 | 0/0/30 |
| SA | 1/2/27 | 2/1/27 | 1/2/27 | WOA | 0/0/30 | 0/0/30 | 0/0/30 |
| TLBO | 0/5/25 | 0/3/27 | 1/2/27 | AFSA | 0/0/30 | 0/0/30 | 0/0/30 |
| AHA | 1/4/25 | 1/2/27 | 3/3/24 | GPC | 0/0/30 | 0/0/30 | 0/0/30 |
| FPA | 5/3/22 | 3/1/26 | 1/2/27 | FHO | 0/0/30 | 0/0/30 | 0/0/30 |
| CHIO | 1/3/26 | 0/3/27 | 0/0/30 | SCA | 0/0/30 | 0/0/30 | 0/0/30 |
| PSO | 1/3/26 | 0/2/28 | 0/1/29 | ChOA | 0/0/30 | 0/0/30 | 0/0/30 |
| WHO | 0/3/27 | 0/2/28 | 0/4/26 | CO | 0/0/30 | 0/0/30 | 0/0/30 |
| HBA | 0/3/27 | 0/2/28 | 0/4/26 | AO | 0/0/30 | 0/0/30 | 0/0/30 |
| MRFO | 2/2/26 | 1/4/25 | 2/6/22 | AOA | 0/0/30 | 0/0/30 | 0/0/30 |
| BBO | 0/2/28 | 1/1/28 | 2/1/27 | RSA | 0/0/30 | 0/0/30 | 0/0/30 |
| FLA | 0/2/28 | 0/1/29 | 1/0/29 | BFO | 0/0/30 | 0/0/30 | 0/0/30 |
| AFT | 6/5/19 | 3/6/21 | 2/6/22 | DOA | 0/0/30 | 0/0/30 | 0/0/30 |
| CMAES | 5/1/24 | 5/1/24 | 7/1/22 | BWO | 0/0/30 | 0/0/30 | 0/0/30 |
| SSA | 0/2/28 | 0/2/28 | 0/2/28 | ROA | 0/0/30 | 0/0/30 | 0/0/30 |
| SSO | 3/0/27 | 1/1/28 | 0/1/29 | BOA | 0/0/30 | 0/0/30 | 0/0/30 |
| GSO | 1/3/26 | 1/1/28 | 2/3/25 | SRS | 0/0/30 | 0/0/30 | 0/0/30 |
| GA | 2/3/25 | 2/4/24 | 3/4/23 | BWOA | 0/0/30 | 0/0/30 | 0/0/30 |

MSA problem by training high-quality HMM. However, this is not an easy task because aligning one column of the aligned sequences can potentially misalign another.

MSA is a challenging interdisciplinary frontier problem. Through this experiment, the practical value of AE is demonstrated, as it can effectively address the MSA problem with a competitive advantage. Furthermore, as shown in Fig. 14, AE has an advantage in solving problems requiring faster response times, as it efficiently converges early in the iterations and finds meaningful solutions. However, according to Table 13, AE does not exhibit a significant advantage in terms of runtime. Nevertheless, it is worthwhile to obtain better results through additional computations.

## 5. Application II : Engineering design problems

Algorithms designed for specific problems often cannot directly adapt to other problems, which may affect users' progress in solving application problems (Yüksel et al., 2023). Therefore, it is essential to test the adaptability of algorithms through different problems in practice. Engineering design optimization problems are usually highly nonlinear, involving diverse design variables and complex design constraints (Chou and Ngo, 2017; Abbaszadeh Shahri et al., 2021). Therefore, using engineering design problems to evaluate algorithm performance is indispensable. This section considers three constrained engineering design problems: pressure vessel design, tension/compression spring design, and welded beam design. These problems are inequality-constrained optimization problems, and hence it is necessary for such problems to use penalty functions to constrain the variables. Penalty functions commonly used for these problems are the static penalty, the dynamic penalty, the annealing penalty, the adaptive penalty, the co-evolutionary penalty, and the death penalty (Coello, 2002). This experiment uses the death penalty, which is a simple and efficient method to deal with constraints. Its objective function value is assigned a sufficiently large penalty value when a solution violates a constraint. Therefore, optimization algorithms easily recognize and eliminate it as an infeasible solution.

The top 6 algorithms of the numerical optimization experiments, including AE, NOA, GSK, EFO, SWO, and BSA, were involved in these three engineering problems. All these algorithms use the experimental setup of $N = 40$, $MaxFEs = 20{,}000$, and $run = 30$. Furthermore, please refer to Table 7 for their other parameter settings.

**Table 10**
Comparison results between AE and advanced adaptive algorithms on the CEC'17 test suite.

| No. | | APGSK-IMODE | JADE | IMODE | AGDE | DEET | DSPSO | AE |
|---|---|---|---|---|---|---|---|---|
| CEC'17 F1 | Mean | 4.06E+03 | **0.00E+00** | 1.21E+08 | 5.71E+02 | **0.00E+00** | 6.74E+03 | 6.97E+03 |
| | Std | 2.36E+03 | **1.49E−19** | 3.63E+08 | 1.07E+03 | **4.22E−16** | 6.81E+03 | 5.23E+03 |
| CEC'17 F2 | Mean | 4.46E+76 | 1.50E+94 | 7.00E+145 | 2.98E+13 | 1.05E+32 | 5.42E+71 | **1.78E+00** |
| | Std | 8.09E+76 | 8.22E+94 | 1.75E+146 | 1.14E+14 | 5.14E+32 | 2.97E+72 | **6.87E+00** |
| CEC'17 F3 | Mean | 3.55E+05 | 3.33E+05 | **7.00E−05** | 2.41E+02 | 1.48E+05 | 5.85E+03 | 3.19E+00 |
| | Std | 1.03E+04 | 9.48E+04 | **1.22E−06** | 2.46E+02 | 1.55E+05 | 2.71E+03 | 2.83E+00 |
| CEC'17 F4 | Mean | 2.57E+02 | **5.42E+01** | 1.16E+02 | 2.01E+02 | 1.03E+02 | 4.98E+02 | 2.17E+02 |
| | Std | 3.06E+01 | **4.97E+01** | 8.94E+01 | 2.85E+01 | 5.91E+01 | 3.70E+01 | 1.78E+01 |
| CEC'17 F5 | Mean | 9.03E+02 | 4.18E+02 | 9.90E+02 | 7.25E+02 | 1.28E+02 | 8.82E+01 | **6.28E+01** |
| | Std | 8.30E+01 | 2.17E+01 | 7.30E+01 | 2.41E+01 | 2.14E+01 | 1.18E+01 | **9.45E+00** |
| CEC'17 F6 | Mean | 1.77E+01 | 1.94E−04 | 6.18E+01 | **0.00E+00** | 2.01E−02 | 1.86E−02 | 3.22E−03 |
| | Std | 3.41E+00 | 7.44E−04 | 2.24E+00 | **0.00E+00** | 3.71E−02 | 2.49E−02 | 4.28E−03 |
| CEC'17 F7 | Mean | 7.98E+02 | 5.10E+02 | 2.27E+03 | 8.47E+02 | 2.47E+02 | 1.58E+02 | **1.51E+02** |
| | Std | 1.86E+01 | 2.03E+01 | 4.19E+02 | 2.44E+01 | 1.92E+01 | 9.08E+00 | **7.29E+00** |
| CEC'17 F8 | Mean | 6.74E+02 | 4.11E+02 | 1.04E+03 | 7.08E+02 | 1.39E+02 | 8.80E+01 | **6.12E+01** |
| | Std | 2.11E+01 | 2.47E+01 | 1.01E+02 | 2.77E+01 | 2.25E+01 | 1.07E+01 | **8.32E+00** |
| CEC'17 F9 | Mean | 1.02E+02 | 1.83E+00 | 3.09E+04 | **8.37E−01** | 7.63E+01 | 1.15E+01 | 1.68E+00 |
| | Std | 1.70E+02 | 1.29E+00 | 4.32E+03 | **1.17E+00** | 6.54E+01 | 6.62E+00 | 3.17E+00 |
| CEC'17 F10 | Mean | 2.02E+04 | 2.13E+04 | 1.12E+04 | 2.40E+04 | 1.06E+04 | **7.90E+03** | 1.15E+04 |
| | Std | 3.84E+02 | 4.39E+02 | 1.11E+03 | 5.52E+02 | 1.49E+03 | **9.22E+02** | 1.17E+03 |
| CEC'17 F11 | Mean | 1.12E+03 | 1.81E+04 | 1.39E+03 | **1.41E+02** | 2.31E+03 | 1.10E+03 | 8.46E+02 |
| | Std | 2.14E+02 | 7.18E+03 | 3.50E+02 | **3.93E+01** | 1.53E+03 | 2.11E+02 | 1.00E+02 |
| CEC'17 F12 | Mean | 9.56E+05 | 2.27E+04 | **4.23E+03** | 2.70E+05 | 2.16E+04 | 3.63E+06 | 3.52E+05 |
| | Std | 5.05E+05 | 9.34E+03 | **7.84E+02** | 1.46E+05 | 1.20E+04 | 6.64E+06 | 1.14E+05 |
| CEC'17 F13 | Mean | 2.47E+03 | 2.14E+03 | **6.25E+02** | 3.39E+03 | 3.85E+03 | 1.94E+03 | 7.18E+03 |
| | Std | 9.86E+02 | 1.44E+03 | **9.27E+01** | 2.70E+03 | 4.03E+03 | 1.62E+03 | 3.82E+03 |
| CEC'17 F14 | Mean | 2.16E+04 | 4.44E+05 | 4.40E+02 | 6.22E+03 | **3.37E+02** | 5.74E+04 | 8.58E+03 |
| | Std | 7.51E+03 | 1.39E+06 | 8.48E+01 | 7.48E+03 | **1.07E+02** | 1.89E+04 | 7.68E+03 |
| CEC'17 F15 | Mean | 3.58E+02 | 2.79E+03 | **2.58E+02** | 2.81E+03 | 1.10E+03 | 1.45E+03 | 4.60E+03 |
| | Std | 1.47E+02 | 1.39E+04 | **6.79E+01** | 3.18E+03 | 1.14E+03 | 1.73E+03 | 6.29E+03 |
| CEC'17 F16 | Mean | 5.93E+03 | 3.83E+03 | 4.51E+03 | 3.62E+03 | 2.64E+03 | 1.62E+03 | **1.61E+03** |
| | Std | 3.46E+02 | 2.86E+02 | 8.09E+02 | 2.53E+02 | 3.89E+02 | 6.61E+02 | **4.58E+02** |
| CEC'17 F17 | Mean | 3.02E+03 | 2.66E+03 | 3.79E+03 | 2.35E+03 | 1.66E+03 | 1.52E+03 | **1.51E+03** |
| | Std | 1.36E+02 | 2.14E+02 | 6.28E+02 | 2.53E+02 | 2.88E+02 | 3.64E+02 | **4.20E+02** |
| CEC'17 F18 | Mean | 1.73E+05 | 1.94E+05 | **7.19E+02** | 1.96E+04 | 2.69E+03 | 2.21E+05 | 2.49E+04 |
| | Std | 2.44E+04 | 1.06E+06 | **1.61E+03** | 9.91E+03 | 2.30E+03 | 7.57E+04 | 1.93E+04 |
| CEC'17 F19 | Mean | 9.11E+02 | **4.31E+02** | 6.11E+02 | 2.26E+03 | 1.65E+03 | 2.52E+03 | 7.39E+03 |
| | Std | 3.87E+02 | **1.43E+03** | 3.33E+02 | 2.97E+03 | 1.94E+03 | 2.38E+03 | 8.80E+03 |
| CEC'17 F20 | Mean | 3.26E+03 | 2.81E+03 | 3.08E+03 | 2.32E+03 | 1.69E+03 | **9.84E+02** | 1.61E+03 |
| | Std | 2.44E+02 | 2.61E+02 | 4.94E+02 | 2.98E+02 | 3.04E+02 | **3.55E+02** | 4.58E+02 |
| CEC'17 F21 | Mean | 8.94E+02 | 6.52E+02 | 1.46E+03 | 9.49E+02 | 3.49E+02 | 3.19E+02 | **2.83E+02** |
| | Std | 2.56E+01 | 7.28E+01 | 1.21E+02 | 2.80E+01 | 2.50E+01 | 1.40E+01 | **7.28E+00** |
| CEC'17 F22 | Mean | 2.13E+04 | 2.15E+04 | 1.36E+04 | 2.52E+04 | 1.17E+04 | **2.15E+03** | 1.16E+04 |
| | Std | 6.35E+02 | 4.00E+03 | 9.08E+02 | 5.73E+02 | 1.10E+03 | **4.18E+03** | 1.62E+03 |
| CEC'17 F23 | Mean | 1.11E+03 | 9.37E+02 | 1.99E+03 | 1.03E+03 | 6.71E+02 | 6.93E+02 | **6.09E+02** |
| | Std | 3.72E+01 | 2.53E+01 | 1.21E+02 | 1.65E+01 | 2.70E+01 | 2.43E+01 | **1.84E+01** |
| CEC'17 F24 | Mean | 1.41E+03 | 1.21E+03 | 2.85E+03 | 1.54E+03 | 1.02E+03 | 9.66E+02 | **9.21E+02** |
| | Std | 3.45E+01 | 7.53E+01 | 3.79E+02 | 2.60E+01 | 2.17E+01 | 2.35E+01 | **1.29E+01** |
| CEC'17 F25 | Mean | 8.59E+02 | 7.56E+02 | **7.31E+02** | 7.51E+02 | 7.44E+02 | 1.07E+03 | 7.73E+02 |
| | Std | 3.27E+01 | 4.63E+01 | **4.63E+01** | 4.87E+01 | 6.48E+01 | 4.96E+01 | 4.92E+01 |
| CEC'17 F26 | Mean | 8.70E+03 | 6.25E+03 | 1.33E+04 | 9.72E+03 | 4.56E+03 | **1.90E+03** | 3.32E+03 |
| | Std | 2.14E+02 | 2.75E+02 | 2.48E+03 | 2.83E+02 | 2.63E+02 | **1.64E+03** | 1.33E+02 |
| CEC'17 F27 | Mean | 8.54E+02 | 7.19E+02 | 1.56E+03 | 6.53E+02 | 7.15E+02 | 9.38E+02 | **6.54E+02** |
| | Std | 8.71E+01 | 3.79E+01 | 3.22E+02 | 2.24E+01 | 3.16E+01 | 7.76E+01 | **1.86E+01** |
| CEC'17 F28 | Mean | 6.39E+02 | 5.55E+02 | **5.12E+02** | 5.48E+02 | 9.53E+02 | 8.71E+02 | 5.49E+02 |
| | Std | 2.85E+01 | 3.16E+01 | **3.83E+01** | 3.51E+01 | 2.26E+03 | 4.95E+01 | 2.86E+01 |
| CEC'17 F29 | Mean | 3.95E+03 | 2.75E+03 | 4.43E+03 | 2.79E+03 | 1.97E+03 | 2.14E+03 | **1.93E+03** |
| | Std | 2.12E+02 | 2.48E+02 | 5.26E+02 | 2.43E+02 | 2.79E+02 | 3.61E+02 | **3.67E+02** |
| CEC'17 F30 | Mean | 4.96E+03 | 3.84E+03 | 9.13E+03 | 4.15E+03 | **2.76E+03** | 2.64E+04 | 3.81E+03 |
| | Std | 9.39E+02 | 8.49E+02 | 1.69E+03 | 1.83E+03 | **4.59E+02** | 1.07E+04 | 5.38E+02 |
| | +/ ≈ /− | 3/1/26 | 7/5/18 | 9/3/18 | 6/9/15 | 9/10/11 | 4/8/18 | – |

**Table 11**
The algorithm complexity of AE.

| Dimension | $T0$ | $T1$ | $\overline{T2}$ | $(\overline{T2}-T1)/T0$ |
|---|---|---|---|---|
| $D = 30$ | 3.07E−02 | 1.18E+00 | 6.10E+00 | 1.60E+02 |
| $D = 50$ | 3.07E−02 | 1.51E+00 | 6.49E+00 | 1.62E+02 |
| $D = 100$ | 3.07E−02 | 3.64E+00 | 10.03E+00 | 2.08E+02 |

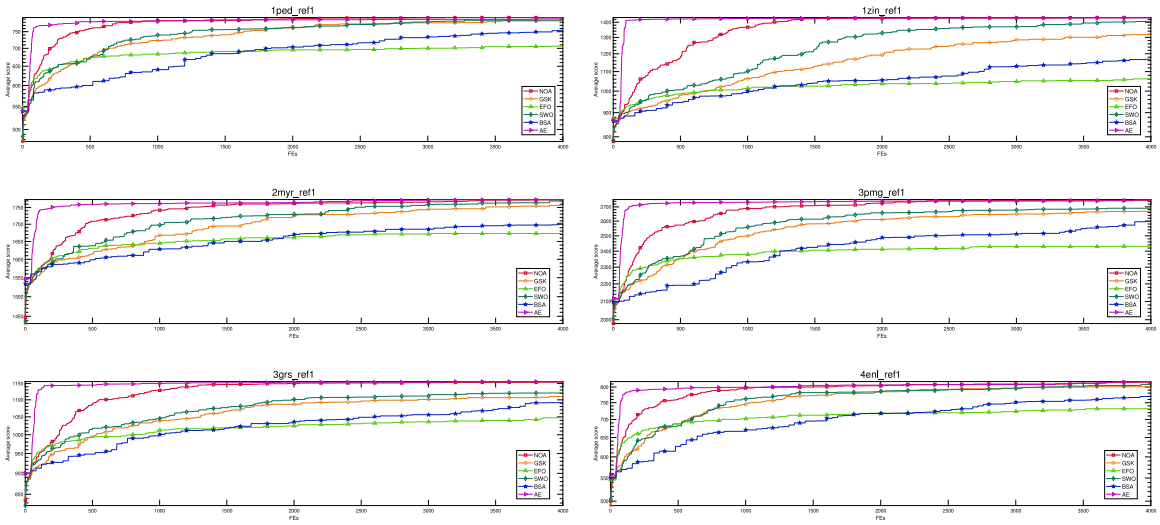



**Fig. 14.** The average fitness values found by the top 6 algorithms during the training of the HMM for the sequences.


**Fig. 15.** The partial results were obtained after AE aligned 1ped_ref1, 2myr_ref1, 3pmg_ref1, and 4nel_ref sequences (in order from top to bottom).

### 5.1. Pressure vessel design

The pressure vessel consists of a cylindrical vessel and two hemispherical heads, as shown in Fig. 16. The optimization objective is to minimize the total cost of the pressure vessel design (Moyya et al., 2023). There are four variables to be optimized: the thickness of the shell ($Ts$), the thickness of the head ($Th$), the inner radius ($R$), and the length of the cylindrical portion of the vessel ($L$), excluding the heads. $Ts$ and $Th$ are integer multiples of 0.0625 inch, while $R$ and $L$ are continuous variables. Therefore, the mathematical model of this

**Table 12**
The gene sequence information.

| Sequence name | Number of sub-sequences | Length range | Dimension |
|---|---|---|---|
| 1ped_ref1 | 3 | [981,1122] | 22 906 |
| 1zin_ref1 | 4 | [618,648] | 13 233 |
| 2myr_ref1 | 4 | [1020,1422] | 29 026 |
| 3pmg_ref1 | 4 | [1620,1701] | 34 721 |
| 3grs_ref1 | 4 | [657,711] | 14 525 |
| 4enl_ref1 | 3 | [1002,1263] | 25 779 |

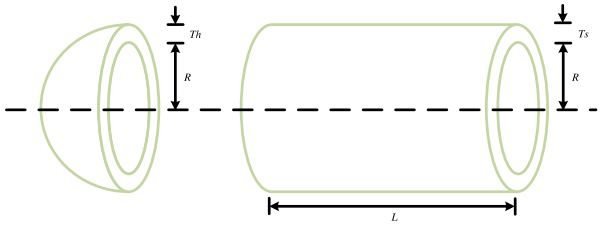



**Fig. 16.** The schematic view of the pressure vessel design problem.

problem is represented as Eq. (19).

$$\begin{aligned}
&\text{Solution representation}: && X=[X_1,X_2,X_3,X_4]=[Ts,Th,R,L]\\
&\text{Objective function}: && f(X)=0.6224X_1X_3X_4+1.7781X_2X_3^2\\
&&&\quad +3.1661X_1^2X_4+19.84X_1^2X_3\\
&\text{Subject to}: && g_1(X)=-X_1+0.0193X_3\le 0\\
&&& g_2(X)=-X_2+0.00954X_3\le 0\\
&&& g_3(X)=-\pi X_3^2X_4-\tfrac{4}{3}\pi X_3^3+1296000\le 0\\
&&& g_4(X)=X_4-240\le 0\\
&\text{Variable range}: && 1\times 0.0625\le X_1,X_2\le 100\times 0.0625\\
&&& 10\le X_3,X_4\le 200
\end{aligned}\tag{19}$$

Table 14 shows the results obtained by the top 6 algorithms for solving the pressure vessel design problem. It is obvious that in terms of best, mean, and std metrics, AE is superior compared to the other algorithms. The $p$-value results were obtained through the Wilcoxon rank sum test, and there was a significant difference in AE and competitor results. Moreover, it is proven from Fig. 17 that AE is more advantageous in convergence and stability when solving this problem.

### 5.2. Tension/compression spring design

The tension/compression spring design is depicted in Fig. 18, and its optimization objective is to minimize the weight of the spring (Tzanetos and Blondin, 2023). The problem has three constraints: shear stress, surge frequency, and minimum deflection, and three variables to be optimized: the wire diameter ($d$), the mean coil diameter ($D$), and the number of active coils ($N$). The mathematical model of this problem is described as Eq. (20).

$$\begin{aligned}
&\text{Solution representation}: && X=[X_1,X_2,X_3]=[d,D,N]\\
&\text{Objective function}: && f(X)=(X_3+2)X_2X_1^2\\
&\text{Subject to}: && g_1(X)=1-\frac{X_2^3X_3}{71785X_1^4}\le 0\\
&&& g_2(X)=\frac{4X_2^2-X_1X_2}{12566\left(X_2X_1^3-X_1^4\right)}+\frac{1}{5108X_1^2}\le 0\\
&&& g_3(X)=1-\frac{140.45X_1}{X_2^2X_3}\le 0\\
&&& g_4(X)=\frac{X_1+X_2}{1.5}-1\le 0\\
&\text{Variable range}: && 0.05\le X_1\le 2.00\\
&&& 0.25\le X_2\le 1.30\\
&&& 2.00\le X_3\le 15.0
\end{aligned}\tag{20}$$

Table 15 presents the comparison results of the top 6 algorithms for solving the tension/compression spring design problem. Obviously, AE has found the solution with the smallest weight for this problem. The mean metric shows that the average quality of the solutions obtained by the AE is better than that of the competitors. The std metric indicates that the standard deviation obtained from the AE is the smallest. The $p$-value metric proves that the results of the AE are statistically significant. Furthermore, Fig. 19 also confirms that the performance of AE is more stable and competitive.

### 5.3. Welded beam design

The purpose of the welded beam design problem is to make the manufacturing cost of the welded beam as small as possible (Deb, 1991). The structure of the welded beam is depicted in Fig. 20. The optimization constraints for this problem are the shear stress ($\tau$) and bending stress ($\theta$) in the beam, the buckling load on the reinforcement ($Pc$), and the end deflection ($\delta$) of the beam. There are four variables to be optimized in this problem: the thickness of the weld ($h$), the length of the clamped bar ($l$), the height of the bar ($t$), and the thickness of the bar ($b$). Therefore, the mathematical model of this problem is described as Eq. (21).

$$\begin{aligned}
&\text{Solution representation}: && X=[X_1,X_2,X_3,X_4]=[h,l,t,b]\\
&\text{Objective function}: && f(X)=1.10471X_1^2X_2+0.04811X_3X_4(14.0+X_2)\\
&\text{Subject to}: && g_1(X)=\tau(X)-13600\le 0\\
&&& g_2(X)=\sigma(X)-30000\le 0\\
&&& g_3(X)=\delta(X)-0.25\le 0\\
&&& g_4(X)=X_1-X_4\le 0\\
&&& g_5(X)=6000-Pc(X)\le 0\\
&&& g_6(X)=0.125-X_1\le 0\\
&&& g_7(X)=f(X)-5.0\le 0\\
&&& \tau(X)=\sqrt{(\tau')^2+(\tau'')^2+\frac{(X_2\tau'\tau'')}{\sqrt{0.25\left(X_2^2+(X_1+X_3)^2\right)}}}\\
&&& \tau'=\frac{6000}{\sqrt{2X_1X_2}},\ \tau''=\frac{6000(14+0.5X_2)\sqrt{0.25\left(X_2^2+(X_1+X_3)^2\right)}}{2\left(0.707X_1X_2\left(\frac{X_2^2}{12}+0.25(X_1+X_2)^2\right)\right)}\\
&&& Pc(X)=64746.022\left(1-0.0282346X_3\right)X_3X_4^3\\
&&& \sigma(X)=\frac{504000}{X_3^2X_4},\ \delta(X)=\frac{65856000}{(30\times 10^6)X_4X_3^3}\\
&\text{Variable range}: && 0.1\le X_1,X_4\le 2.0\\
&&& 0.1\le X_2,X_3\le 10.0
\end{aligned}\tag{21}$$

Table 16 presents the results of the top 6 algorithms. The results indicate that AE can still obtain the optimal solution by relying on its powerful search operator when solving the design problem of welded beams. The $p$-value metric proves the statistically significant difference in the results between AE and competitors. Additionally, Fig. 21 again verifies the competitiveness and robustness of AE in solving engineering design problems, even if it only has one search operator. It should be noted that due to the inability to identify differences between algorithms by directly presenting convergence curves of the entire evaluation process, Figs. 17, 19, and 21 only present convergence curves of the final stage of evaluation. The curves no longer converge because solutions no longer improve at this stage.

## 6. Conclusion

How optimization algorithms can effectively extract and utilize the information hidden in candidate solutions is an interesting research topic that has motivated the proposal of many metaphor-based algorithms. These algorithms may introduce more search strategies, and excessive metaphorical ambiguity may also obscure the essence and innovation of the algorithms. Moreover, implementing the base

**Table 13**
The mean, standard deviation, and average times obtained by aligning gene sequences.

| Names | Metrics | NOA | GSK | EFO | SWO | BSA | AE |
|---|---|---|---|---|---|---|---|
| 1ped | Mean±Std | **7.973E+02 ± 1.095E+01** | 7.844E+02 ± 1.021E+01 | 7.063E+02 ± 1.155E+01 | 7.869E+02 ± 4.652E+00 | 7.543E+02 ± 3.119E+01 | 7.911E+02 ± 5.594E+00 |
| 1zin | Mean±Std | **1.432E+03 ± 6.754E+00** | 1.317E+03 ± 5.356E+01 | 1.062E+03 ± 1.402E+01 | 1.406E+03 ± 3.420E+01 | 1.168E+03 ± 6.209E+01 | 1.429E+03 ± 3.618E+00 |
| 2myr | Mean±Std | 1.773E+03 ± 7.308E+00 | 1.756E+03 ± 1.367E+01 | 1.673E+03 ± 1.776E+01 | 1.770E+03 ± 1.617E+01 | 1.703E+03 ± 3.121E+01 | **1.774E+03 ± 6.774E+00** |
| 3pmg | Mean±Std | 2.748E+03 ± 1.495E+01 | 2.672E+03 ± 2.588E+01 | 2.433E+03 ± 2.146E+01 | 2.690E+03 ± 3.686E+01 | 2.597E+03 ± 5.411E+01 | **2.753E+03 ± 1.596E+01** |
| 3grs | Mean±Std | 1.154E+03 ± 3.796E+00 | 1.110E+03 ± 1.214E+01 | 1.047E+03 ± 1.214E+01 | 1.121E+03 ± 2.726E+01 | 1.092E+03 ± 3.285E+01 | **1.156E+03 ± 5.883E+00** |
| 4enl | Mean±Std | 8.158E+02 ± 9.188E+00 | 8.009E+02 ± 5.727E+00 | 7.315E+02 ± 9.204E+00 | 8.086E+02 ± 1.490E+01 | 7.693E+02 ± 2.778E+01 | **8.181E+02 ± 9.609E+00** |
| | Average times | 99.73 s | 100.77 s | 95.60 s | 87.51 s | 83.19 s | 91.27 s |

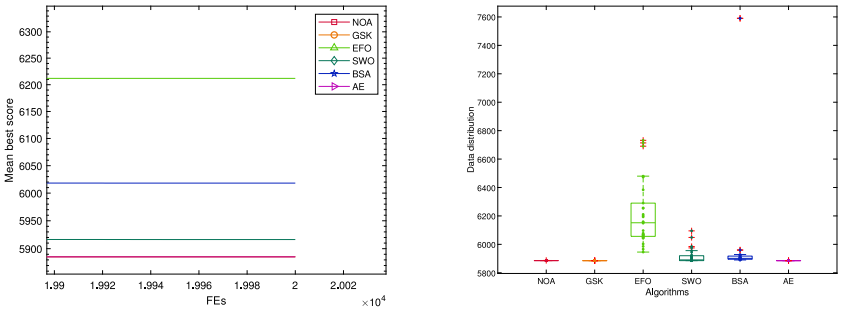



**Fig. 17.** The convergence curve and box plot of the top 6 algorithms for the pressure vessel design problem.

**Table 14**
The comparison results for the pressure vessel design problem.

| Algorithms | $Ts$ | $Th$ | $R$ | $L$ | Best | Mean | Std | $p$-value |
|---|---|---|---|---|---|---|---|---|
| NOA | 0.778172 | 0.384655 | 40.320091 | 199.993566 | 5885.383729 | 5885.865694 | 0.582582 | 3.01986E−11 |
| GSK | 0.778170 | 0.384650 | 40.319675 | 199.999313 | 5885.337244 | 5885.352486 | 0.010714 | 3.01986E−11 |
| EFO | 0.815494 | 0.403099 | 42.253589 | 174.723560 | 5952.243802 | 6283.913621 | 301.624273 | 3.01986E−11 |
| SWO | 0.778169 | 0.384649 | 40.319619 | 199.999992 | 5885.332797 | 5920.180370 | 78.360399 | 4.08395E−05 |
| BSA | 0.778228 | 0.385376 | 40.321072 | 200.000000 | 5888.102166 | 5962.782721 | 307.743126 | 3.01986E−11 |
| AE | **0.778169** | **0.384649** | **40.319619** | **200.000000** | **5885.332774** | **5885.332774** | **0.000000** | – |

**Table 15**
The comparison results for the tension/compression spring design problem.

| Algorithms | $d$ | $D$ | $N$ | Best | Mean | Std | $p$-value |
|---|---|---|---|---|---|---|---|
| NOA | 0.051831 | 0.360134 | 11.091887 | 0.01266621 | 0.01267164 | 0.00000568 | 1.093670E−10 |
| GSK | 0.051681 | 0.356522 | 11.300454 | 0.01266524 | 0.01266574 | 0.00000120 | 8.314609E−03 |
| EFO | 0.051136 | 0.343566 | 12.103793 | 0.01267086 | 0.01394566 | 0.00107273 | 3.019859E−11 |
| SWO | 0.051645 | 0.355657 | 11.351396 | 0.01266527 | 0.01276516 | 0.00012030 | 6.721954E−10 |
| BSA | 0.051714 | 0.357213 | 11.267092 | 0.01267392 | 0.01272098 | 0.00003849 | 3.019859E−11 |
| AE | **0.051689** | **0.356720** | **11.288843** | **0.01266523** | **0.01266554** | **0.00000046** | – |

**Table 16**
The comparison results for the welded beam design problem.

| Algorithms | $h$ | $l$ | $t$ | $b$ | Best | Mean | Std | $p$-value |
|---|---|---|---|---|---|---|---|---|
| NOA | 0.325499 | 1.937314 | 9.036599 | 0.205731 | 1.6523340 | 1.6524689 | 0.0000832 | 3.019859E−11 |
| GSK | 0.322997 | 1.954196 | 9.036624 | 0.205730 | 1.6523063 | 1.6523090 | 0.0000023 | 3.019859E−11 |
| EFO | 0.322557 | 1.957200 | 9.036624 | 0.205730 | 1.6523064 | 1.6536088 | 0.0021940 | 4.997954E−09 |
| SWO | 0.323379 | 1.951593 | 9.036624 | 0.205730 | 1.6523058 | 1.6525888 | 0.0005877 | 7.198788E−05 |
| BSA | 0.325964 | 1.933130 | 9.043303 | 0.205760 | 1.6533750 | 1.6608574 | 0.0065351 | 3.019859E−11 |
| AE | **0.323146** | **1.953179** | **9.036624** | **0.205730** | **1.6523057** | **1.6523057** | **0.0000000** | – |


**Fig. 18.** The schematic view of the tension/compression spring design problem.

vector adaptation is also an aspect they did not consider. Ultimately, the phenomenon of overly cumbersome, difficult-to-understand code and low optimization performance is inevitable. To solve the above problem, this paper proposes a novel optimization algorithm known as alpha evolution (AE). In AE, multiple operations for extracting and utilizing existing information are simultaneously integrated into the alpha operator. Without introducing other search components, this operator can independently undertake the exploration and exploitation functions of the algorithm. In terms of structure, this operator consists of three components: the adaptive base vector, the random step size, and the adaptive step size. Three components extract and utilize information from the perspectives of matrix and dimension, problem space, and differences among solutions, respectively. Moreover, this study designed a complete adaptive mechanism for the base vector of the alpha operator.

### 6.1. Findings

With increasingly severe practical problems, the reliability and convergence of algorithms should be widely verified. Therefore, this research conducted a comprehensive and rigorous number of experiments, some of which did not appear in papers on similar techniques. The experiments evaluate the performance of AE in terms of search bias, invariance, scalability, parameter sensitivity, search behavior, exploration and exploitation, convergence, statistics, complexity, and application scenarios. AE has been verified to have no structural bias toward the origin, and it can also work normally in areas far from 0. AE exhibits shift and scale invariance when performing linear transformations on the problem space. These characteristics enable AE to work reliably and effectively in difficult practical applications. In addition, AE also demonstrates good scalability, and its convergence behavior does not show significant changes with the increase in problem dimensions. The parameter sensitivity analysis experiment found that AE is not sensitive to small-scale changes in population size. The search behavior and qualitative experiments have verified that AE has reliable convergence characteristics, and regardless of where the

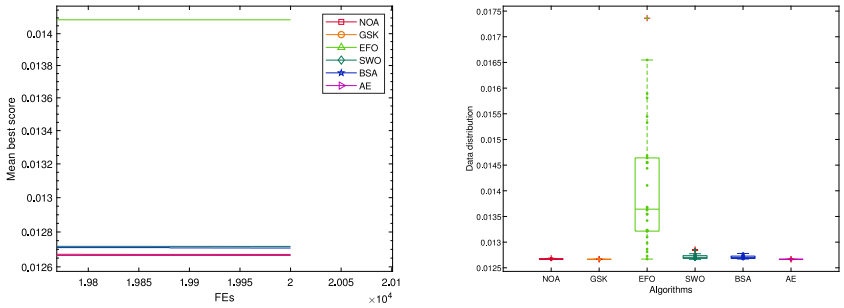



**Fig. 19.** The convergence curve and box plot of the top 6 algorithms for the tension/compression spring design problem.

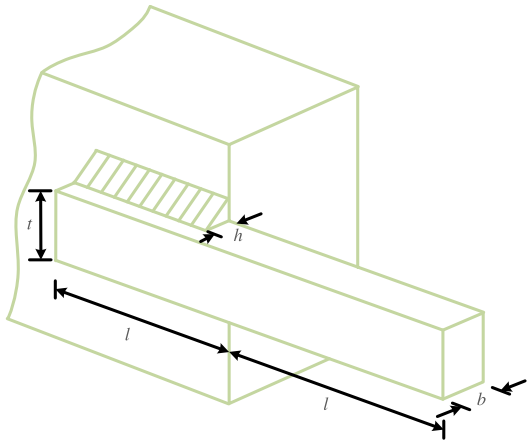



**Fig. 20.** The schematic view of the welded beam design problem.

initial population is generated, they will eventually approach the global optimal region. Furthermore, the multi-point exploitation of AE and the behavior of jumping out of local optima also demonstrate the potential advantages of AE in engineering applications. A moderate balance between exploration and exploitation can effectively improve the performance of algorithms. The experiment found that the search behavior of AE is not significantly affected by dimensional changes, and it will adjust the proportion of exploration and exploitation according to the problem structure to obtain higher-quality solutions. To verify the numerical optimization capability of AE, it was tested simultaneously with 101 optimization algorithms on the CEC'17 test suite in 30, 50, and 100 dimensions. Through the obtained convergence curves, it was found that AE can effectively avoid the risk of falling into local optima, thereby achieving better optimization performance. Moreover, the box plot results demonstrate that this optimization performance is reliable and stable. However, based on the results of the Friedman test and the Wilcoxon rank sum test, AE performed slightly worse than several algorithms at 30 dimensions but significantly better than the compared 100 algorithms at 50 and 100 dimensions. This not only demonstrates that AE has a competitive numerical optimization ability but also indicates the potential of AE for solving large-scale optimization problems. Furthermore, six adaptive algorithms with more robust adaptability to problems have been included in the experimental scope to provide readers with more comprehensive experimental insights. Although AE cannot find the global optimum for certain problems, it can still adapt to problems with different characteristics and maintain high competitiveness. Finally, the research also found that some algorithms claimed their performance was superior to some state-of-the-art algorithms, but their performance was still inferior to AE on the specific problems this paper controlled. However, the no free lunch theorem can reasonably explain this phenomenon.

To verify the practical application value of AE, it is first applied to the multiple sequence alignment problem. The result shows that AE can effectively solve the MSA problem and provide algorithmic support for bioinformatics. The high-quality aligned sequences obtained after optimization can be used in disease diagnosis, vaccine development, gene tracing, etc. Additionally, the experiment also reveals that AE can obtain goals at a faster rate, while other algorithms require longer iterations for the same goals. This demonstrates that AE has competitive real-world application potential and value, especially for real-time decision-making and dynamically changing application scenarios. Further, AE was used to solve three classic engineering design problems. The results demonstrate that AE is more competitive in terms of convergence and robustness.

### 6.2. Research limitations

In summary, AE does not use metaphors as innovation. The amount of code for it is not significant, and it can be applied to different problems with only simple modifications. It has only one core operator that approximates the optimal solution, which avoids the situation where the performances of different operators cancel each other. However, AE is only applied to single-objective constrained optimization problems and requires additional modifications when applied to other problems. In terms of computing complexity, AE does not have a significant advantage. Because the diagonal operation will incur additional computational costs as the dimension increases. Similarly, the generalization of the AE algorithm may be affected as well. AE is essentially a stochastic search technique, so it may fall into local optima in certain problems. While the performance of AE in this study is competitive, it does not necessarily imply similar performance across all problems. In addition, computational resources, problem size, and solution accuracy may all be difficulties AE encounters in real-world scenarios. Considering the fairness of the experiment, this work also does not provide a version of AE with the maximum iterations as the termination criterion.

### 6.3. Recommendations for future research

In the future, AE will develop multi-objective, binary, and parallel versions to expand its application scope. Furthermore, further improving AE's performance while reducing its computational complexity is another development direction for AE.

## CRediT authorship contribution statement

**Hao Gao:** Writing – review & editing, Writing – original draft, Visualization, Validation, Software, Methodology, Investigation, Formal analysis, Data curation, Conceptualization. **Qingke Zhang:** Writing – review & editing, Writing – original draft, Supervision, Software, Resources, Project administration, Methodology, Investigation, Funding acquisition, Data curation, Conceptualization.

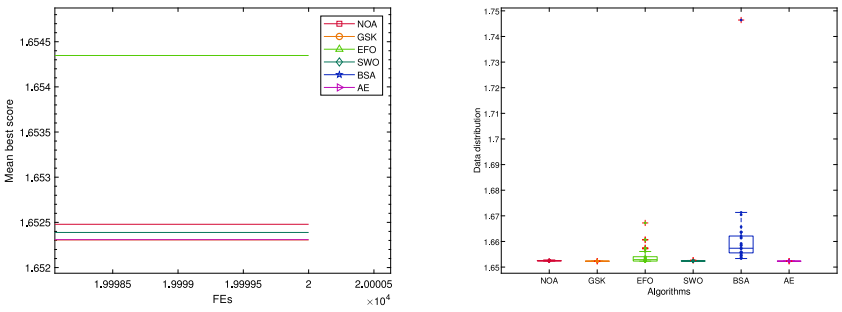



**Fig. 21.** The convergence curve and box plot of the top 6 algorithms for the welded beam design problem.

## Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

## Data availability

Data will be made available on request.

## Acknowledgments

This work is supported by the National Natural Science Foundation of China (No. 62006144).

## References

Ab Wahab, M.N., Nefti-Meziani, S., Atyabi, A., 2015. A comprehensive review of swarm optimization algorithms. PLoS One 10 (5), e0122827. http://dx.doi.org/10.1371/journal.pone.0122827.

Abbaszadeh Shahri, A., Asheghi, R., Khorsand Zak, M., 2021. A hybridized intelligence model to improve the predictability level of strength index parameters of rocks. Neural Comput. Appl. 33, 3841–3854. http://dx.doi.org/10.1007/s00521-020-05223-9.

Abbaszadeh Shahri, A., Khorsand Zak, M., Abbaszadeh Shahri, H., 2022a. A modified firefly algorithm applying on multi-objective radial-based function for blasting. Neural Comput. Appl. 1–17. http://dx.doi.org/10.1007/s00521-021-06544-z.

Abbaszadeh Shahri, A., Pashamohammadi, F., Asheghi, R., Abbaszadeh Shahri, H., 2022b. Automated intelligent hybrid computing schemes to predict blasting induced ground vibration. Eng. Comput. 38 (Suppl 4), 3335–3349. http://dx.doi.org/10.1007/s00366-021-01444-1.

Abdel-Basset, M., El-Shahat, D., Jameel, M., Abouhawwash, M., 2023a. Exponential distribution optimizer (EDO): A novel math-inspired algorithm for global optimization and engineering problems. Artif. Intell. Rev. 1–72. http://dx.doi.org/10.1007/s10462-023-10403-9.

Abdel-Basset, M., El-Shahat, D., Jameel, M., Abouhawwash, M., 2023b. Young's double-slit experiment optimizer: A novel metaheuristic optimization algorithm for global and constraint optimization problems. Comput. Methods Appl. Mech. Engrg. 403, 115652. http://dx.doi.org/10.1016/j.cma.2022.115652.

Abdel-Basset, M., Mohamed, R., Azeem, S.A.A., Jameel, M., Abouhawwash, M., 2023c. Kepler optimization algorithm: A new metaheuristic algorithm inspired by Kepler's laws of planetary motion. Knowl.-Based Syst. 110454. http://dx.doi.org/10.1016/j.knosys.2023.110454.

Abdel-Basset, M., Mohamed, R., Jameel, M., Abouhawwash, M., 2023d. Nutcracker optimizer: A novel nature-inspired metaheuristic algorithm for global optimization and engineering design problems. Knowl.-Based Syst. 110248. http://dx.doi.org/10.1016/j.knosys.2022.110248.

Abdel-Basset, M., Mohamed, R., Jameel, M., Abouhawwash, M., 2023e. Spider wasp optimizer: A novel meta-heuristic optimization algorithm. Artif. Intell. Rev. 1–64. http://dx.doi.org/10.1007/s10462-023-10446-y.

Abdel-Basset, M., Mohamed, R., Sallam, K.M., Chakrabortty, R.K., 2022. Light spectrum optimizer: A novel physics-inspired metaheuristic optimization algorithm. Mathematics 10 (19), 3466. http://dx.doi.org/10.3390/math10193466.

Abdollahzadeh, B., Soleimanian Gharehchopogh, F., Mirjalili, S., 2021. Artificial gorilla troops optimizer: A new nature-inspired metaheuristic algorithm for global optimization problems. Int. J. Intell. Syst. 36 (10), 5887–5958. http://dx.doi.org/10.1002/int.22535.

Abedinpourshotorban, H., Shamsuddin, S.M., Beheshti, Z., Jawawi, D.N., 2016. Electromagnetic field optimization: A physics-inspired metaheuristic optimization algorithm. Swarm Evol. Comput. 26, 8–22. http://dx.doi.org/10.1016/j.swevo.2015.07.002.

Abualigah, L., Abd Elaziz, M., Sumari, P., Geem, Z.W., Gandomi, A.H., 2022. Reptile search algorithm (RSA): A nature-inspired meta-heuristic optimizer. Expert Syst. Appl. 191, 116158. http://dx.doi.org/10.1016/j.eswa.2021.116158.

Abualigah, L., Diabat, A., Mirjalili, S., Abd Elaziz, M., Gandomi, A.H., 2021a. The arithmetic optimization algorithm. Comput. Methods Appl. Mech. Engrg. 376, 113609. http://dx.doi.org/10.1016/j.cma.2020.113609.

Abualigah, L., Yousri, D., Abd Elaziz, M., Ewees, A.A., Al-Qaness, M.A., Gandomi, A.H., 2021b. Aquila optimizer: A novel meta-heuristic optimization algorithm. Comput. Ind. Eng. 157, 107250. http://dx.doi.org/10.1016/j.cie.2021.107250.

Agushaka, J.O., Ezugwu, A.E., Abualigah, L., 2022. Dwarf mongoose optimization algorithm. Comput. Methods Appl. Mech. Engrg. 391, 114570. http://dx.doi.org/10.1016/j.cma.2022.114570.

Ahmadianfar, I., Bozorg-Haddad, O., Chu, X., 2020. Gradient-based optimizer: A new metaheuristic optimization algorithm. Inform. Sci. 540, 131–159. http://dx.doi.org/10.1016/j.ins.2020.06.037.

Ahmadianfar, I., Heidari, A.A., Gandomi, A.H., Chu, X., Chen, H., 2021. RUN beyond the metaphor: An efficient optimization algorithm based on Runge Kutta method. Expert Syst. Appl. 181, 115079. http://dx.doi.org/10.1016/j.eswa.2021.115079.

Ahmadianfar, I., Heidari, A.A., Noshadian, S., Chen, H., Gandomi, A.H., 2022. INFO: An efficient optimization algorithm based on weighted mean of vectors. Expert Syst. Appl. 116516. http://dx.doi.org/10.1016/j.eswa.2022.116516.

Al-Betar, M.A., Alyasseri, Z.A.A., Awadallah, M.A., Abu Doush, I., 2021. Coronavirus herd immunity optimizer (CHIO). Neural Comput. Appl. 33, 5011–5042. http://dx.doi.org/10.1007/s00521-020-05296-6.

Arora, S., Singh, S., 2019. Butterfly optimization algorithm: A novel approach for global optimization. Soft Comput. 23, 715–734. http://dx.doi.org/10.1007/s00500-018-3102-4.

Asheghi, R., Hosseini, S.A., Saneie, M., Shahri, A.A., 2020. Updating the neural network sediment load models using different sensitivity analysis methods: A regional application. J. Hydroinform. 22 (3), 562–577. http://dx.doi.org/10.2166/hydro.2020.098.

Askari, Q., Saeed, M., Younas, I., 2020a. Heap-based optimizer inspired by corporate rank hierarchy for global optimization. Expert Syst. Appl. 161, 113702. http://dx.doi.org/10.1016/j.eswa.2020.113702.

Askari, Q., Younas, I., Saeed, M., 2020b. Political optimizer: A novel socio-inspired meta-heuristic for global optimization. Knowl.-Based Syst. 195, 105709. http://dx.doi.org/10.1016/j.knosys.2020.105709.

Atashpaz-Gargari, E., Lucas, C., 2007. Imperialist competitive algorithm: An algorithm for optimization inspired by imperialistic competition. In: 2007 IEEE Congress on Evolutionary Computation. IEEE, pp. 4661–4667. http://dx.doi.org/10.1109/cec.2007.4425083.

Azizi, M., 2021. Atomic orbital search: A novel metaheuristic algorithm. Appl. Math. Model. 93, 657–683. http://dx.doi.org/10.1016/j.apm.2020.12.021.

Azizi, M., Aickelin, U., A. Khorshidi, H., Baghalzadeh Shishehgarkhaneh, M., 2023a. Energy valley optimizer: A novel metaheuristic algorithm for global and engineering optimization. Sci. Rep. 13 (1), 226. http://dx.doi.org/10.1038/s41598-022-27344-y.

Azizi, M., Talatahari, S., Gandomi, A.H., 2023b. Fire hawk optimizer: A novel metaheuristic algorithm. Artif. Intell. Rev. 56 (1), 287–363. http://dx.doi.org/10.1007/s10462-022-10173-w.

Bäck, T., Fogel, D.B., Michalewicz, Z., 2018. Evolutionary Computation 1: Basic Algorithms and Operators. CRC Press, https://oa.mg/work/1603762601.

Banzhaf, W., et al., 2006. From artificial evolution to computational evolution: A research agenda. Nature Rev. Genet. 7 (9), 729–735. http://dx.doi.org/10.1038/nrg1921.

Barua, S., Merabet, A., 2024. Lévy arithmetic algorithm: An enhanced metaheuristic algorithm and its application to engineering optimization. Expert Syst. Appl. 241, 122335. http://dx.doi.org/10.1016/j.eswa.2023.122335.

Bayraktar, Z., Komurcu, M., Werner, D.H., 2010. Wind driven optimization (WDO): A novel nature-inspired optimization algorithm and its application to electromagnetics. In: 2010 IEEE Antennas and Propagation Society International Symposium. IEEE, pp. 1–4. http://dx.doi.org/10.1109/aps.2010.5562213.

Benbraika, M.K., Kraa, O., Himeur, Y., Telli, K., Atalla, S., Mansoor, W., 2024. Interference management based on meta-heuristic algorithms in 5G device-to-device communications. Computers 13 (2), 44. http://dx.doi.org/10.3390/computers13020044.

Biswas, A., Mishra, K., Tiwari, S., Misra, A., 2013. Physics-inspired optimization algorithms: A survey. J. Optim. 2013, http://dx.doi.org/10.1155/2013/438152.

Bouchekara, H., 2014. Optimal power flow using black-hole-based optimization approach. Appl. Soft Comput. 24, 879–888. http://dx.doi.org/10.1016/j.asoc.2014.08.056.

Brahim, B., Kobayashi, M., Al Ali, M., Khatir, T., Elmeliani, M.E.A.E., 2024. Metaheuristic optimization algorithms: An overview. HCMCOU J. Sci.-Adv. Comput. Struct. http://dx.doi.org/10.46223/HCMCOUJS.acs.en.14.1.47.2024.

Braik, M., Hammouri, A., Atwan, J., Al-Betar, M.A., Awadallah, M.A., 2022a. White shark optimizer: A novel bio-inspired meta-heuristic algorithm for global optimization problems. Knowl.-Based Syst. 243, 108457. http://dx.doi.org/10.1016/j.knosys.2022.108457.

Braik, M., Ryalat, M.H., Al-Zoubi, H., 2022b. A novel meta-heuristic algorithm for solving numerical optimization problems: Ali Baba and the forty thieves. Neural Comput. Appl. 34 (1), 409–455. http://dx.doi.org/10.1007/s00521-021-06392-x.

Cai, Z., Gao, S., Yang, X., Yang, G., Cheng, S., Shi, Y., 2022. Alternate search pattern-based brain storm optimization. Knowl.-Based Syst. 238, 107896. http://dx.doi.org/10.1016/j.knosys.2021.107896.

Can, Ü., Alataş, B., 2015. Physics Based Metaheuristic Algorithms for Global Optimization. American Institute of Science, https://api.semanticscholar.org/CorpusID:14138620.

Caraffini, F., Neri, F., 2019. A study on rotation invariance in differential evolution. Swarm Evol. Comput. 50, 100436. http://dx.doi.org/10.1016/j.swevo.2018.08.013.

Castelli, M., Manzoni, L., Mariot, L., Nobile, M.S., Tangherloni, A., 2022. Salp swarm optimization: A critical review. Expert Syst. Appl. 189, 116029. http://dx.doi.org/10.1016/j.eswa.2021.116029.

Chen, D., Ge, Y., Wan, Y., Deng, Y., Chen, Y., Zou, F., 2022. Poplar optimization algorithm: A new meta-heuristic optimization technique for numerical optimization and image segmentation. Expert Syst. Appl. 200, 117118. http://dx.doi.org/10.1016/j.eswa.2022.117118.

Cheng, S., Shi, Y., Qin, Q., Zhang, Q., Bai, R., 2014. Population diversity maintenance in brain storm optimization algorithm. J. Artif. Intell. Soft Comput. Res. 4 (2), 83–97. http://dx.doi.org/10.1515/jaiscr-2015-0001.

Chivers, I., Sleightholme, J., Chivers, I., Sleightholme, J., 2015. An introduction to algorithms and the big O notation. In: Introduction to Programming with Fortran: With Coverage of Fortran 90, 95, 2003, 2008 and 77. Springer, pp. 359–364. http://dx.doi.org/10.1007/978-3-319-17701-4_23.

Chou, J.-S., Ngo, N.-T., 2017. Modified firefly algorithm for multidimensional optimization in structural design problems. Struct. Multidiscip. Optim. 55, 2013–2028. http://dx.doi.org/10.1007/s00158-016-1624-x.

Civicioglu, P., 2012. Transforming geocentric cartesian coordinates to geodetic coordinates by using differential search algorithm. Comput. Geosci. 46, 229–247. http://dx.doi.org/10.1016/j.cageo.2011.12.011.

Civicioglu, P., 2013. Backtracking search optimization algorithm for numerical optimization problems. Appl. Math. Comput. 219 (15), 8121–8144. http://dx.doi.org/10.1016/j.amc.2013.02.017.

Coello, C.A.C., 2002. Theoretical and numerical constraint-handling techniques used with evolutionary algorithms: A survey of the state of the art. Comput. Methods Appl. Mech. Engrg. 191 (11–12), 1245–1287. http://dx.doi.org/10.1016/s0045-7825(01)00323-1.

Črepinšek, M., Liu, S.-H., Mernik, M., 2013. Exploration and exploitation in evolutionary algorithms: A survey. ACM Comput. Surv. 45 (3), 1–33. http://dx.doi.org/10.1145/2480741.2480752.

Cuevas, E., Cienfuegos, M., Zaldívar, D., Pérez-Cisneros, M., 2013. A swarm optimization algorithm inspired in the behavior of the social-spider. Expert Syst. Appl. 40 (16), 6374–6384. http://dx.doi.org/10.1016/j.eswa.2013.05.041.

Cui, Y., Hu, W., Rahmani, A., 2024. Multi-robot path planning using learning-based artificial bee colony algorithm. Eng. Appl. Artif. Intell. 129, 107579. http://dx.doi.org/10.1016/j.engappai.2023.107579.

Das, S., Biswas, A., Dasgupta, S., Abraham, A., 2009. Bacterial foraging optimization algorithm: Theoretical foundations, analysis, and applications. In: Foundations of Computational Intelligence Volume 3: Global Optimization. Springer, pp. 23–55. http://dx.doi.org/10.1007/978-3-642-01085-9_2.

Das, S., Suganthan, P.N., 2010. Differential evolution: A survey of the state-of-the-art. IEEE Trans. Evol. Comput. 15 (1), 4–31. http://dx.doi.org/10.1109/tevc.2010.2059031.

de Anda-Suárez, J., Calzada-Ledesma, V., Ortiz-Aguilar, L., 2022. Metaheuristic–Boltzmannian optimization model: A new methodology for convergence using the Jensen–Shannon metric in continuous optimization problems. Swarm Evol. Comput. 75, 101193. http://dx.doi.org/10.1016/j.swevo.2022.101193.

Deb, K., 1991. Optimal design of a welded beam via genetic algorithms. AIAA J. 29 (11), 2013–2015. http://dx.doi.org/10.2514/3.10834.

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. http://dx.doi.org/10.1016/j.knosys.2022.110011.

Del Ser, J., et al., 2019. Bio-inspired computation: Where we stand and what's next. Swarm Evol. Comput. 48, 220–250. http://dx.doi.org/10.1016/j.swevo.2019.04.008.

Deng, L., Liu, S., 2023. Snow ablation optimizer: A novel metaheuristic technique for numerical optimization and engineering design. Expert Syst. Appl. 225, 120069. http://dx.doi.org/10.1016/j.eswa.2023.120069.

Deng, L., Liu, S., 2024. Deficiencies of the whale optimization algorithm and its validation method. Expert Syst. Appl. 237, 121544. http://dx.doi.org/10.1016/j.eswa.2023.121544.

Derrac, J., García, S., Molina, D., Herrera, F., 2011. A practical tutorial on the use of nonparametric statistical tests as a methodology for comparing evolutionary and swarm intelligence algorithms. Swarm Evol. Comput. 1 (1), 3–18. http://dx.doi.org/10.1016/j.swevo.2011.02.002.

Dorigo, M., 1992. Optimization, learning and natural algorithms (Ph. D. thesis). Politecnico di Milano, https://oa.mg/work/1492640216.

Eiben, A.E., Smith, J., 2015. From evolutionary computation to the evolution of things. Nature 521 (7553), 476–482. http://dx.doi.org/10.1038/nature14544.

Eskandar, H., Sadollah, A., Bahreininejad, A., Hamdi, M., 2012. Water cycle algorithm–a novel metaheuristic optimization method for solving constrained engineering optimization problems. Comput. Struct. 110, 151–166. http://dx.doi.org/10.1016/j.compstruc.2012.07.010.

Eusuff, M., Lansey, K., Pasha, F., 2006. Shuffled frog-leaping algorithm: A memetic meta-heuristic for discrete optimization. Eng. Optim. 38 (2), 129–154. http://dx.doi.org/10.1080/03052150500384759.

Ezugwu, A.E., Agushaka, J.O., Abualigah, L., Mirjalili, S., Gandomi, A.H., 2022. Prairie dog optimization algorithm. Neural Comput. Appl. 34 (22), 20017–20065. http://dx.doi.org/10.1007/s00521-022-07530-9.

Faramarzi, A., Heidarinejad, M., Mirjalili, S., Gandomi, A.H., 2020a. Marine predators algorithm: A nature-inspired metaheuristic. Expert Syst. Appl. 152, 113377. http://dx.doi.org/10.1016/j.eswa.2020.113377.

Faramarzi, A., Heidarinejad, M., Stephens, B., Mirjalili, S., 2020b. Equilibrium optimizer: A novel optimization algorithm. Knowl.-Based Syst. 191, 105190. http://dx.doi.org/10.1016/j.knosys.2019.105190.

Fogel, D.B., 1998. Artificial Intelligence Through Simulated Evolution. Wiley-IEEE Press, http://dx.doi.org/10.1109/9780470544600.ch7.

Forrest, S., 1993. Genetic algorithms: Principles of natural selection applied to computation. Science 261 (5123), 872–878. http://dx.doi.org/10.1126/science.8346439.

Friedman, M., 1937. The use of ranks to avoid the assumption of normality implicit in the analysis of variance. J. Amer. Statist. Assoc. 32 (200), 675–701. http://dx.doi.org/10.1080/01621459.1937.10503522.

Friedman, M., 1940. A comparison of alternative tests of significance for the problem of m rankings. Ann. Math. Stat. 11 (1), 86–92. http://dx.doi.org/10.1214/aoms/1177731944.

Gandomi, A.H., Alavi, A.H., 2012. Krill herd: A new bio-inspired optimization algorithm. Commun. Nonlinear Sci. Numer. Simul. 17 (12), 4831–4845. http://dx.doi.org/10.1016/j.cnsns.2012.05.010.

Gao, H., Zhang, Q., Bu, X., Zhang, H., 2024. Quadruple parameter adaptation growth optimizer with integrated distribution, confrontation, and balance features for optimization. Expert Syst. Appl. 235, 121218. http://dx.doi.org/10.1016/j.eswa.2023.121218.

Geem, Z.W., Kim, J.H., Loganathan, G.V., 2001. A new heuristic optimization algorithm: Harmony search. Simulation 76 (2), 60–68. http://dx.doi.org/10.1177/003754970107600201.

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. http://dx.doi.org/10.1016/j.knosys.2022.109484.

Han, M., Du, Z., Yuen, K.F., Zhu, H., Li, Y., Yuan, Q., 2024. Walrus optimizer: A novel nature-inspired metaheuristic algorithm. Expert Syst. Appl. 239, 122413. http://dx.doi.org/10.1016/j.eswa.2023.122413.

Hansen, N., Müller, S.D., Koumoutsakos, P., 2003. Reducing the time complexity of the derandomized evolution strategy with covariance matrix adaptation (CMA-ES). Evol. Comput. 11 (1), 1–18. http://dx.doi.org/10.1162/106365603321828970.

Hansen, N., Ostermeier, A., 1996. Adapting arbitrary normal mutation distributions in evolution strategies: The covariance matrix adaptation. In: Proceedings of IEEE International Conference on Evolutionary Computation. IEEE, pp. 312–317. http://dx.doi.org/10.1109/ICEC.1996.542381.

Harifi, S., Mohammadzadeh, J., Khalilian, M., Ebrahimnejad, S., 2021. Giza pyramids construction: An ancient-inspired metaheuristic algorithm for optimization. Evol. Intell. 14, 1743–1761. http://dx.doi.org/10.1007/s12065-020-00451-3.

Hashim, F.A., Houssein, E.H., Hussain, K., Mabrouk, M.S., Al-Atabany, W., 2022. Honey badger algorithm: New metaheuristic algorithm for solving optimization problems. Math. Comput. Simulation 192, 84–110. http://dx.doi.org/10.1016/j.matcom.2021.08.013.

Hashim, F.A., Hussien, A.G., 2022. Snake optimizer: A novel meta-heuristic optimization algorithm. Knowl.-Based Syst. 242, 108320. http://dx.doi.org/10.1016/j.knosys.2022.108320.

Hashim, F.A., Mostafa, R.R., Hussien, A.G., Mirjalili, S., Sallam, K.M., 2023. Fick's law algorithm: A physical law-based algorithm for numerical optimization. Knowl.-Based Syst. 260, 110146. http://dx.doi.org/10.1016/j.knosys.2022.110146.
Hayyolalam, V., Kazem, A.A.P., 2020. Black widow optimization algorithm: A novel meta-heuristic approach for solving engineering optimization problems. Eng. Appl. Artif. Intell. 87, 103249. http://dx.doi.org/10.1016/j.engappai.2019.103249.
He, S., Wu, Q.H., Saunders, J.R., 2009. Group search optimizer: An optimization algorithm inspired by animal searching behavior. IEEE Trans. Evol. Comput. 13 (5), 973–990. http://dx.doi.org/10.1109/tevc.2009.2011992.
Heidari, A.A., Mirjalili, S., Faris, H., Aljarah, I., Mafarja, M., Chen, H., 2019. Harris hawks optimization: Algorithm and applications. Future Gener. Comput. Syst. 97, 849–872. http://dx.doi.org/10.1016/j.future.2019.02.028.
Houssein, E.H., Saad, M.R., Hashim, F.A., Shaban, H., Hassaballah, M., 2020. Lévy flight distribution: A new metaheuristic algorithm for solving engineering optimization problems. Eng. Appl. Artif. Intell. 94, 103731. http://dx.doi.org/10.1016/j.engappai.2020.103731.
Jia, H., Peng, X., Lang, C., 2021. Remora optimization algorithm. Expert Syst. Appl. 185, 115665. http://dx.doi.org/10.1016/j.eswa.2021.115665.
Johnvictor, A.C., Durgamahanthi, V., Pariti Venkata, R.M., Jethi, N., 2022. Critical review of bio-inspired optimization techniques. Wiley Interdiscip. Rev. Comput. Stat. 14 (1), e1528. http://dx.doi.org/10.1002/wics.1528.
Kadavy, T., Viktorin, A., Kazikova, A., Pluhacek, M., Senkerik, R., 2022. Impact of boundary control methods on bound-constrained optimization benchmarking. IEEE Trans. Evol. Comput. 26 (6), 1271–1280. http://dx.doi.org/10.1109/tevc.2022.3204412.
Karaboga, D., Basturk, B., 2007. A powerful and efficient algorithm for numerical function optimization: Artificial bee colony (ABC) algorithm. J. Global Optim. 39, 459–471. http://dx.doi.org/10.1007/s10898-007-9149-x.
Karimkashi, S., Kishk, A.A., 2010. Invasive weed optimization and its features in electromagnetics. IEEE Trans. Antennas and Propagation 58 (4), 1269–1278. http://dx.doi.org/10.1109/tap.2010.2041163.
Karplus, K., Barrett, C., Hughey, R., 1998. Hidden Markov models for detecting remote protein homologies. Bioinformatics (Oxford, England) 14 (10), 846–856. http://dx.doi.org/10.1093/bioinformatics/14.10.846.
Kennedy, J., Eberhart, R., 1995. Particle swarm optimization. In: Proceedings of ICNN'95-International Conference on Neural Networks, Vol. 4. IEEE, pp. 1942–1948. http://dx.doi.org/10.1109/icnn.1995.488968.
Khanduja, N., Bhushan, B., 2021. Recent advances and application of metaheuristic algorithms: A survey (2014–2020). Metaheuristic Evol. Comput. Algorithms Appl. 207–228. http://dx.doi.org/10.1007/978-981-15-7571-6_10.
Khishe, M., Mosavi, M.R., 2020. Chimp optimization algorithm. Expert Syst. Appl. 149, 113338. http://dx.doi.org/10.1016/j.eswa.2020.113338.
Khoshniat, N., Jamarani, A., Ahmadzadeh, A., Haghi Kashani, M., Mahdipour, E., 2024. Nature-inspired metaheuristic methods in software testing. Soft Comput. 28 (2), 1503–1544. http://dx.doi.org/10.1007/s00500-023-08382-8.
Kiran, M.S., 2015. TSA: Tree-seed algorithm for continuous optimization. Expert Syst. Appl. 42 (19), 6686–6698. http://dx.doi.org/10.1016/j.eswa.2015.04.055.
Kirkpatrick, S., Gelatt, Jr., C.D., Vecchi, M.P., 1983. Optimization by simulated annealing. Science 220 (4598), 671–680. http://dx.doi.org/10.1126/science.220.4598.671.
Kononova, A.V., Corne, D.W., De Wilde, P., Shneer, V., Caraffini, F., 2015. Structural bias in population-based algorithms. Inform. Sci. 298, 468–490. http://dx.doi.org/10.1016/j.ins.2014.11.035.
Koza, J.R., 1994. Genetic programming as a means for programming computers by natural selection. Stat. Comput. 4, 87–112. http://dx.doi.org/10.1007/bf00175355.
Kudela, J., 2023. The evolutionary computation methods no one should use. http://dx.doi.org/10.48550/arXiv.2301.01984, arXiv preprint arXiv:2301.01984.
Kwakye, B.D., Li, Y., Mohamed, H.H., Baidoo, E., Asenso, T.Q., 2024. Particle guided metaheuristic algorithm for global optimization and feature selection problems. Expert Syst. Appl. 123362. http://dx.doi.org/10.1016/j.eswa.2024.123362.
Lee, Z.-J., Su, S.-F., Chuang, C.-C., Liu, K.-H., 2008. Genetic algorithm with ant colony optimization (GA-ACO) for multiple sequence alignment. Appl. Soft Comput. 8 (1), 55–78. http://dx.doi.org/10.1016/j.asoc.2006.10.012.
Li, S., Chen, H., Wang, M., Heidari, A.A., Mirjalili, S., 2020. Slime mould algorithm: A new method for stochastic optimization. Future Gener. Comput. Syst. 111, 300–323. http://dx.doi.org/10.1016/j.future.2020.03.055.
Li, Y., Li, G., 2020. Differential evolutionary algorithm with an evolutionary state estimation method and a two-level selection mechanism. Soft Comput. 24 (15), 11561–11581. http://dx.doi.org/10.1007/s00500-019-04621-z.
Ma, Z., Wu, G., Suganthan, P.N., Song, A., Luo, Q., 2023. Performance assessment and exhaustive listing of 500+ nature-inspired metaheuristic algorithms. Swarm Evol. Comput. 77, 101248. http://dx.doi.org/10.1016/j.swevo.2023.101248.
Mahdavi-Meymand, A., Zounemat-Kermani, M., 2022. Homonuclear molecules optimization (HMO) meta-heuristic algorithm. Knowl.-Based Syst. 258, 110032. http://dx.doi.org/10.1016/j.knosys.2022.110032.
Miikkulainen, R., Forrest, S., 2021. A biological perspective on evolutionary computation. Nat. Mach. Intell. 3 (1), 9–15. http://dx.doi.org/10.1038/s42256-020-00278-8.
Mirjalili, S., 2015. Moth-flame optimization algorithm: A novel nature-inspired heuristic paradigm. Knowl.-Based Syst. 89, 228–249. http://dx.doi.org/10.1016/j.knosys.2015.07.006.
Mirjalili, S., 2016. SCA: A sine cosine algorithm for solving optimization problems. Knowl.-Based Syst. 96, 120–133. http://dx.doi.org/10.1016/j.knosys.2015.12.022.
Mirjalili, S., Gandomi, A.H., Mirjalili, S.Z., Saremi, S., Faris, H., Mirjalili, S.M., 2017. Salp swarm algorithm: A bio-inspired optimizer for engineering design problems. Adv. Eng. Softw. 114, 163–191. http://dx.doi.org/10.1016/j.advengsoft.2017.07.002.
Mirjalili, S., Lewis, A., 2016. The whale optimization algorithm. Adv. Eng. Softw. 95, 51–67. http://dx.doi.org/10.1016/j.advengsoft.2016.01.008.
Mirjalili, S., Mirjalili, S.M., Hatamlou, A., 2016. Multi-verse optimizer: A nature-inspired algorithm for global optimization. Neural Comput. Appl. 27, 495–513. http://dx.doi.org/10.1007/s00521-015-1870-7.
Mirjalili, S., Mirjalili, S.M., Lewis, A., 2014. Grey wolf optimizer. Adv. Eng. Softw. 69, 46–61. http://dx.doi.org/10.1016/j.advengsoft.2013.12.007.
Mirrashid, M., Naderpour, H., 2023. Incomprehensible but intelligible-in-time logics: Theory and optimization algorithm. Knowl.-Based Syst. 110305. http://dx.doi.org/10.1016/j.knosys.2023.110305.
Mohamed, A.W., Hadi, A.A., Agrawal, P., Sallam, K.M., Mohamed, A.K., 2021. Gaining-sharing knowledge based algorithm with adaptive parameters hybrid with IMODE algorithm for solving CEC 2021 benchmark problems. In: 2021 IEEE Congress on Evolutionary Computation. CEC, IEEE, pp. 841–848. http://dx.doi.org/10.1109/cec45853.2021.9504814.
Mohamed, A.W., Hadi, A.A., Mohamed, A.K., 2020. Gaining-sharing knowledge based algorithm for solving optimization problems: A novel nature-inspired algorithm. Int. J. Mach. Learn. Cybern. 11 (7), 1501–1529. http://dx.doi.org/10.1007/s13042-019-01053-x.
Mohamed, A.W., Mohamed, A.K., 2019. Adaptive guided differential evolution algorithm with novel mutation for numerical optimization. Int. J. Mach. Learn. Cybern. 10, 253–277. http://dx.doi.org/10.1007/s13042-017-0711-7.
Mohammadi-Balani, A., Nayeri, M.D., Azar, A., Taghizadeh-Yazdi, M., 2021. Golden eagle optimizer: A nature-inspired metaheuristic algorithm. Comput. Ind. Eng. 152, 107050. http://dx.doi.org/10.1016/j.cie.2020.107050.
Morales-Castañeda, B., Zaldivar, D., Cuevas, E., Fausto, F., Rodríguez, A., 2020. A better balance in metaheuristic algorithms: Does it exist? Swarm Evol. Comput. 54, 100671. http://dx.doi.org/10.1016/j.swevo.2020.100671.
Moscato, P., et al., 1989. On evolution, search, optimization, genetic algorithms and martial arts: Towards memetic algorithms. Caltech Concurr. Comput. Program, C3P Rep. 826 (1989), 37, https://oa.mg/work/18216792.
Moyya, S., Thejasree, P., Abraham, B.C., Mangalathu, G.S., 2023. Design and analysis of single and multi-layer pressure vessel. Mater. Today Proc. http://dx.doi.org/10.1016/j.matpr.2023.06.393.
Naruei, I., Keynia, F., 2022. Wild horse optimizer: A new meta-heuristic algorithm for solving engineering optimization problems. Eng. Comput. 38 (Suppl 4), 3025–3056. http://dx.doi.org/10.1007/s00366-021-01438-z.
Neshat, M., Sepidnam, G., Sargolzaei, M., Toosi, A.N., 2014. Artificial fish swarm algorithm: A survey of the state-of-the-art, hybridization, combinatorial and indicative applications. Artif. Intell. Rev. 42 (4), 965–997. http://dx.doi.org/10.1007/s10462-012-9342-2.
Niu, P., Niu, S., Chang, L., et al., 2019. The defect of the grey wolf optimization algorithm and its verification method. Knowl.-Based Syst. 171, 37–43. http://dx.doi.org/10.1016/j.knosys.2019.01.018.
Noel, M.M., Muthiah-Nakarajan, V., Amali, G.B., Trivedi, A.S., 2021. A new biologically inspired global optimization algorithm based on firebug reproductive swarming behaviour. Expert Syst. Appl. 183, 115408. http://dx.doi.org/10.1016/j.eswa.2021.115408.
Oyelade, O.N., Ezugwu, A.E.-S., Mohamed, T.I., Abualigah, L., 2022. Ebola optimization search algorithm: A new nature-inspired metaheuristic optimization algorithm. IEEE Access 10, 16150–16177. http://dx.doi.org/10.1109/access.2022.3147821.
Öztürk, C., Aslan, S., 2016. A new artificial bee colony algorithm to solve the multiple sequence alignment problem. Int. J. Data Min. Bioinform. 14 (4), 332–353. http://dx.doi.org/10.1504/ijdmb.2016.075823.
Peraza-Vázquez, H., Peña-Delgado, A.F., Echavarría-Castillo, G., Morales-Cepeda, A.B., Velasco-Álvarez, J., Ruiz-Perez, F., 2021. A bio-inspired method for engineering design optimization inspired by dingoes hunting strategies. Math. Probl. Eng. 2021, 1–19. http://dx.doi.org/10.1155/2021/9107547.
Pérez, C., Climent, L., Nicoló, G., Arbelaez, A., Salido, M.A., 2023. A hybrid metaheuristic with learning for a real supply chain scheduling problem. Eng. Appl. Artif. Intell. 126, 107188. http://dx.doi.org/10.1016/j.engappai.2023.107188.
Pickard, J.K., Carretero, J.A., Bhavsar, V.C., 2016. On the convergence and origin bias of the teaching-learning-based-optimization algorithm. Appl. Soft Comput. 46, 115–127. http://dx.doi.org/10.1016/j.asoc.2016.04.029.
Pierezan, J., Coelho, L.D.S., 2018. Coyote optimization algorithm: A new metaheuristic for global optimization problems. In: 2018 IEEE Congress on Evolutionary Computation. CEC, IEEE, pp. 1–8. http://dx.doi.org/10.1109/cec.2018.8477769.
Piotrowski, A.P., Napiorkowski, J.J., 2018. Some metaheuristics should be simplified. Inform. Sci. 427, 32–62. http://dx.doi.org/10.1016/j.ins.2017.10.039.
Połap, D., Woźniak, M., 2021. Red fox optimization algorithm. Expert Syst. Appl. 166, 114107. http://dx.doi.org/10.1016/j.eswa.2020.114107.
Qing, A., 2008. A study on base vector for differential evolution. In: 2008 IEEE Congress on Evolutionary Computation (IEEE World Congress on Computational Intelligence). IEEE, pp. 550–556. http://dx.doi.org/10.1109/CEC.2008.4630850.

Rajwar, K., Deep, K., Das, S., 2023. An exhaustive review of the metaheuristic algorithms for search and optimization: Taxonomy, applications, and open challenges. Artif. Intell. Rev. 1–71. http://dx.doi.org/10.1007/s10462-023-10470-y.

Rao, R., 2016. Jaya: A simple and new optimization algorithm for solving constrained and unconstrained optimization problems. Int. J. Ind. Eng. Comput. 7 (1), 19–34. http://dx.doi.org/10.5267/j.ijiec.2015.8.004.

Rao, R., 2020. Rao algorithms: Three metaphor-less simple algorithms for solving optimization problems. Int. J. Ind. Eng. Comput. 11 (1), 107–130. http://dx.doi.org/10.5267/j.ijiec.2019.6.002.

Rao, R.V., Savsani, V.J., Vakharia, D., 2011. Teaching–learning-based optimization: A novel method for constrained mechanical design optimization problems. Comput. Aided Des. 43 (3), 303–315. http://dx.doi.org/10.1016/j.cad.2010.12.015.

Rashedi, E., Nezamabadi-Pour, H., Saryazdi, S., 2009. GSA: A gravitational search algorithm. Inform. Sci. 179 (13), 2232–2248. http://dx.doi.org/10.1016/j.ins.2009.03.004.

Rechenberg, I., 1965. Cybernetic solution path of an experimental problem. Roy. Aircr. Establ., Libr. transl. 1122, https://aitopics.org/doc/classics:30AE00D2/.

Reynolds, R.G., 1994. An introduction to cultural algorithms. In: Proceedings of the 3rd Annual Conference on Evolutionary Programming, World Scientific Publishing. World Scientific, pp. 131–139. http://dx.doi.org/10.1142/9789814534116.

Rudolph, G., 1996. Convergence of evolutionary algorithms in general search spaces. In: Proceedings of IEEE International Conference on Evolutionary Computation. IEEE, pp. 50–54. http://dx.doi.org/10.1109/ICEC.1996.542332.

Salimi, H., 2015. Stochastic fractal search: A powerful metaheuristic algorithm. Knowl.-Based Syst. 75, 1–18. http://dx.doi.org/10.1016/j.knosys.2014.07.025.

Sallam, K.M., Elsayed, S.M., Chakrabortty, R.K., Ryan, M.J., 2020. Improved multi-operator differential evolution algorithm for solving unconstrained problems. In: 2020 IEEE Congress on Evolutionary Computation. CEC, IEEE, pp. 1–8. http://dx.doi.org/10.1109/cec48606.2020.9185577.

Sameh, S.M., Moustafa, H.E.-D., AbdelHay, E.H., Ata, M.M., 2024. An effective chaotic maps image encryption based on metaheuristic optimizers. J. Supercomput. 80 (1), 141–201. http://dx.doi.org/10.1007/s11227-023-05413-x.

Sampson, J.R., 1976. Adaptation in Natural and Artificial Systems. Society for Industrial and Applied Mathematics, http://dx.doi.org/10.7551/mitpress/1090.001.0001.

Sayood, K., Otu, H.H., 2023. Multiple sequence alignment. In: Bioinformatics: A One Semester Course. Springer, pp. 85–101. http://dx.doi.org/10.1007/978-3-031-20017-5_5.

Shabani, A., Asgarian, B., Salido, M., Gharebaghi, S.A., 2020. Search and rescue optimization algorithm: A new optimization method for solving constrained engineering optimization problems. Expert Syst. Appl. 161, 113698. http://dx.doi.org/10.1016/j.eswa.2020.113698.

Shareef, H., Ibrahim, A.A., Mutlag, A.H., 2015. Lightning search algorithm. Appl. Soft Comput. 36, 315–333. http://dx.doi.org/10.1016/j.asoc.2015.07.028.

Shi, Y., 2011. Brain storm optimization algorithm. In: Advances in Swarm Intelligence: Second International Conference, ICSI 2011, Chongqing, China, June 12–15, 2011, Proceedings, Part I 2. Springer, pp. 303–309. http://dx.doi.org/10.1007/978-3-642-21515-5_36.

Simon, D., 2008. Biogeography-based optimization. IEEE Trans. Evol. Comput. 12 (6), 702–713. http://dx.doi.org/10.1109/tevc.2008.919004.

Singh, R., Gaurav, K., Pathak, V.K., Singh, P., Chaudhary, H., 2020. Best-worst-play (BWP): A metaphor-less optimization algorithm. J. Phys. Conf. Ser. 1455 (1), 012007. http://dx.doi.org/10.1088/1742-6596/1455/1/012007.

Singh, A., Kumar, A., 2021. Applications of nature-inspired meta-heuristic algorithms: A survey. Int. J. Adv. Intell. Paradigms 20 (3–4), 388–417. http://dx.doi.org/10.1504/ijaip.2021.119026.

Slowik, A., Kwasnicka, H., 2017. Nature inspired methods and their industry applications—Swarm intelligence algorithms. IEEE Trans. Ind. Inform. 14 (3), 1004–1015. http://dx.doi.org/10.1109/tii.2017.2786782.

Solis, F.J., Wets, R.J.-B., 1981. Minimization by random search techniques. Math. Oper. Res. 6 (1), 19–30. http://dx.doi.org/10.1287/moor.6.1.19.

Sörensen, K., 2015. Metaheuristics—the metaphor exposed. Int. Trans. Oper. Res. 22 (1), 3–18. http://dx.doi.org/10.1111/itor.12001.

Storn, R., Price, K., 1997. Differential evolution-a simple and efficient heuristic for global optimization over continuous spaces. J. Global Optim. 11 (4), 341. http://dx.doi.org/10.1023/A:1008202821328.

Su, H., Zhao, D., Heidari, A.A., Liu, L., Zhang, X., Mafarja, M., Chen, H., 2023. RIME: A physics-based optimization. Neurocomputing 532, 183–214. http://dx.doi.org/10.1016/j.neucom.2023.02.010.

Sun, J., Wu, X., Fang, W., Ding, Y., Long, H., Xu, W., 2012. Multiple sequence alignment using the hidden Markov model trained by an improved quantum-behaved particle swarm optimization. Inform. Sci. 182 (1), 93–114. http://dx.doi.org/10.1016/j.ins.2010.11.014.

Talatahari, S., Azizi, M., 2021. Chaos game optimization: A novel metaheuristic algorithm. Artif. Intell. Rev. 54, 917–1004. http://dx.doi.org/10.1007/s10462-020-09867-w.

Tzanetos, A., Blondin, M., 2023. A qualitative systematic review of metaheuristics applied to tension/compression spring design problem: Current situation, recommendations, and research direction. Eng. Appl. Artif. Intell. 118, 105521. http://dx.doi.org/10.1016/j.engappai.2022.105521.

Velasco, L., Guerrero, H., Hospitaler, A., 2024. A literature review and critical analysis of metaheuristics recently developed. Arch. Comput. Methods Eng. 31 (1), 125–146. http://dx.doi.org/10.1007/s11831-023-09975-0.

Vermetten, D., Doerr, C., Wang, H., Kononova, A.V., Bäck, T., 2024. Large-scale benchmarking of metaphor-based optimization heuristics. http://dx.doi.org/10.48550/arXiv.2402.09800, arXiv preprint arXiv:2402.09800.

Vermetten, D., van Stein, B., Caraffini, F., Minku, L.L., Kononova, A.V., 2022. Bias: A toolbox for benchmarking structural bias in the continuous domain. IEEE Trans. Evol. Comput. 26 (6), 1380–1393. http://dx.doi.org/10.1109/TEVC.2022.3189848.

Veysari, E.F., et al., 2022. A new optimization algorithm inspired by the quest for the evolution of human society: Human felicity algorithm. Expert Syst. Appl. 193, 116468. http://dx.doi.org/10.1016/j.eswa.2021.116468.

Wang, L., Cao, Q., Zhang, Z., Mirjalili, S., Zhao, W., 2022. Artificial rabbits optimization: A new bio-inspired meta-heuristic algorithm for solving engineering optimization problems. Eng. Appl. Artif. Intell. 114, 105082. http://dx.doi.org/10.1016/j.engappai.2022.105082.

Wang, K., Wang, Y., Tao, S., Cai, Z., Lei, Z., Gao, S., 2023a. Spherical search algorithm with adaptive population control for global continuous optimization problems. Appl. Soft Comput. 132, 109845. http://dx.doi.org/10.1016/j.asoc.2022.109845.

Wang, X., Xu, J., Huang, C., 2023b. Fans optimizer: A human-inspired optimizer for mechanical design problems optimization. Expert Syst. Appl. 228, 120242. http://dx.doi.org/10.1016/j.eswa.2023.120242.

Wilcoxon, F., 1992. Individual Comparisons by Ranking Methods. Springer, http://dx.doi.org/10.2307/3001968.

Wu, G., Mallipeddi, R., Suganthan, P.N., 2017. Problem Definitions and Evaluation Criteria for the CEC 2017 Competition and Special Session on Constrained Single Objective Real-Parameter Optimization. Computational Intelligence Laboratory, Zhengzhou University, Zhengzhou China and Technical Report, Nanyang Technological University, Singapore, https://github.com/P-N-Suganthan.

Xu, P., Luo, W., Xu, J., Qiao, Y., Zhang, J., Gu, N., 2021. An alternative way of evolutionary multimodal optimization: Density-based population initialization strategy. Swarm Evol. Comput. 67, 100971. http://dx.doi.org/10.1016/j.swevo.2021.100971.

Xue, J., Shen, B., 2023. Dung beetle optimizer: A new meta-heuristic algorithm for global optimization. J. Supercomput. 79 (7), 7305–7336. http://dx.doi.org/10.1007/s11227-022-04959-6.

Yadav, A., et al., 2019. AEFA: Artificial electric field algorithm for global optimization. Swarm Evol. Comput. 48, 93–108. http://dx.doi.org/10.1016/j.swevo.2019.03.013.

Yang, X.-S., 2009. Firefly algorithms for multimodal optimization. In: Stochastic Algorithms: Foundations and Applications: 5th International Symposium, SAGA 2009, Sapporo, Japan, October 26–28, 2009. Proceedings 5. Springer, pp. 169–178. http://dx.doi.org/10.1007/978-3-642-04944-6_14.

Yang, X.-S., 2012. Flower pollination algorithm for global optimization. In: Unconventional Computation and Natural Computation: 11th International Conference, UCNC 2012, OrlÉan, France, September 3–7, 2012. Proceedings 11. Springer, pp. 240–249. http://dx.doi.org/10.1007/978-3-642-32894-7_27.

Yang, Y., Chen, H., Heidari, A.A., Gandomi, A.H., 2021a. Hunger games search: Visions, conception, implementation, deep analysis, perspectives, and towards performance shifts. Expert Syst. Appl. 177, 114864. http://dx.doi.org/10.1016/j.eswa.2021.114864.

Yang, X.-S., Deb, S., 2009. Cuckoo search via Lévy flights. In: 2009 World Congress on Nature & Biologically Inspired Computing. NaBIC, IEEE, pp. 210–214. http://dx.doi.org/10.1109/NABIC.2009.5393690.

Yang, X.-S., Deb, S., Fong, S., 2014. Metaheuristic algorithms: Optimal balance of intensification and diversification. Appl. Math. Inf. Sci. 8 (3), 977. http://dx.doi.org/10.12785/amis/080306.

Yang, Z., Deng, L., Wang, Y., Liu, J., 2021b. Aptenodytes forsteri optimization: Algorithm and applications. Knowl.-Based Syst. 232, 107483. http://dx.doi.org/10.1016/j.knosys.2021.107483.

Yang, H., Yu, Y., Cheng, J., Lei, Z., Cai, Z., Zhang, Z., Gao, S., 2022. An intelligent metaphor-free spatial information sampling algorithm for balancing exploitation and exploration. Knowl.-Based Syst. 250, 109081. http://dx.doi.org/10.1016/j.knosys.2022.109081.

Yang, Q., et al., 2024. Triple competitive differential evolution for global numerical optimization. Swarm Evol. Comput. 84, 101450. http://dx.doi.org/10.1016/j.swevo.2023.101450.

Yüksel, N., Börklü, H.R., Sezer, H.K., Canyurt, O.E., 2023. Review of artificial intelligence applications in engineering design perspective. Eng. Appl. Artif. Intell. 118, 105697. http://dx.doi.org/10.1016/j.engappai.2022.105697.

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. http://dx.doi.org/10.1016/j.knosys.2022.110206.

Zhang, J., Sanderson, A.C., 2009. JADE: Adaptive differential evolution with optional external archive. IEEE Trans. Evol. Comput. 13 (5), 945–958. http://dx.doi.org/10.1109/TEVC.2009.2014613.

Zhang, X., Wang, X., Kang, Q., Cheng, J., 2019. Differential mutation and novel social learning particle swarm optimization algorithm. Inform. Sci. 480, 109–129. http://dx.doi.org/10.1016/j.ins.2018.12.030.

Zhang, Y., Zhang, Q., Zhou, J., Zou, Q., 2022. A survey on the algorithm and development of multiple sequence alignment. Brief. Bioinform. 23 (3), bbac069. http://dx.doi.org/10.1093/bib/bbac069.

Zhao, W., Wang, L., Mirjalili, S., 2022. Artificial hummingbird algorithm: A new bio-inspired optimizer with its engineering applications. Comput. Methods Appl. Mech. Engrg. 388, 114194. http://dx.doi.org/10.1016/j.cma.2021.114194.

Zhao, W., Wang, L., Zhang, Z., 2019a. Atom search optimization and its application to solve a hydrogeologic parameter estimation problem. Knowl.-Based Syst. 163, 283–304. http://dx.doi.org/10.1016/j.knosys.2018.08.030.

Zhao, W., Wang, L., Zhang, Z., 2019b. Supply-demand-based optimization: A novel economics-inspired algorithm for global optimization. IEEE Access 7, 73182–73206. http://dx.doi.org/10.1109/access.2019.2918753.

Zhao, W., Wang, L., Zhang, Z., 2020a. Artificial ecosystem-based optimization: A novel nature-inspired meta-heuristic algorithm. Neural Comput. Appl. 32, 9383–9425. http://dx.doi.org/10.1007/s00521-019-04452-x.

Zhao, W., Zhang, Z., Wang, L., 2020b. Manta ray foraging optimization: An effective bio-inspired optimizer for engineering applications. Eng. Appl. Artif. Intell. 87, 103300. http://dx.doi.org/10.1016/j.engappai.2019.103300.

Zhong, C., Li, G., Meng, Z., 2022. Beluga whale optimization: A novel nature-inspired metaheuristic algorithm. Knowl.-Based Syst. 251, 109215. http://dx.doi.org/10.1016/j.knosys.2022.109215.