Expert Systems With Applications 235 (2024) 121218

Contents lists available at ScienceDirect

Expert Systems With Applications

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

Quadruple parameter adaptation growth optimizer with integrated distribution, confrontation, and balance features for optimization ☆

Hao Gao, Qingke Zhang *, Xianglong Bu, Huaxiang Zhang

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

ARTICLE INFO

Keywords:
Optimization algorithm
Quadruple parameter adaptation
Jensen–Shannon divergence
Operator refinement
Image segmentation
Sensor deployment

ABSTRACT

Growth optimizer is a novel metaheuristic algorithm that has powerful numerical optimization capabilities. However, its parameters and search operators become crucial factors that significantly impact its optimization capability for engineering problems and benchmarks. Therefore, this paper proposes a quadruple parameter adaptation growth optimizer (QAGO) integrated with distribution, confrontation, and balance features. In QAGO, the quadruple parameter adaptation mechanism aims to reduce the algorithmic sensitivity for parameter setting and enhance the algorithmic adaptability. By employing parameter sampling that adheres to specific probability distributions, the parameter adaptation mechanism achieves dynamic tuning of the algorithm hyperparameters. Moreover, one-dimensional mapping and fitness difference methods are designed in the triple parameter self-adaptation mechanism based on the contradictory relationship to adjust the operator's parameters. After that, "spear" and "shield" are balanced based on the Jensen–Shannon divergence in information theory. Furthermore, the topological structure of the operators is redesigned, and by combining the parameter adaptation mechanism, operator refinement is achieved. Refined operators can effectively utilize different evolutionary information to improve the quality of the solution. The experiment evaluates the performance of QAGO on distinct optimization problems on the CEC 2017 and CEC 2022 test suites. To demonstrate the capability of QAGO in solving real-world applications, it was applied to tackle two specific problems: multilevel threshold image segmentation and wireless sensor network node deployment. The results demonstrated that QAGO delivers highly promising optimization results compared to seventy-one high-performance competing algorithms, including the five IEEE CEC competition winners. The source code of the QAGO algorithm is publicly available at https://github.com/tsingke/QAGO.

1. Introduction

As real-world application problems become increasingly complex, exact algorithms may encounter bottlenecks when solving optimization problems with multimodal, discontinuous, noisy, and other complex features (Eiben & Smith, 2015; Wu, Mallipeddi, & Suganthan, 2019). However, metaheuristic algorithms provide an effective paradigm for solving these problems (Dokeroglu, Sevinc, Kucukyilmaz, & Cosar, 2019; Ma, Wu, Suganthan, Song, & Luo, 2023). These problem-independent advanced techniques can be applied to a wide range of problems, showcasing their versatility and effectiveness (Ezugwu et al., 2021). They have been extensively studied up to the present time.

The classical metaheuristics such as differential evolution (DE) (Storn & Price, 1997), particle swarm optimization (PSO) (Kennedy & Eberhart, 1995), genetic algorithm (GA) (Holland, 1992; Zames, 1981), and ant colony optimization (ACO) (Dorigo, Birattari, & Stutzle, 2006), and novel metaheuristics such as fans optimizer (FO) (Wang, Xu, & Huang, 2023), poplar optimization algorithm (POA) (Chen et al., 2022), meerkat optimization algorithm (MOA) (Xian & Feng, 2023), human felicity algorithm (HFA) (Veysari et al., 2022), reptile search algorithm (RSA) (Abualigah, Abd Elaziz, Sumari, Geem, & Gandomi, 2022), search and rescue optimization algorithm (SAR) (Shabani, Asgarian, Salido, & Gharebaghi, 2020), weighted mean of vectors (INFO) (Ahmadianfar, Heidari, Noshadian, Chen, & Gandomi, 2022), remora optimization algorithm (ROA) (Jia, Peng, & Lang, 2021), equilibrium optimizer (EO) (Faramarzi, Heidarinejad, Stephens, & Mirjalili, 2020), kepler optimization algorithm (KOA) (Abdel-Basset, Mohamed, Azeem, Jameel, & Abouhawwash, 2023), nutcracker optimization algorithm (NOA) (Abdel-Basset, Mohamed, Jameel, & Abouhawwash,

☆ This work is supported by the National Natural Science Foundation of China (No. 62006144), the Major Fundamental Research Project of Shandong, China (No. ZR2019ZD03), and the Taishan Scholar Project of Shandong, China (No. ts20190924).

* Corresponding author.
E-mail addresses: GaooHaoo@hotmail.com (H. Gao), tsingke@sdnu.edu.cn (Q. Zhang), bullong@foxmail.com (X. Bu), huaxzhang@sdnu.edu.cn (H. Zhang).

https://doi.org/10.1016/j.eswa.2023.121218
Received 22 June 2023; Received in revised form 27 July 2023; Accepted 14 August 2023
Available online 19 August 2023

2023), and growth optimizer (GO) (Zhang, Gao, Zhan, Li, & Zhang, 2023). They do not rely on gradient information about the problem. Instead, they use an intelligent stochastic search process and an iterative evolutionary pattern of the population, which is a key factor in their success at solving black-box optimization problems (Das & Suganthan, 2011; Eiben & Smith, 2015). Currently, they involve many fields, such as engineering design problems (Balochian & Baloochian, 2019), feature selection problems (Houssein, Oliva, Çelik, Emam, & Ghoniem, 2023), flow shop scheduling problems (Du, Zhou, Wu, & Fei, 2023), image segmentation problems (Chen et al., 2022), power flow study problem (Dash, Subhashini, & Chinta, 2022), environmental economic dispatch (Xin-gang, Ji, Jin, & Ying, 2020), and prediction problems (Ding et al., 2023; Zhong, You, & Cheng, 2023).

Variants are improved versions of basic algorithms aimed at enhancing their search performance in different optimization environments. Some variants such as improved arithmetic optimization algorithm (IAOA) (Liu, Zhang, Zhang, Li, & Chen, 2023), heuristic crossing search and rescue optimization algorithm (HC-SAR) (Zhang, Zhou, Qin, & Tang, 2023), multi-strategy snake optimizer (MSO) (Wang, Jiao, Li, & Zhang, 2023), improved seagull optimization algorithm (ISOA) (Abdelhamid, Houssein, Mahdy, Selim, & Kamel, 2022), co-evolutionary migrating birds optimization algorithm based on online learning policy gradient (CMBO-PG) (Zhao, Jiang, Xu, Zhu, et al., 2023), modified adaptive ant colony optimization algorithm (MAACO) (Wu, Huang, Cui, Liu, & Xiao, 2023), hybrid optimization algorithm based on DE and harris hawks optimization (DEHHO) (Zhong et al., 2023), and improved corona-virus herd immunity optimizer algorithm (ICHIOA) (Naderipour, Abdullah, Marzbali, & Nowdeh, 2022). Whether they are metaheuristic algorithms or variants, their primary objective is to find the global solution to an optimization problem. However, effectively exploring the global solution to highly complex optimization problems is often a challenging or even impossible task. In this situation, the criterion for success is to find a solution that is closest to the global optimum (Aras, Gedikli, & Kahraman, 2021). To achieve this task, algorithms need to strike a balance between exploration and exploitation. However, this is no easy task, as these two aspects are inherently contradictory. If there is excessive exploration, it may result in an inability to obtain higher-quality solutions. In contrast, if there is excessive exploitation, there is a higher chance of overlooking the region where the optimal solution lies (Del Ser et al., 2019). Hence, it is crucial to employ effective and stable methods to strike a balance between exploration and exploitation. Growth optimizer (GO) is a powerful and effective approach that has demonstrated its efficacy in achieving this objective through numerical optimization and real-world application optimizations (Zhang, Gao, et al., 2023).

However, this capability is affected by critical factors such as parameters and search operators. The fixed hyperparameter settings of GO can only be adapted to one class of problems, and when GO is applied to another, its performance may be offset by improper parameter settings. Moreover, GO's operator structure can limit the diversity of its search patterns, which in turn limits its performance on different problems. Therefore, this research will analyze and address the GO's challenge from two perspectives: parameter tuning and operator refinement.

To alleviate the parameter sensitivity problem, there are numerous advanced techniques for parameter dynamic tuning. A new adaptive parameter control method is proposed in Sun, Liu, Bäck, and Xu (2021). In this method, parameter control is modeled as a finite time-domain Markov decision process. Apply a reinforcement learning algorithm called strategy gradient to learn agents, which can adaptively provide control parameters during the search process. The literature (Awad, Ali, & Suganthan, 2017) proposes non-adaptive sine decreasing adjustment and adaptive sine increasing adjustment methods to tune the parameters in the algorithm. The literature (Tanabe & Fukunaga, 2013) uses the historical memory of successful control parameter settings to guide the selection of future control parameter values. In the literature (Wang et al., 2022), the parameter tuning process is achieved by sampling parameters from a predefined Gaussian distribution. The literature (Zhou, Ma, Gu, Chen, & Deng, 2022) designed a dynamic parameter adjustment mechanism using particle swarm optimization and fuzzy systems to adaptively adjust relevant parameters. In the literature (Liu et al., 2018), adaptation parameters are derived based on previous successful experiences of population transformation and heuristic information related to the individual's current state. Additionally, parameter dynamic tuning in the covariance matrix adaptation evolution strategy (CMA-ES) is widely regarded as the most advanced technique for parameter self-adaptation (Hansen, 2006). More extensive information on parameter adaptation is described in detail in the literature (de Lacerda, de Araujo Pessoa, de Lima Neto, Ludermir, & Kuchen, 2021; Huang, Li, & Yao, 2020). However, it is an undeniable fact that appropriate parameter dynamic tuning methods can maximize the performance of the algorithm. Although many adaptive methods can alleviate the algorithm's sensitivity to parameters, this may sacrifice computational complexity, such as using a covariance matrix to record historical evolution path information to achieve parameter self-adaptation (Hansen, 2016). Given the limitations of fixed parameter settings and the lack of flexibility in random sampling methods, the development of efficient parameter adaptation methods remains a highly active research topic.

In addition to parameter adjustments, the operator serves as the core search component of the algorithm. Well-known search operators have shown excellent optimization results in numerous optimization problems (Ahmad, Isa, Lim, & Ang, 2022; Opara & Arabas, 2019). However, how to use different population evolution information to construct efficient search operators is still a problem worth studying.

As previously discussed, GO is encountering challenges with parameter adjustment and operator structure. To showcase the motive driving this work, we have distilled it into the following bullet points for enhanced clarity:

(1) According to the no free lunch (NFL) theorem (Wolpert & Macready, 1997), there is no universal algorithm that can effectively solve all problems, which has prompted the design of problem-driven optimization algorithms like QAGO to solve specific problems.
(2) The related flaws of GO have been discussed. The sensitivity of parameters and limitations of operator structure are not exclusive to GO but rather common flaws found in most metaheuristic algorithms. Addressing such similar deficiencies can provide universal design inspiration for algorithmic improvements.
(3) Owing to the fact that GO has just been proposed, there are no other GO-based improvement methods to draw inspiration from. Therefore, this becomes a challenge and has inspired our motivation for improvement.
(4) Considering the challenges posed by engineering problems, adaptive methods have the ability to adapt to the diverse characteristics of the problems. Therefore, designing an adaptive approach ensures the effectiveness and practicality of the algorithm in addressing real-world problems.

As mentioned above, this work proposes a quadruple parameter adaptation growth optimizer integrated with distribution, confrontation, and balance features (QAGO) to solve numerical optimization and real-world applications. The main contributions are summarized as follows:

(1) A novel optimization algorithm, named QAGO, has been proposed that primarily focuses on parameter tuning and operator refinement.
(2) A parameter adaptation mechanism based on distribution has been developed to dynamically adjust the algorithmic hyperparameters.

(3) Triple parameter self-adaptation mechanisms have been designed to achieve high adaptability in the operator search process. They are one-dimensional mapping of vectors, fitness difference, and Jensen–Shannon divergence methods. The first two methods confront each other and achieve a stable and effective parameter self-adaptation process by balancing through the third method.
(4) The operators' topological structure has been redesigned, and the parameter adaptation tuning mechanism has been employed to refine the operators.
(5) The numerical optimization performance of QAGO was evaluated under different problem dimensions using the challenging CEC 2017 and CEC 2022 test suites. Ten novel metaheuristic algorithms, fifty classical metaheuristic algorithms, six self-adaptive algorithms, and five winners from the CEC competitions participated in the evaluation process. Various statistical methods were used to validate the experimental results.
(6) In two challenging real-world problems, namely multilevel threshold image segmentation problem and node deployment problem on wireless sensor networks, QAGO has exhibited competitiveness when compared to GO and CEC competition winners.

The rest of this paper is organized as follows. Section 2 introduces the standard GO. Section 3 presents the QAGO algorithm proposed in this paper. The results of the numerical optimization experiment are given and discussed in Section 4. Sections 5 and 6 show the applications of QAGO. The discussion in this work was presented in Section 7. Section 8 gives the conclusion of this paper.

2. Growth optimizer

This section provides a detailed introduction to the standard GO algorithm. Its inspiration is derived from human learning and reflection behaviors.

2.1. Initialization

GO is a population-based metaheuristic algorithm that uses a uniform distribution to randomly generate individuals in the search space. Eq. (1) gives a description in detail:

$$X_{i,j} = lb + (ub - lb) \times \mathrm{rand}(0,1) \tag{1}$$

where i is the individual index, j is the dimension index of individual X_i, and ub and lb are the upper and lower bounds of the search space, respectively. Additionally, $\mathrm{rand}(0,1)$ generates a random number that obeys a uniform distribution within the range of $[0,1]$.

2.2. Population division

The standard GO divides the population into three levels by using hyperparameter P_1: the top level, the middle level, and the bottom level. The top level contains the best individual X_{best} and the better individuals X_{better}. The bottom level contains the worse individuals X_{worse}. The remaining normal individuals X_{normal} are included in the middle level. All individuals are sorted from best to worst according to their fitness. The best individual is given a ranking of 1, while the other better individuals are ranked from 2 to P_1. The worse individuals are ranked from $N - P_1 + 1$ to N, while the normal individuals are ranked from $P_1 + 1$ to $N - P_1$. The algorithm will sample individuals from a specific level to complete subsequent calculations. Note that the integer P_1 is set to 5 in standard GO.

2.3. Learning phase

The learning phase of GO simulates the process by which an individual learns and grows from the gaps between different individuals. In this phase, individual X_i will learn from the gaps and benefit from them to grow. These gaps include four groups: Gap_1 represents the gap between the best individual and the better individual, Gap_2 represents the gap between the best individual and the worse individual, Gap_3 represents the gap between the better individual and the worse individual, and Gap_4 represents the gap between two random individuals that are different from individual X_i. Eq. (2) describes these gaps in the following manner:

$$\begin{cases} Gap_1 = X_{best} - X_{better} \\ Gap_2 = X_{best} - X_{worse} \\ Gap_3 = X_{better} - X_{worse} \\ Gap_4 = X_{r_1} - X_{r_2} \end{cases} \tag{2}$$

where X_{best}, X_{better}, X_{worse}, X_{r_1}, and X_{r_2} are the best individual, the better individual, the worse individual, and two random individuals, selected by individual X_i, respectively.

For the kth gap (Gap_k), individual X_i transforms it into the kth group of knowledge acquisition (KA_k) and undergoes an accumulation and evolutionary process. For this process, GO uses the concepts of self-perception factor (SF) and learning factor (LF) to control KA, and this process will be described by Eq. (3) as follows:

$$KA_k = SF_i \cdot LF_k \cdot Gap_k, (k = 1,2,3,4) \tag{3}$$

where SF_i represents the ith individual's assessment of its own state and LF_k represents the individual's assessment of the external state. During population evolution, these two control parameters are adjusted adaptively by Eqs. (4) and (5):

$$SF_i = \frac{f(X_i)}{f(X_{worst})} \tag{4}$$

where $f(X_i)$ is the objective function value of the ith individual and $f(X_{worst})$ is the objective function value of the worst individual in the population.

$$LF_k = \frac{\|Gap_k\|}{\sum_{k=1}^{4} \|Gap_k\|}, \ (k = 1,2,3,4) \tag{5}$$

where $\|Gap_k\|$ is the Euclidean distance of the kth group of gaps, and LF_k is a normalized result.

By absorbing the gaps between different individuals, the ith individual completes the learning process, which will be described by Eq. (6) as follows:

$$X_i^{t+1} = X_i^t + \sum_{k=1}^{4} KA_k \tag{6}$$

2.4. Reflection phase

The reflection phase of GO simulates the individual's reflection process. In this phase, individual X_i further grows through three different reflection operations. Its detailed process is described through Eq. (7) as follows:

$$X_{i,j}^{t+1} = \begin{cases} \begin{cases} lb + \mathrm{rand}(0,1) \times (ub - lb), & if\ \mathrm{rand}(0,1) < AF \\ X_{i,j}^t + \mathrm{rand}(0,1) \times \left(R_j - X_{i,j}^t\right), & else \end{cases} & if\ \mathrm{rand}(0,1) < P_3 \\ X_{i,j}^t, & else \end{cases} \tag{7}$$

where R is one of the better P_1 random individuals in the population. Hyperparameter P_3 controls the probability of performing different operations. Note that P_3 is set to 0.3 in standard GO. The attenuation

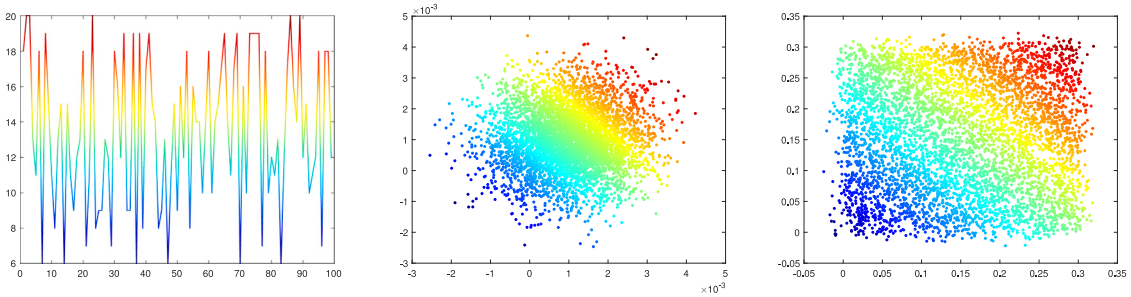


Fig. 1. Sampling parameters P_1, P_2, and P_3 in the QAGO algorithm.

factor (AF) controls the restart operation, and its mathematical model is described by Eq. (8):

$$AF = 0.01 + 0.09 \times (1 - \frac{FEs}{MaxFEs}) \tag{8}$$

where FEs refers to the current number of evaluations, while $MaxFEs$ presents the maximum number of evaluations.

2.5. Boundary constraint

In both the learning and reflection phases, the following boundary constraint methods will be used to constrain certain variables of individual X_i that are out of bounds. Eq. (9) provides its specific mathematical model:

$$X_{i,j} = \begin{cases} ub, & if\ ub < X_{i,j} \\ lb, & if\ lb > X_{i,j} \end{cases} \tag{9}$$

where $X_{i,j}$ denotes the jth dimensional variable of individual X_i, and if it crosses a boundary, it is limited to the nearby boundary.

2.6. Update rule

The selection mechanism of GO is similar to simulated annealing (SA) (Kirkpatrick, Gelatt, & Vecchi, 1983). They both use Monte Carlo sampling to preserve the solution to the evolutionary failure with a certain probability. However, GO uses a fixed probability P_2 to preserve the inferior solution, while SA uses an adaptive approach. Note that hyperparameter P_2 is set to 0.001 in standard GO. Eq. (10) is the description of the update rule for GO:

$$X_i^{t+1} = \begin{cases} X_i^{t+1} & if\ f\left(X_i^{t+1}\right) < f\left(X_i^{t}\right) \\ \begin{cases} X_i^{t+1} & if\ \mathrm{rand}(0,1) < P_2 \\ X_i^{t} & else \end{cases} & else \end{cases} \tag{10}$$

where $f(X)$ is the objective function, it returns the objective function value.

3. A quadruple parameter adaptation growth optimizer (QAGO) integrated with distribution, confrontation, and balance features

This section mainly explains the proposed QAGO. Additionally, **Algorithm 1** gives the pseudocode of the QAGO algorithm, and Fig. 2 provides a flowchart of QAGO.

3.1. Parameter adaptation based on distribution

In this work, a parameter adaptation mechanism is designed to achieve dynamic tuning of the algorithmic hyperparameters by using parameter sampling that matches certain distributions. Similar methods have been proven to be effective in improving parameter diversity (Tanabe & Fukunaga, 2013; Zhang & Sanderson, 2009). This mechanism preserves the simplicity and diversity of randomly sampled parameters and also has the dynamic tuning characteristics of the parameter adaptation mechanism (Mohamed, Hadi, Fattouh, & Jambi, 2017).

The hyperparameter P_1 plays a pivotal role in population partitioning, and selecting appropriate values for this parameter can significantly improve the algorithm's capability to differentiate between high-quality and low-quality information in the population. This, in turn, facilitates effective population evolution, leading to better outcomes. In standard GO, P_1 is fixed, and specific convergence information is always contributed by individuals in a specific range, which leads to a tendency for the algorithm to get stuck in the local optimum. Therefore, partitioning the population based on dynamic parameter tuning is more effective in alleviating this problem. According to the description in Zhang and Sanderson (2009), the optimal proportion of elite individuals is in the range of $[0.05, 0.2]$. Therefore, this paper has designed the dynamic parameter P_1 in accordance with this idea, as shown in Eq. (11):

$$P_1 = \lfloor \mathrm{rand}(0.05, 0.2) \times N \rfloor \tag{11}$$

where N is the population size and $\mathrm{rand}(0.05, 0.2)$ is a random number obeying a uniform distribution in the range of $[0.05, 0.2]$. Fig. 1 shows the sampling process of P_1.

The hyperparameter P_2 represents the probability of retaining individuals that fail to evolve. In GO, the fixed parameter P_2 is used to probabilistically retain these individuals, which not only avoids wasting computing resources but also endows the algorithm with the ability to escape from the local optimum. However, if the value of P_2 is excessively large, it may impede the algorithm's convergence, whereas if it is minuscule, it may not offer sufficient escape capability. Therefore, this research adjusts P_2 using a Gaussian distribution, with its mean and standard deviation based on the setting of P_2 in GO, both of which are 0.001. Fig. 1 shows the sampling process of P_2. Eq. (12) describes this setting:

$$P_2 = \mathrm{Gaussian}\,(0.001, 0.001) \tag{12}$$

The hyperparameter P_3 controls the probability of using different calculation methods. A fixed P_3 in GO may hamper the algorithm's ability to conduct local searches since it frequently changes more dimensions. In contrast, QAGO uses a smaller parameter value to improve the algorithm's local search capabilities. However, since the algorithm is unable to determine the optimal parameter value, it makes use of a Gaussian distribution to tune P_3, and arrives at an approximate optimal parameter configuration. The center of the Gaussian distribution can be generated at any location within the range of $[0, 0.3]$. For the value 0.3, it is determined based on the setting of parameter P_3 in the standard GO. There may exist better parameters near any center; this paper thereby sets the standard deviation to 0.01 to give the distribution a smaller level of discreteness. Fig. 1 shows the sampling process of P_3. Eq. (13) describes its specific mathematical model:

$$P_3 = \mathrm{Gaussian}\,(\mathrm{rand}(0, 0.3), 0.01) \tag{13}$$

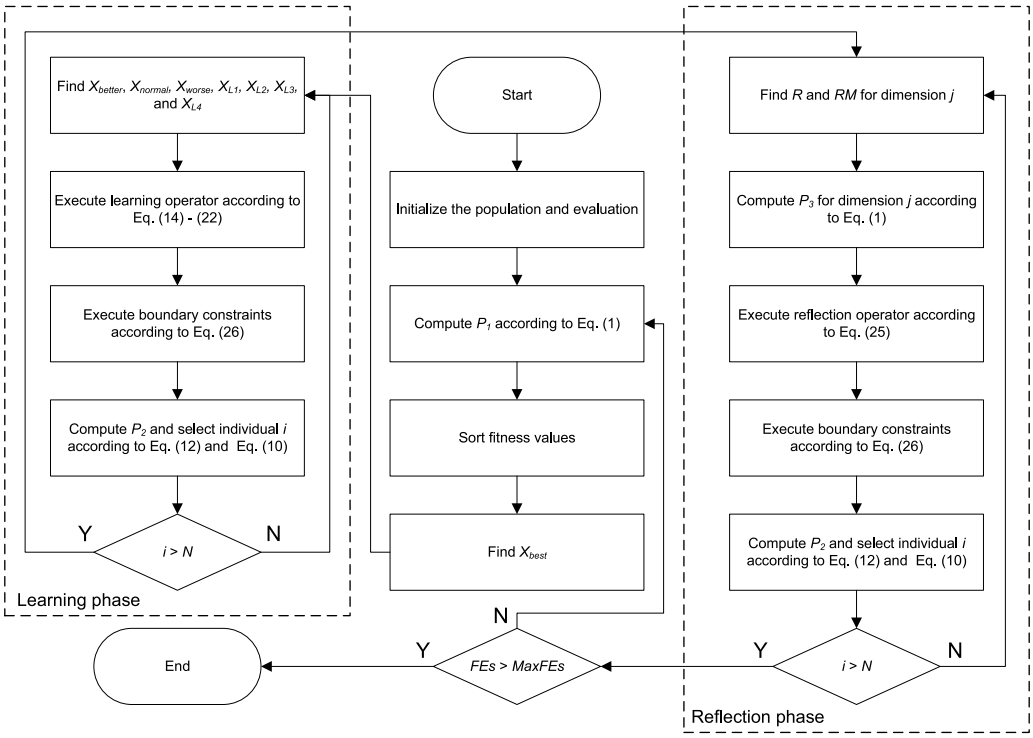


Fig. 2. Flow chart of QAGO.

where $\mathrm{rand}(0, 0.3)$ is a random number obeying the uniform distribution in the range of $[0, 0.3]$. Note that there is a corresponding P_3 for each dimension j of individual X_i.

3.2. Triple parameter self-adaptation based on confrontation and balance

The learning operator in the learning phase of GO completes the approximation process of the individual search direction by accumulating gaps. However, the information of normal individuals is never used in standard GO. As a result, certain information representing the problem space can be lost. Therefore, the calculation method for gaps has been improved in the QAGO algorithm. Each individual has the potential to participate in the computational process to ensure that certain information around the globally optimal region is not lost. In each gap, individuals at neighboring hierarchical levels are always involved in operations, contributing different evolutionary information to the population. Especially, the latter two gaps do not follow this rule, providing uncertain perturbations for the evolution. Specifically, the learning operator of QAGO utilizes five gaps, aiming to approximate the fitness landscape. Eq. (14) describes the gaps involved:

$$\begin{cases} Gap_1 = X_{best} - X_{better} \\ Gap_2 = X_{better} - X_{normal} \\ Gap_3 = X_{normal} - X_{worse} \\ Gap_4 = X_{L1} - X_{L2} \\ Gap_5 = X_{L3} - X_{L4} \end{cases} \tag{14}$$

where $L1$, $L2$, $L3$, and $L4$ are the individual indexes that differ from i. For the explanation of other symbols, please refer to Table 1.

3.2.1. Parameter self-adaptation based on one-dimensional mapping of vectors

One-dimensional mapping can simplify the representation and processing of high-dimensional vectors by mapping them to a one-dimensional space. This reduces computational complexity and memory usage, ultimately improving algorithm efficiency (Zhang, Gong, Zhang, Gu, & Zhang, 2016). Therefore, we apply this idea to parameter self-adaptation. This method involves the one-dimensional mapping of two vectors and utilizes it in the adaptive process of LF_k. Eq. (15) describes the mapping process of paired vectors within Gap_k:

Table 1
Type of individuals, number of individuals in the type, and rank range of the type after ascending ranking based on objective function values.

Individual type	Individual number	Ranking range
X_{best}	1	1
X_{better}	$P_1 - 1$	$[2, P_1]$
X_{normal}	$N - 2P_1$	$[P_1 + 1, N - P_1]$
X_{worse}	P_1	$[N - P_1 + 1, N]$

$$\begin{cases} DGap_1 = \langle X_{best}, X_{better} \rangle \\ DGap_2 = \langle X_{better}, X_{normal} \rangle \\ DGap_3 = \langle X_{normal}, X_{worse} \rangle \\ DGap_4 = \langle X_{L1}, X_{L2} \rangle \\ DGap_5 = \langle X_{L3}, X_{L4} \rangle \end{cases} \tag{15}$$

where $DGap_k$ is the one-dimensional mapping value of two vectors within Gap_k, and $\langle X_{best}, X_{better} \rangle$ represents the inner product operation.

To prevent negative values generated during the mapping process from generating complex numbers in subsequent calculations, we will convert all mapping values into fields of non-negative real numbers. To avoid extremely small positive values and zero, we will shift all values by $2\,|\min(DGap)|$. Therefore, LF_k is calculated based on Eq. (16) using the normalization step after processing the mapped data:

$$LF_k = \frac{DGap_k + 2\,|\min(DGap)|}{\sum_{n=1}^{5}(DGap_n + 2\,|\min(DGap)|)} \tag{16}$$

where min is the function for finding the minimum value, both n and k denote the index of Gap, and $k \in [1, 2, 3, 4, 5]$.

Moreover, we will explore the following from a dimensional perspective: when the corresponding dimensions of two vectors within Gap_k point in opposite directions, this implies instability in those

dimensions; thus, this instability should be weakened in the one-dimensional mapping process. When this situation occurs, less evolutionary information should be obtained from Gap_k using LF_k. Conversely, when the corresponding dimensions of two vectors within Gap_k point in the same direction, this implies stability in those dimensions, and therefore this stability should be maintained in the one-dimensional mapping process. When this situation occurs, more evolutionary information should be obtained from Gap_k using LF_k. This change is advantageous for local exploitation. Therefore, when there is a significant difference between the two vectors within Gap_k, one-dimensional mapping of vectors seems to be a way to weaken the difference.

3.2.2. Parameter self-adaptation based on fitness difference

In standard GO, only individual X_i and worst individuals are considered for adapting SF_i, and the same SF_i is used for scaling any Gap_k, which seems to ignore the differences among different Gap_k. To solve this problem, we have designed a novel SF adaptive method that considers the fitness differences between two vectors within Gap_k. If the fitness difference between two vectors in Gap_k is large, they are considered to provide global information to promote global exploration of individuals. Conversely, if the fitness difference between two vectors within Gap_k is small, they are considered to provide local information to promote the local exploitation of individuals. By considering the differences among different Gap_k, the algorithm can adaptively adjust the contribution of each Gap_k to the evolution process, thus completing the individual's evolutionary process. The calculation process of $FGap_k$ is described by Eq. (17):

$$\begin{cases} FGap_1 = \left|f(X_{best}) - f(X_{better})\right| \\ FGap_2 = \left|f(X_{better}) - f(X_{normal})\right| \\ FGap_3 = \left|f(X_{normal}) - f(X_{worse})\right| \\ FGap_4 = \left|f(X_{L1}) - f(X_{L2})\right| \\ FGap_5 = \left|f(X_{L3}) - f(X_{L4})\right| \end{cases} \tag{17}$$

where $\left|f(X_{best}) - f(X_{better})\right|$ is the fitness difference absolute value between two individuals. Furthermore, to better complete the adaptive process of SF, we normalize all $FGap$ values. Eq. (18) described the process of how SF_k was affected:

$$SF_k = \frac{FGap_k}{\sum_{n=1}^{5} FGap_n}, (k = 1, 2, 3, 4, 5) \tag{18}$$

where SF_k is the adaptive scaling parameter of Gap_k.

Furthermore, we will approach our discussion with a focus on fitness. When the fitness gap between two vectors within Gap_k is large, it means that the similarity between the two vectors is low. More evolutionary information should be obtained from Gap_k using SF_k. Conversely, when the fitness gap between two vectors within Gap_k is small, it means that their similarity is high. Less evolutionary information should be obtained from Gap_k using SF_k. This change is advantageous for global exploration. Therefore, when there are significant differences between two vectors within Gap_k, the fitness-based approach seems to be a way to enhance the difference.

3.2.3. Parameter self-adaptation based on Jensen–Shannon divergence

In QAGO, the design of LF and SF seems to be a "spear and shield" relationship. The contradictions are interdependent yet mutually restrictive. However, contradiction is also a driving force for the development of things.

In information theory, the square root of Jensen–Shannon divergence is a measure of the distance between two probability functions (de Anda-Suárez, Calzada-Ledesma, & Ortiz-Aguilar, 2022; Lamberti, Majtey, Borras, Casas, & Plastino, 2008). Actually, the normalized LF and SF seem to resemble two systems or probability distributions. Therefore, we introduced the concept of Jensen–Shannon divergence during the parameter self-adaptation mechanism to observe and estimate the distance between the two systems and to better balance the adaptive process of the learning operator in QAGO.

The distance between the LF and SF systems, the distance d_{JS}, is calculated using Eq. (19):

$$d_{JS}(LF, SF) = \sqrt{D_{JS}(LF, SF)} \tag{19}$$

where $D_{JS}(LF, SF)$ is the Jensen–Shannon divergence given by Eq. (20):

$$\begin{aligned} D_{JS}(LF, SF) = &\frac{1}{2} KL\left(LF, \frac{LF + SF}{2}\right) \\ &+ \frac{1}{2} KL\left(SF, \frac{LF + SF}{2}\right) \end{aligned} \tag{20}$$

and $KL(LF, SF)$ is the Kullback–Leibler divergence calculated by Eq. (21):

$$KL(LF, SF) = \sum_{k=1}^{5} LF_k \ln\left(\frac{LF_k}{SF_k}\right) \tag{21}$$

and ln denotes the natural logarithm function, and LF_k and SF_k represent only the kth values in LF and SF, respectively.

3.3. Operator refinement

This section redesigns the structure of the learning and reflection operators and implements operator refinement based on parameter adaptation mechanisms.

3.3.1. Learning operator refinement

In standard GO, the learning operator does not utilize normal individuals, resulting in the loss of local information in the problem space representation. Therefore, we have redefined the learning operator's structure for a more efficient evolution for the population. The novel operator structure is described in Eq. (14), and we now use the triple parameter self-adaptation method based on confrontation and balance features to refine the learning operator.

After calculating the distance d_{JS} between LF and SF, we use it to balance the parameter self-adaptation processes of LF and SF. The learning operator has been highly adaptive and fully refined through the above process. Eq. (22) illustrates the process by which the adaptive rules refine the learning operator:

$$X_i^{t+1} = X_i^t + \sum_{k=1}^{5} \left(d_{JS} LF_k + \left(1 - d_{JS}\right) SF_k\right) \cdot Gap_k \tag{22}$$

3.3.2. Reflection operator refinement

This paper redesigned the reflection operator in the reflection phase to refine the chance of premature convergence for the QAGO algorithm. For the jth dimension of individual X_i, Eqs. (23) and (24) give the relevant mathematical model:

$$M = \text{rand}(0, 1) \times \left((R_j - X_{i,j}^t) + (RM_j - X_{i,j}^t)\right) \tag{23}$$

$$X_{i,j}^{t+1} = X_{i,j}^t + M \tag{24}$$

where R_j is the jth dimension of elite R (sampling at the top level), and RM_j is the jth dimension of a random individual, which is different from individual X_i and individual R.

Therefore, the complete reflection operator is reconstructed as Eq. (25):

$$X_{i,j}^{t+1} = \begin{cases} \begin{cases} lb + \text{rand}(0, 1) \times (ub - lb), & if\ \text{rand}(0, 1) < AF \\ X_{i,j}^{t+1} = X_{i,j}^t + M, & else \end{cases} & if\ \text{rand}(0, 1) < P_3 \\ X_{i,j}^t, & else \end{cases} \tag{25}$$

where M represents the evolution information of individual X_i in the jth dimension, while AF linearly decays from 0.01 to 0.

3.4. Boundary constraint

During the search process, individuals may violate the predefined boundaries, resulting in an invalid search outside the effective search space of the algorithm. Common boundary constraint methods include the restart strategy, the penalty strategy, etc (Kreischer, Magalhaes, Barbosa, & Krempser, 2017). In QAGO, different boundary constraint schemes will be applied to different phases to improve the adaptation between components.

In the improved learning phase, due to the accumulation of gaps, it is easy to cause some variables to violate the constraints and thus lose the direction of convergence in this dimension. In this phase, Eq. (26) will be used to correct for some variables:

$$X_{i,j} = \begin{cases} lb + (ub - lb) \times \text{rand}(0,1), & if\ lb > X_{i,j} \\ lb + (ub - lb) \times \text{rand}(0,1), & if\ ub < X_{i,j} \end{cases} \tag{26}$$

In the improved reflection phase, different dimensions of the individual may be guided by different elites. Therefore, at this phase, using Eq. (27) to preserve the tendency of variable changes is more conducive to the convergence of the algorithm. The relevant mathematical model is described as follows:

$$X_{i,j} = \begin{cases} (ub + X_{i,j})/2, & if\ ub < X_{i,j} \\ (lb + X_{i,j})/2, & if\ lb > X_{i,j} \end{cases} \tag{27}$$

Algorithm 1: QAGO

Input: N, D, ub, lb, FEs=0
Output: Global optimal solution:X_{best}
Initialize the population using Eq. (1) and evaluate the individuals ($i = 1, ..., N$)
while $FEs <= MaxFEs$ **do**
  Compute P_1 according to Eq. (11)
  [~,ind]=sort($f(\mathbf{X})$)
  $X_{best} = X(ind(1), :)$
  %Improved reflection phase::
  for $i = 1 : N$ **do**
    $X_{better} = X(ind(\text{randi}([2, 1 + P_1], 1)), :)$
    $X_{normal} = X(ind(\text{randi}([P_1 + 1, N - P_1], 1)), :)$
    $X_{worse} = X(ind(\text{randi}([N - P_1 + 1, N], 1)), :)$
    Find four random individuals that are different from X_i: X_{L1}, X_{L2}, X_{L3}, and X_{L4}
    Compute Gap_k by Eq. (14)
    Compute LF_k according to Eq. (15) and Eq. (16)
    Compute SF_k according to Eq. (17) and Eq. (18)
    Compute d_{JS} between the LF system and the SF system according to Eq. (19), Eq. (20), and Eq. (21)
    Complete the learning process for the ith individual once according to Eq. (22)
    Boundary constraints on variables using Eq. (26)
    Compute P_2 according to Eq. (12)
    Complete the update of the ith individual according to Eq. (10)
    $FEs = FEs + 1$
  end
  % Improved reflection phase::
  for $i = 1 : N$ **do**
    Compute P_3 according to Eq. (13)
    Complete the reflection process for the ith individual once according to Eq. (25)
    Boundary constraints on variables using Eq. (27)
    Compute P_2 according to Eq. (12)
    Complete the update of the ith individual according to Eq. (10)
    $FEs = FEs + 1$
  end
end
Output X_{best}

4. Experimental studies

The experiments are conducted using MATLAB and implemented on a computer equipped with an Intel (R) Core (TM) i7-10700 CPU @ 2.90 GHz, 16 GB of RAM, and an X64 processor. The experiment involves a total of ten novel metaheuristic algorithms, fifty classical metaheuristic algorithms, six advanced self-adaptive algorithms, and five IEEE CEC competition winners.

4.1. Benchmark suites

In classical test benchmarks such as the Sphere function, the fitness landscape is centrosymmetric, and the global optimal solution is located at the center of the search space, which gives a significant advantage to certain algorithms with a bias towards search. Moreover, since the optimal solution of such functions is a zero solution, this allows the algorithms to quickly approach the global optimal solution even if they perform an abnormal search. The shift operation will break the symmetric nature of the problem by transferring the global optimal solution to a non-zero solution, thus avoiding these problems. The rotation operation is an algebraic linear transformation that occurs on the candidate solution when performing a function call. It complicates the exploitation of the problem's symmetry, undermines prior knowledge, and increases unpredictability and complexity, bringing the problem closer to reality (Caraffini & Neri, 2019). Certainly, when the shift property is fused with the rotation property, the problem becomes significantly more challenging. Fig. 3 provides three groups of functions to better illustrate the relevant discussion, each including the basic function, the shifted function, the rotated function, and the shifted and rotated function. It is worth mentioning that the test functions selected in this work are all challenging shifted and rotated functions.

In this experiment, thirty test questions from the CEC 2017 test suite were utilized to evaluate the performance of QAGO. The valid search space for all problems is $[-100, 100]$. According to the CEC 2017 problem definition and evaluation criteria (Wu, Mallipeddi, & Suganthan, 2017), these problems are classified as unimodal functions (F1–F3), simple multimodal functions (F4–F10), hybrid functions (F11–F20), and composition functions (F21–F30). Furthermore, this research employed the IEEE CEC competition winners as competitors for QAGO and evaluated them using twelve test functions from the CEC 2022 test suite. These functions are categorized into unimodal function (CEC 2022 F1), basic functions (CEC 2022 F2-F5), hybrid functions (CEC 2022 F6-F8), and composition functions (CEC 2022 F9-F12). The valid search space for all problems ranges from −100 to 100. All problems exhibit shift and rotation properties, simulating real-world problem scenarios.

4.2. Experimental settings

The experiment compared the performance of QAGO with standard GO, novel high-performance algorithms, classical algorithms, and CEC winners. Table 2 shows algorithm information and related parameter settings involved in the experiment. Note that consistency was maintained with regard to the remaining 50 classical algorithms and relevant literature.

The algorithms were tested in dimensions $D = 30$, $D = 50$, and $D = 100$ for each test function of the CEC 2017 test suite. Similarly, for each test function of the CEC 2022 test suite, the algorithms were tested in dimension $D = 10$. The population size N for general algorithms is set to 40, while the population size settings for the winners are listed in Table 2. Each algorithm was run 30 times independently, with each run calling the function $10,000 \times D$ times to obtain the mean and standard deviation on this problem. Moreover, if the fitness error between the currently found optimal solution and the theoretical optimal solution is lower than the threshold value of 1E-8, it is considered as 0.

In general, the algorithm should be terminated if the obtained results cannot be further improved or if FEs reaches a predefined limit. To ensure fairness, all algorithms in this experiment utilize the stopping criterion of $FEs > MaxFEs$. Additionally, the original boundary constraint method of the algorithms remains unchanged in this experiment.

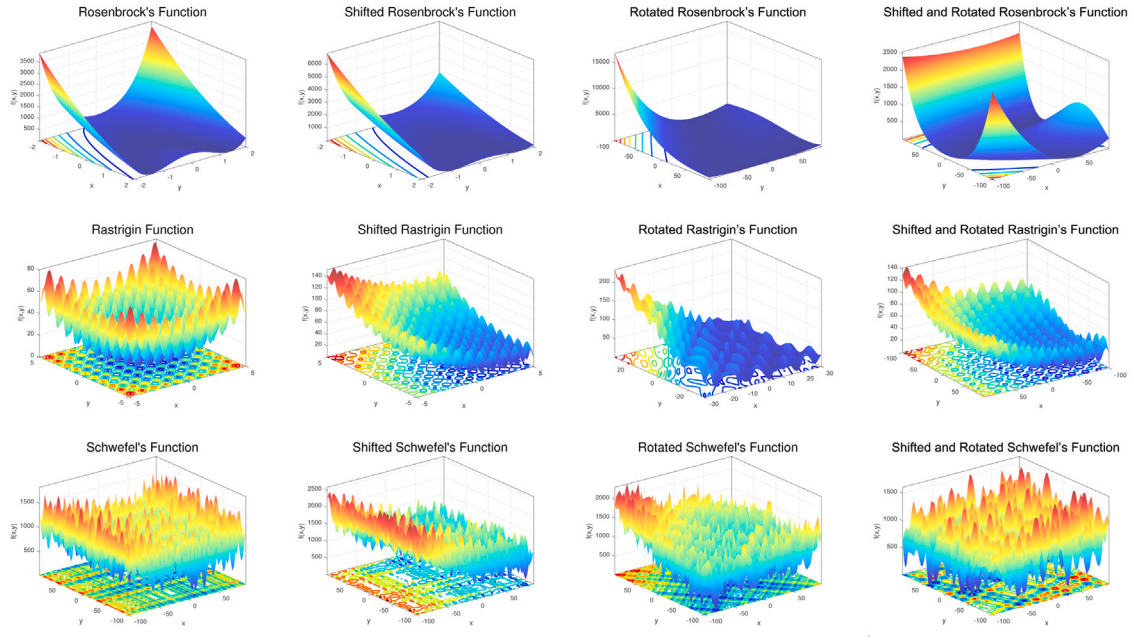


Fig. 3. Increase the complexity of the problem by using shift and rotation operations.

Table 2
Algorithm information and parameter settings.

Algorithm	Parameter settings	Proposed time
Growth optimizer (GO) (Zhang, Gao, et al., 2023)	$P_1 = 5,\ P_2 = 0.001,\ P_3 = 0.3$	2022
Equilibrium optimizer (EO) (Faramarzi, Heidarinejad, Stephens, & Mirjalili, 2020)	$V = 1,\ a_1 = 2,\ a_2 = 1,\ GP = 0.5$	2019
Homonuclear molecules optimization (HMO) (Mahdavi-Meymand & Zounemat-Kermani, 2022)	$\beta = 2$	2022
Aptenodytes forsteri optimization (AFO) (Yang, Deng, Wang, & Liu, 2021)	$w_2 = 0.5,\ w_4 = 1,\ w_5 = 1,\ pe = 0.01,\ gapmin = 5,\ dec = 2,\ L = 10$	2021
Nutcracker optimization algorithm (NOA) (Abdel-Basset, Mohamed, Jameel, et al., 2023)	$\alpha = 0.05,\ Pa_2 = 0.2,\ Prb = 0.2$	2023
Kepler optimization algorithm (KOA) (Abdel-Basset, Mohamed, Azeem, et al., 2023)	$Tc = 3,\ M_0 = 0.1,\ \lambda = 15$	2023
Fick's law algorithm (FLA) (Hashim, Mostafa, Hussien, Mirjalili, & Sallam, 2023)	$c_1 = 0.5,\ c_2 = 2,\ c_3 = 0.1,\ c_4 = 0.2,\ c_5 = 2$	2023
Special relativity search (SRS) (Goodarzimehr, Shojaee, Hamzehei-Javaran, & Talatahari, 2022)	$\mu = 1,\ m = 1$	2022
Coati optimization algorithm (COA) (Dehghani, Montazeri, Trojovská, & Trojovský, 2023)	$division\ \ ratio = 0.5$	2023
Beluga whale optimization (BWO) (Zhong, Li, & Meng, 2022)	$W_f \in [0.05, 0.1],\ \alpha = 1.5,\ KD = 0.05$	2022
L-SHADE (IEEE CEC 2014 winner) (Tanabe & Fukunaga, 2014)	$N = 18D,\ N_{min} = 4,\ \mu F = \mu CR = \mu freq = 0.5,\ r = 2.6,\ p = 0.11,\ H = 6$	2014
L-SHADE-EpSin (IEEE CEC 2016 joint winner) (Awad et al., 2017)	$N = 18D,\ N_{min} = 4,\ \mu F = \mu CR = \mu freq = 0.5,\ r = 1.4,\ p = 0.11,\ H = 5,\ G_{ls} = 250$	2016
UMOEAs-II (IEEE CEC 2016 joint winner) (Elsayed, Hamza, & Sarker, 2016)	$N = 18D,\ N_{min} = 4,\ N_{CMA} = 4 + \lfloor 3log(D) \rfloor,\ \sigma = 0.3,\ \mu F = \mu CR = 0.5,\ r = 2.6,\ p = 0.1,\ H = 6,\ CS = 50,\ cFES_{ls} = 0.2$	2016
EBOwithCMAR (IEEE CEC 2017 winner) (Kumar, Misra, & Singh, 2017)	$N = 18D,\ N_{min} = 4,\ N_{CMA} = 4 + \lfloor 3log(D) \rfloor,\ \sigma = 0.3,\ \mu F = \mu CR = 0.5,\ \mu freq = 0.7,\ r = 2.6,\ H = 6,\ CS = 50/100/150,\ prob_{ls} = 0.1,\ cFES_{ls} = 0.25$	2017
HS-ES (IEEE CEC 2018 winner) (Zhang & Shi, 2018)	$\lambda = 80 + \lfloor 3log(D) \rfloor,\ M_1 = 200,\ N_1 = 100,\ cc = 0.96,\ I = 20,\ M_2 = 200/450/600,\ N_2 = 160/360/480$	2018

4.3. Compare QAGO with novel metaheuristic algorithms

Tables 3, 4, and 5 summarize the test results of GO, EO, HMO, AFO, NOA, KOA, FLA, SRS, COA, BWO, and QAGO algorithms in three different dimensions ($D = 30/50/100$), which include the mean and the standard deviation over the 30 tested functions on the CEC 2017 test suite. Additionally, the Wilcoxon rank-sum test (Carrasco, García, Rueda, Das, & Herrera, 2020) is performed at the significance level of $P = 0.05$ to estimate the statistical significance of the results of the different algorithms. The statistical results are summarized in Table 6, where the symbols "+", "≈" and "−", indicate the number of times the competing algorithms outperform, approximate, and underperform the QAGO algorithm at the significance level, respectively. The ranking of all algorithms based on statistics is calculated using the Friedman test (Friedman, 1940) and the Kruskal–Wallis test (McKight & Najab, 2010). Additionally, the Kruskal–Wallis test is employed to determine the number of times each algorithm attains the first and last positions.

The detailed experimental results are listed in Tables 3, 4, and 5. The following can be observed.

For the unimodal functions F1–F3, QAGO is able to converge to the global optimal solution on the F1 function at $D = 30$, 50, and 100. KOA and GO are only achieved on $D = 30$. Function F2 causes most algorithms to exhibit significant instability with increasing dimensionality. However, QAGO is able to mitigate this phenomenon and shows significant performance. The global optimum of F3 is in a huge smooth space, which makes the convergence speed of most algorithms slow down rapidly. However, GO and QAGO have the best performance as they avoid incorrect convergence directions and improve convergence speed. Furthermore, the performance of all algorithms decreases with increasing dimensions. However, QAGO seems not to undergo influence with dimensional increase. This is because QAGO's quadruple parameter adaptation mechanism allows for adaptive parameter adjustments, enabling the algorithm to better adapt to the search environment and achieve improved performance.

For the simple multimodal functions F4–F10, all algorithms have a reduced ability to find the best solution with increasing problem dimensions. GO and NOA show better optimization abilities at low dimensions. However, QAGO is able to maintain stability and achieve

Table 3
Mean and standard deviation of the fitness error for the CEC 2017 test suite of 30D functions F1–F30 on 30 runs (the best entries are in bold).

No.	Metric	EO	GO	HMO	AFO	NOA	KOA	FLA	SRS	COA	BWO	QAGO
F1	Mean	5.61E+03	**0.00E+00**	4.40E+03	5.68E+08	1.27E−05	**0.00E+00**	1.47E+05	5.01E+10	2.78E+10	4.51E+10	**0.00E+00**
	Std	6.54E+03	**2.54E−23**	4.63E+03	8.45E+08	1.70E−05	**6.88E−21**	6.24E+04	6.98E+09	6.73E+09	3.82E+09	**1.74E−25**
F2	Mean	2.84E+08	**0.00E+00**	3.12E+12	8.32E+26	6.57E+02	7.21E−08	7.70E+06	8.53E+45	2.40E+38	2.30E+42	**0.00E+00**
	Std	6.76E+08	**2.87E−09**	4.31E+12	4.54E+27	2.59E+03	3.04E−07	1.49E+07	4.46E+46	1.07E+39	4.09E+42	**2.83E−12**
F3	Mean	2.06E+01	**0.00E+00**	9.67E+03	5.50E+02	2.20E−06	2.72E−05	1.72E+02	7.79E+04	6.00E+04	7.54E+04	**0.00E+00**
	Std	3.49E+01	**3.41E−28**	3.98E+03	4.15E+02	2.20E−06	1.32E−04	8.62E+01	5.70E+03	8.07E+03	3.09E+03	**2.26E−28**
F4	Mean	4.80E+01	2.16E+01	6.00E+01	1.33E+02	**5.55E+00**	6.54E+00	9.96E+01	8.72E+03	4.08E+03	9.82E+03	2.60E+01
	Std	3.14E+01	2.86E+01	2.03E+01	6.44E+01	**1.01E+01**	1.72E+01	3.21E+01	5.74E+03	1.78E+03	9.95E+02	2.96E+01
F5	Mean	7.02E+01	**2.52E+01**	1.06E+02	1.25E+02	5.38E+01	8.03E+01	7.89E+01	3.87E+02	3.25E+02	3.95E+02	4.09E+01
	Std	1.64E+01	**7.71E+00**	2.05E+01	3.42E+01	1.15E+01	1.72E+01	1.63E+01	2.31E+01	2.26E+01	1.73E+01	7.85E+00
F6	Mean	1.03E−01	2.07E−03	2.14E+01	3.91E−01	1.27E−03	4.59E−03	1.65E−01	6.82E+01	6.69E+01	8.09E+01	**0.00E+00**
	Std	1.64E−01	2.33E−03	9.01E+00	8.82E−01	1.17E−03	9.35E−03	3.39E−02	3.64E+00	7.52E+00	4.42E+00	**3.47E−08**
F7	Mean	1.10E+02	**5.66E+01**	1.19E+02	1.63E+02	8.67E+01	1.25E+02	1.09E+02	6.60E+02	4.41E+02	6.31E+02	7.74E+01
	Std	2.53E+01	**8.47E+00**	1.95E+01	3.33E+01	1.59E+01	1.35E+01	2.24E+01	3.51E+01	5.13E+01	3.00E+01	7.18E+00
F8	Mean	6.71E+01	**2.62E+01**	1.02E+02	1.27E+02	5.27E+01	7.97E+01	9.10E+01	3.14E+02	2.67E+02	3.13E+02	3.90E+01
	Std	1.63E+01	**6.08E+00**	1.69E+01	3.65E+01	1.04E+01	1.96E+01	2.27E+01	1.94E+01	1.75E+01	1.46E+01	7.76E+00
F9	Mean	7.24E+01	2.83E+00	5.07E+02	1.03E+03	2.16E+00	5.14E+00	4.28E+02	6.43E+03	6.55E+03	8.95E+03	**0.00E+00**
	Std	1.26E+02	3.10E+00	4.11E+02	6.21E+02	3.00E+00	5.84E+00	5.11E+02	4.38E+02	1.12E+03	9.41E+02	**1.03E−31**
F10	Mean	3.51E+03	**2.55E+03**	3.16E+03	3.00E+03	2.62E+03	4.79E+03	2.84E+03	5.88E+03	6.38E+03	7.21E+03	3.74E+03
	Std	5.73E+02	**1.36E+03**	2.97E+02	4.98E+02	5.69E+02	3.85E+02	4.24E+02	6.28E+02	3.78E+02	3.28E+02	4.37E+02
F11	Mean	7.33E+01	**2.15E+01**	1.23E+02	1.80E+02	2.34E+01	4.98E+01	6.51E+01	7.17E+03	3.59E+03	4.65E+03	3.25E+01
	Std	5.40E+01	**2.13E+01**	2.46E+01	4.65E+01	8.14E+00	3.03E+01	3.55E+01	1.84E+03	1.37E+03	8.13E+02	2.88E+01
F12	Mean	7.35E+04	2.41E+03	1.23E+06	5.06E+06	9.13E+03	1.42E+04	2.69E+06	1.75E+10	2.45E+09	7.62E+09	**6.73E+02**
	Std	7.87E+04	5.86E+03	8.70E+05	7.76E+06	6.63E+03	1.36E+04	1.63E+06	3.70E+09	8.20E+08	1.45E+09	**6.69E+02**
F13	Mean	1.37E+04	8.42E+01	2.19E+04	3.28E+05	1.01E+02	2.21E+03	3.41E+04	1.98E+10	6.60E+08	3.82E+09	**4.13E+01**
	Std	1.69E+04	3.90E+01	8.07E+03	1.08E+06	4.61E+01	7.73E+03	4.66E+04	5.35E+09	5.26E+08	1.03E+09	**2.35E+01**
F14	Mean	8.23E+03	3.39E+01	1.80E+03	3.09E+04	**2.13E+01**	4.79E+01	1.60E+05	5.20E+06	8.67E+04	1.51E+06	2.98E+01
	Std	8.45E+03	4.16E+00	1.21E+03	2.62E+04	**8.95E+00**	8.33E+00	9.03E+04	3.45E+06	6.75E+04	6.40E+05	4.33E+00
F15	Mean	6.44E+03	2.49E+01	1.20E+04	9.06E+03	2.09E+01	3.17E+01	1.21E+04	4.76E+08	7.21E+05	1.01E+08	**1.08E+01**
	Std	7.48E+03	2.58E+01	5.55E+03	1.09E+04	7.37E+00	1.99E+01	1.32E+04	3.25E+08	7.81E+05	4.14E+07	**3.10E+00**
F16	Mean	7.36E+02	**2.38E+02**	8.37E+02	1.04E+03	4.38E+02	3.89E+02	1.07E+03	3.69E+03	2.34E+03	3.31E+03	3.57E+02
	Std	2.33E+02	**1.58E+02**	1.94E+02	3.05E+02	1.31E+02	1.91E+02	3.20E+02	1.59E+03	2.55E+02	2.72E+02	1.36E+02
F17	Mean	2.21E+02	7.47E+01	2.61E+02	4.96E+02	**4.84E+01**	9.26E+01	4.41E+02	3.44E+03	1.02E+03	1.66E+03	5.36E+01
	Std	1.13E+02	5.43E+01	7.99E+01	2.22E+02	**1.35E+01**	5.12E+01	1.96E+02	3.43E+03	1.69E+02	3.15E+02	1.09E+01
F18	Mean	1.59E+05	3.76E+01	8.51E+04	7.30E+05	3.19E+01	5.95E+02	3.75E+05	1.64E+07	1.55E+06	1.61E+07	**2.67E+01**
	Std	1.53E+05	7.60E+00	3.32E+04	8.41E+05	5.42E+00	6.80E+02	2.98E+05	1.73E+07	1.26E+06	8.34E+06	**2.37E+00**
F19	Mean	8.24E+03	1.29E+01	1.14E+05	2.33E+04	1.30E+01	2.10E+01	1.59E+04	1.12E+09	5.79E+06	1.65E+08	**1.04E+01**
	Std	1.34E+04	2.81E+00	1.94E+05	6.78E+04	2.59E+00	3.69E+00	1.89E+04	5.67E+08	5.02E+06	7.41E+07	**2.21E+00**
F20	Mean	2.68E+02	8.57E+01	4.22E+02	4.40E+02	**8.17E+01**	9.25E+01	4.08E+02	9.16E+02	6.41E+02	8.09E+02	9.84E+01
	Std	1.65E+02	6.02E+01	9.38E+01	1.46E+02	**6.39E+01**	7.11E+01	1.39E+02	1.95E+02	1.27E+02	1.03E+02	5.66E+01
F21	Mean	2.64E+02	2.23E+02	**2.10E+02**	3.43E+02	2.44E+02	2.68E+02	2.87E+02	7.05E+02	5.01E+02	3.87E+02	2.38E+02
	Std	2.16E+01	5.94E+00	**9.34E+01**	3.76E+01	9.68E+00	1.93E+01	1.88E+01	7.38E+01	2.89E+01	1.15E+02	6.92E+00
F22	Mean	1.13E+03	4.54E+02	6.57E+02	2.34E+03	**1.00E+02**	1.01E+02	2.16E+03	7.31E+03	3.36E+03	5.22E+03	**1.00E+02**
	Std	1.54E+03	8.87E+02	1.29E+03	1.81E+03	**4.49E−01**	1.24E+00	1.53E+03	6.14E+02	8.38E+02	6.97E+02	**1.14E−12**
F23	Mean	4.07E+02	3.78E+02	4.35E+02	5.87E+02	3.91E+02	4.15E+02	4.40E+02	1.46E+03	8.21E+02	9.43E+02	**3.64E+02**
	Std	1.73E+01	8.95E+00	3.34E+01	6.61E+01	1.19E+01	2.07E+01	2.18E+01	2.71E+02	6.89E+01	3.32E+01	**1.03E+01**
F24	Mean	4.92E+02	4.58E+02	**3.81E+02**	6.92E+02	4.70E+02	4.74E+02	5.68E+02	1.78E+03	8.50E+02	1.07E+03	4.41E+02
	Std	1.87E+01	1.10E+01	**1.57E+02**	7.98E+01	1.20E+01	2.07E+01	3.92E+01	3.61E+02	6.76E+01	6.26E+01	9.50E+00
F25	Mean	**3.87E+02**	**3.87E+02**	3.89E+02	4.05E+02	**3.87E+02**	3.89E+02	3.93E+02	2.20E+03	1.50E+03	1.70E+03	**3.87E+02**
	Std	**2.08E+00**	**2.26E+00**	3.28E+00	2.75E+01	**3.55E+00**	8.23E+00	1.35E+01	8.60E+02	2.81E+02	1.13E+02	**1.01E−01**
F26	Mean	1.63E+03	1.18E+03	**2.41E+02**	2.84E+03	1.39E+03	1.43E+03	1.94E+03	8.13E+03	5.59E+03	6.09E+03	1.13E+03
	Std	3.13E+02	3.81E+02	**4.87E+01**	1.18E+03	6.35E+02	4.34E+02	5.00E+02	1.30E+03	6.42E+02	1.02E+03	1.09E+02
F27	Mean	5.22E+02	4.99E+02	5.32E+02	5.63E+02	5.01E+02	5.05E+02	5.18E+02	2.73E+03	7.84E+02	1.04E+03	**5.03E+02**
	Std	9.84E+00	1.11E+01	1.04E+01	3.28E+01	1.21E+01	1.39E+01	1.00E+01	4.93E+02	7.94E+01	1.07E+02	**8.12E+00**
F28	Mean	3.12E+02	3.44E+02	4.17E+02	4.85E+02	3.14E+02	3.46E+02	4.45E+02	3.59E+03	2.32E+03	3.14E+03	**3.07E+02**
	Std	3.58E+01	6.06E+01	8.92E+00	6.37E+01	3.57E+01	5.82E+01	2.35E+01	4.66E+02	4.84E+02	2.66E+02	**2.62E+01**
F29	Mean	6.86E+02	**4.74E+02**	1.01E+03	9.31E+02	4.90E+02	5.64E+02	8.45E+02	4.58E+03	1.95E+03	3.22E+03	4.96E+02
	Std	1.64E+02	**5.22E+01**	1.18E+02	2.54E+02	4.79E+01	7.88E+01	1.96E+02	1.13E+03	3.21E+02	3.78E+02	2.97E+01
F30	Mean	6.09E+03	2.10E+03	5.02E+05	2.21E+05	2.28E+03	2.92E+03	9.83E+03	3.09E+09	9.48E+07	4.35E+08	**2.03E+03**
	Std	3.98E+03	2.44E+02	3.87E+05	9.49E+05	4.00E+02	1.43E+03	4.12E+03	1.07E+09	4.15E+07	1.31E+08	**5.78E+01**

a high-quality solution in different dimensions. As the dimension increases, the performance of standard GO gradually deteriorates. However, the advantage of QAGO gradually becomes significant. Furthermore, the potential of the standard GO algorithm is still visible when performing numerical optimization.

For the hybrid functions F11–F20, QAGO obtains better results on such functions compared to other algorithms. Especially on F13, F15, and F19, the dominance of QAGO remains unchanged. NOA also seems to have some advantage at $D = 30$, especially on F14, F17, and F20.

Table 4
Mean and standard deviation of the fitness error for the CEC 2017 test suite of 50D functions F1–F30 on 30 runs (the best entries are in bold).

No.	Metric	EO	GO	HMO	AFO	NOA	KOA	FLA	SRS	COA	BWO	QAGO
F1	Mean	3.72E+03	3.10E−02	4.07E+03	7.74E+08	1.03E+03	6.07E−03	3.47E+06	1.10E+11	7.10E+10	9.58E+10	**0.00E+00**
	Std	5.38E+03	7.08E−02	4.95E+03	1.18E+09	1.32E+03	2.11E−02	8.20E+05	3.80E+09	9.76E+09	4.98E+09	**8.69E−25**
F2	Mean	2.23E+21	4.34E−07	6.99E+20	1.85E+53	2.15E+08	2.39E+00	2.18E+21	1.49E+82	2.47E+70	1.06E+75	**0.00E+00**
	Std	1.21E+22	6.02E−07	1.65E+21	1.01E+54	1.12E+09	9.13E+00	9.75E+21	2.64E+82	6.59E+70	3.61E+75	**1.08E−09**
F3	Mean	3.82E+03	**0.00E+00**	4.21E+04	2.67E+03	3.06E−01	4.80E+01	1.00E+03	1.71E+05	1.78E+05	1.78E+05	**0.00E+00**
	Std	2.55E+03	**7.35E−11**	1.08E+04	1.60E+03	4.19E−01	4.16E+01	4.32E+02	1.50E+04	2.80E+04	1.10E+04	**5.63E−17**
F4	Mean	7.53E+01	**2.18E+01**	1.02E+02	2.64E+02	6.61E+01	6.35E+01	1.66E+02	3.18E+04	1.72E+04	2.72E+04	3.45E+01
	Std	4.70E+01	**3.23E+01**	2.91E+01	1.50E+02	5.27E+01	4.55E+01	4.84E+01	1.03E+04	4.90E+03	2.81E+03	4.41E+01
F5	Mean	1.79E+02	6.06E+01	2.28E+02	2.60E+02	1.57E+02	1.77E+02	1.72E+02	5.63E+02	5.49E+02	6.49E+02	**4.86E+01**
	Std	3.48E+01	1.08E+01	3.98E+01	5.05E+01	2.29E+01	5.40E+01	3.29E+01	5.11E+01	1.93E+01	1.52E+01	**3.73E+01**
F6	Mean	1.33E+00	2.05E−02	3.62E+01	4.99E−01	1.40E−02	4.37E−01	5.49E−01	7.88E+01	8.61E+01	9.56E+01	**1.02E−05**
	Std	8.09E−01	1.24E−02	1.02E+01	4.72E−01	1.02E−02	4.74E−01	1.08E−01	2.74E+00	5.12E+00	3.62E+00	**2.88E−05**
F7	Mean	2.38E+02	**1.18E+02**	2.51E+02	3.38E+02	1.96E+02	2.99E+02	2.36E+02	1.32E+03	9.46E+02	1.18E+03	1.71E+02
	Std	5.30E+01	**1.30E+01**	3.63E+01	6.01E+01	2.68E+01	3.94E+01	3.15E+01	8.20E+01	6.38E+01	4.44E+01	3.13E+01
F8	Mean	1.88E+02	5.66E+01	2.26E+02	2.90E+02	1.43E+02	1.88E+02	1.82E+02	6.35E+02	5.82E+02	6.59E+02	**4.78E+01**
	Std	3.76E+01	8.14E+00	3.79E+01	5.55E+01	1.81E+01	4.11E+01	3.48E+01	5.73E+01	2.56E+01	1.93E+01	**3.07E+01**
F9	Mean	9.67E+02	3.87E+01	3.54E+03	9.44E+03	2.15E+02	3.56E+02	5.02E+03	1.99E+04	2.73E+04	3.33E+04	**1.11E−01**
	Std	1.08E+03	1.89E+01	2.74E+03	5.82E+03	2.45E+02	7.95E+02	3.32E+03	1.46E+03	3.72E+03	2.02E+03	**2.04E−01**
F10	Mean	6.05E+03	**3.80E+03**	5.72E+03	5.76E+03	4.60E+03	9.26E+03	5.00E+03	1.24E+04	1.16E+04	1.32E+04	7.70E+03
	Std	7.89E+02	**2.02E+03**	3.52E+02	7.91E+02	7.43E+02	6.31E+02	6.52E+02	5.59E+02	6.13E+02	3.65E+02	9.03E+02
F11	Mean	1.62E+02	**5.34E+01**	2.90E+02	3.14E+02	1.00E+02	9.68E+01	1.30E+02	2.04E+04	9.47E+03	1.74E+04	6.18E+01
	Std	6.70E+01	**1.02E+01**	4.61E+01	3.81E+02	1.54E+01	2.77E+01	3.22E+01	2.10E+03	1.84E+03	1.81E+03	2.23E+01
F12	Mean	8.00E+05	**2.22E+04**	1.21E+07	4.52E+08	1.13E+05	8.87E+04	1.34E+07	9.23E+10	1.78E+10	4.20E+10	4.49E+04
	Std	4.51E+05	**2.46E+04**	6.54E+06	6.42E+08	6.40E+04	5.64E+04	5.89E+06	2.03E+10	6.67E+09	4.82E+09	3.47E+04
F13	Mean	5.85E+03	7.27E+02	3.55E+04	2.21E+07	1.35E+03	2.73E+03	2.42E+04	6.27E+10	5.35E+09	2.48E+10	**1.54E+02**
	Std	4.12E+03	4.07E+02	9.70E+03	6.92E+07	1.78E+03	4.32E+03	2.00E+04	1.90E+10	3.06E+09	5.36E+09	**4.73E+01**
F14	Mean	4.14E+04	8.53E+01	1.95E+04	4.74E+05	7.71E+01	7.27E+02	7.18E+05	1.91E+08	2.03E+06	2.31E+07	**4.95E+01**
	Std	3.27E+04	2.12E+01	1.31E+04	6.83E+05	1.04E+01	9.57E+02	4.23E+05	1.49E+08	1.16E+06	7.48E+06	**6.67E+00**
F15	Mean	9.56E+03	6.80E+02	1.50E+04	2.39E+06	1.47E+02	2.07E+03	1.63E+04	8.10E+09	1.85E+08	3.50E+09	**6.69E+01**
	Std	7.45E+03	3.26E+03	8.02E+03	1.30E+07	4.19E+01	3.13E+03	1.30E+04	5.06E+09	1.43E+08	9.79E+08	**2.43E+01**
F16	Mean	1.49E+03	**5.13E+02**	1.44E+03	1.93E+03	7.91E+02	9.86E+02	1.93E+03	1.07E+04	4.54E+03	5.83E+03	6.28E+02
	Std	4.91E+02	**2.35E+02**	2.45E+02	4.52E+02	2.47E+02	2.79E+02	3.23E+02	1.92E+03	3.76E+02	5.37E+02	1.97E+02
F17	Mean	1.06E+03	**4.58E+02**	1.19E+03	1.64E+03	6.63E+02	6.74E+02	1.28E+03	1.31E+04	2.81E+03	3.76E+03	5.92E+02
	Std	2.59E+02	**2.23E+02**	1.95E+02	3.40E+02	1.55E+02	2.16E+02	3.07E+02	6.61E+03	3.93E+02	3.60E+02	1.64E+02
F18	Mean	2.56E+05	1.38E+02	2.06E+05	1.74E+06	9.47E+02	1.11E+04	2.24E+06	2.60E+08	9.70E+06	6.69E+07	**6.16E+01**
	Std	1.39E+05	5.89E+01	1.06E+05	1.44E+06	7.62E+02	7.71E+03	1.68E+06	1.21E+08	6.36E+06	2.64E+07	**1.92E+01**
F19	Mean	1.95E+04	4.20E+01	1.34E+05	2.52E+05	3.56E+01	4.49E+03	1.76E+04	6.79E+09	3.17E+08	2.03E+09	**2.14E+01**
	Std	1.19E+04	1.23E+01	1.58E+05	1.00E+06	1.16E+01	9.37E+03	1.43E+04	1.96E+09	1.40E+08	3.43E+08	**4.13E+00**
F20	Mean	9.30E+02	**2.49E+02**	1.03E+03	1.12E+03	3.30E+02	5.00E+02	9.32E+02	2.01E+03	1.37E+03	1.71E+03	3.20E+02
	Std	3.29E+02	**1.71E+02**	1.72E+02	3.51E+02	1.54E+02	1.71E+02	2.44E+02	3.15E+02	1.62E+02	1.56E+02	1.19E+02
F21	Mean	3.45E+02	2.54E+02	3.97E+02	5.32E+02	3.20E+02	3.76E+02	3.88E+02	1.11E+03	8.66E+02	1.01E+03	**2.35E+02**
	Std	3.88E+01	1.02E+01	6.07E+01	6.83E+01	2.22E+01	5.00E+01	3.66E+01	1.33E+02	5.00E+01	3.39E+01	**6.46E+00**
F22	Mean	7.06E+03	**3.34E+03**	6.07E+03	7.05E+03	4.38E+03	9.15E+03	5.65E+03	1.41E+04	1.29E+04	1.33E+04	5.22E+03
	Std	8.23E+02	**1.27E+03**	1.23E+03	8.57E+02	2.27E+03	2.51E+03	5.77E+02	5.68E+02	5.75E+02	7.69E+02	3.77E+03
F23	Mean	5.61E+02	4.90E+02	6.61E+02	1.01E+03	5.60E+02	5.66E+02	6.40E+02	2.98E+03	1.44E+03	1.64E+03	**4.61E+02**
	Std	4.20E+01	1.47E+01	5.39E+01	1.10E+02	2.78E+01	5.00E+01	3.51E+01	3.96E+02	1.00E+02	6.93E+01	**1.37E+01**
F24	Mean	6.17E+02	5.78E+02	7.07E+02	1.20E+03	6.33E+02	6.26E+02	7.92E+02	3.48E+03	1.51E+03	1.83E+03	**5.49E+02**
	Std	2.45E+01	2.39E+01	5.81E+01	1.42E+02	2.03E+01	4.41E+01	7.11E+01	3.15E+02	1.10E+02	9.14E+01	**1.42E+01**
F25	Mean	5.54E+02	5.25E+02	**5.15E+02**	5.89E+02	5.49E+02	5.54E+02	5.55E+02	1.29E+04	7.00E+03	1.04E+04	5.21E+02
	Std	4.30E+01	3.39E+01	**2.09E+01**	6.91E+01	4.03E+01	3.68E+01	3.81E+01	4.71E+02	1.05E+03	6.47E+02	3.21E+01
F26	Mean	2.81E+03	1.98E+03	3.76E+02	5.50E+03	2.88E+03	2.59E+03	3.38E+03	1.39E+04	1.03E+04	1.30E+04	**1.63E+03**
	Std	1.36E+03	2.04E+02	4.11E+02	1.64E+03	1.01E+03	4.63E+02	4.05E+02	1.12E+03	1.08E+03	3.69E+02	**1.49E+02**
F27	Mean	6.70E+02	5.85E+02	7.37E+02	9.77E+02	5.74E+02	6.64E+02	6.67E+02	6.18E+03	1.81E+03	2.45E+03	**5.45E+02**
	Std	8.71E+01	6.89E+01	4.17E+01	1.50E+02	5.58E+01	7.94E+01	9.57E+01	9.32E+02	2.29E+02	2.34E+02	**2.18E+01**
F28	Mean	5.03E+02	4.87E+02	4.81E+02	6.74E+02	4.92E+02	4.96E+02	5.10E+02	1.03E+04	7.00E+03	8.24E+03	**4.86E+02**
	Std	3.15E+01	2.97E+01	1.42E+01	2.00E+02	1.69E+01	2.97E+01	2.52E+01	1.73E+03	1.07E+03	5.62E+02	**2.21E+01**
F29	Mean	1.20E+03	4.97E+02	1.87E+03	1.77E+03	6.11E+02	6.13E+02	1.14E+03	6.29E+04	5.61E+03	1.09E+04	**4.59E+02**
	Std	3.81E+02	1.10E+02	3.38E+02	4.56E+02	1.49E+02	1.42E+02	2.69E+02	9.27E+04	8.34E+02	2.24E+03	**7.56E+01**
F30	Mean	1.09E+06	5.86E+05	4.11E+07	4.95E+06	5.97E+05	6.26E+05	1.29E+06	1.01E+10	1.04E+09	3.02E+09	**5.85E+05**
	Std	2.38E+05	1.18E+04	8.68E+06	8.53E+06	2.67E+04	5.66E+04	5.17E+05	2.61E+09	4.55E+08	5.17E+08	**1.92E+04**

Nevertheless, it is similar to HMO, AFO, KOA, FLA, SRS, COA, or BWO, which does not have the advantage when $D = 50$ and $D = 100$.

For the composition functions F21–F30, QAGO still holds a dominant advantage in these problems in terms of performance. EO, GO, HMO, AFO, NOA, and KOA only have advantages in a few situations. In addition, FLA, SRS, COA, and BWO algorithms cannot solve these problems well in any dimension. In summary, QAGO's quadruple parameter adaptation and new operators effectively iterated the solution, resulting in competitive performance, especially compared with GO.

To further investigate the convergence behavior of different algorithms during the iterative process and their sensitivity to different

Table 5
Mean and standard deviation of the fitness error for the CEC 2017 test suite of 100D functions F1–F30 on 30 runs (the best entries are in bold).

No.	Metric	EO	GO	HMO	AFO	NOA	KOA	FLA	SRS	COA	BWO	QAGO
F1	Mean	8.44E+03	5.40E+03	6.15E+03	7.49E+08	4.39E+03	1.51E−01	8.76E+07	2.49E+11	2.00E+11	2.42E+11	**0.00E+00**
	Std	9.76E+03	6.89E+03	5.69E+03	1.07E+09	2.80E+03	2.81E−01	8.81E+06	1.10E+10	1.56E+10	3.96E+09	**2.11E−13**
F2	Mean	6.76E+56	4.32E+23	2.38E+60	2.73E+127	1.22E+28	1.26E+18	9.30E+50	8.16E+178	1.46E+164	2.09E+165	**1.73E+01**
	Std	3.03E+57	2.19E+24	1.30E+61	1.05E+128	5.46E+28	6.89E+18	2.72E+51	6.55E+04	6.55E+04	6.55E+04	**8.44E+01**
F3	Mean	6.49E+04	1.15E+01	2.35E+05	2.04E+04	2.21E+03	2.36E+04	2.51E+04	3.25E+05	3.58E+05	3.32E+05	**1.39E−05**
	Std	1.18E+04	1.85E+01	3.01E+04	5.35E+03	8.24E+02	6.07E+03	3.84E+03	2.89E+03	3.50E+04	8.91E+03	**2.32E−05**
F4	Mean	2.26E+02	2.02E+02	2.58E+02	5.85E+02	2.22E+02	1.78E+02	2.74E+02	1.05E+05	7.13E+04	8.03E+04	**1.12E+02**
	Std	4.10E+01	4.48E+01	1.57E+01	2.31E+02	4.59E+01	3.65E+01	3.89E+01	1.67E+04	1.30E+04	5.03E+03	**5.07E+01**
F5	Mean	5.12E+02	1.52E+02	5.88E+02	8.28E+02	4.93E+02	4.28E+02	4.91E+02	1.41E+03	1.32E+03	1.53E+03	**1.30E+02**
	Std	6.96E+01	2.19E+01	9.55E+01	1.06E+02	6.05E+01	1.15E+02	7.02E+01	6.83E+01	4.11E+01	2.01E+01	**1.66E+01**
F6	Mean	1.57E+01	6.30E−02	5.46E+01	8.08E−01	1.09E+00	5.65E+00	3.36E+00	8.58E+01	9.61E+01	1.06E+02	**3.43E−03**
	Std	8.91E+00	2.26E−02	7.87E+00	5.61E−01	6.22E−01	2.28E+00	1.23E+00	3.16E+00	3.72E+00	2.43E+00	**2.57E−03**
F7	Mean	9.04E+02	3.09E+02	6.98E+02	1.08E+03	6.22E+02	8.50E+02	9.68E+02	3.01E+03	2.64E+03	3.02E+03	**2.16E+02**
	Std	1.91E+02	3.80E+01	8.70E+01	1.65E+02	9.46E+01	1.64E+02	7.52E+01	7.64E+01	1.12E+02	5.08E+01	**1.93E+01**
F8	Mean	4.81E+02	1.51E+02	5.68E+02	8.69E+02	4.61E+02	4.47E+02	5.20E+02	1.65E+03	1.50E+03	1.71E+03	**1.27E+02**
	Std	7.38E+01	2.27E+01	7.60E+01	1.04E+02	7.52E+01	1.13E+02	6.59E+01	6.70E+01	5.05E+01	3.18E+01	**1.42E+01**
F9	Mean	1.57E+04	3.77E+02	2.08E+04	3.12E+04	9.34E+03	1.32E+04	3.60E+04	5.94E+04	6.33E+04	7.10E+04	**7.64E+01**
	Std	3.65E+03	1.41E+02	1.22E+04	1.83E+04	3.07E+03	5.52E+03	1.31E+04	6.48E+03	3.10E+03	3.12E+03	**7.83E+01**
F10	Mean	1.41E+04	**7.77E+03**	1.30E+04	1.38E+04	1.22E+04	2.28E+04	1.33E+04	2.77E+04	2.61E+04	2.98E+04	1.53E+04
	Std	1.19E+03	**9.75E+02**	9.11E+02	1.29E+03	1.20E+03	7.53E+02	1.20E+03	2.01E+03	1.30E+03	4.11E+02	7.60E+03
F11	Mean	8.25E+02	2.17E+02	1.86E+03	1.37E+03	6.73E+02	4.89E+02	6.85E+02	1.52E+05	1.63E+05	1.92E+05	**2.02E+02**
	Std	2.28E+02	6.92E+01	2.73E+02	3.43E+02	1.26E+02	1.47E+02	9.14E+01	1.29E+04	2.63E+04	1.96E+04	**5.07E+01**
F12	Mean	1.54E+06	1.99E+05	8.53E+07	8.06E+08	4.81E+05	3.14E+05	8.40E+07	2.15E+11	1.10E+11	1.63E+11	**1.49E+05**
	Std	8.60E+05	7.47E+04	2.79E+07	1.51E+09	1.49E+05	1.93E+05	3.58E+07	1.38E+10	2.89E+10	9.89E+09	**5.58E+04**
F13	Mean	1.30E+04	5.65E+03	3.71E+04	3.14E+07	3.25E+03	5.65E+03	3.25E+05	5.17E+10	1.76E+10	3.53E+10	**2.29E+03**
	Std	7.06E+03	6.73E+03	6.07E+03	1.03E+08	2.15E+03	7.05E+03	5.50E+05	2.58E+09	6.66E+09	3.00E+09	**2.18E+03**
F14	Mean	2.66E+05	3.18E+02	3.02E+05	2.34E+06	1.79E+03	2.60E+04	2.02E+06	1.43E+08	1.99E+07	3.45E+07	**2.21E+02**
	Std	1.15E+05	1.76E+02	1.01E+05	1.69E+06	3.16E+03	1.86E+04	7.87E+05	8.20E+07	6.04E+06	4.75E+06	**4.17E+01**
F15	Mean	2.71E+03	2.51E+03	2.88E+04	2.30E+07	6.97E+02	1.47E+03	4.67E+04	2.90E+10	4.53E+09	1.76E+10	**4.46E+02**
	Std	2.22E+03	4.95E+03	8.26E+03	1.24E+08	5.71E+02	1.66E+03	1.00E+05	2.56E+09	1.60E+09	1.80E+09	**4.08E+02**
F16	Mean	3.67E+03	**2.07E+03**	4.03E+03	4.51E+03	3.17E+03	2.86E+03	4.08E+03	2.51E+04	1.26E+04	1.73E+04	2.18E+03
	Std	5.65E+02	**5.69E+02**	3.92E+02	5.49E+02	3.75E+02	6.72E+02	8.44E+02	2.53E+03	1.08E+03	9.55E+02	1.12E+03
F17	Mean	3.37E+03	**1.37E+03**	3.11E+03	4.80E+03	2.25E+03	2.13E+03	3.19E+03	1.84E+07	4.81E+04	9.01E+05	2.02E+03
	Std	5.35E+02	**6.32E+02**	3.34E+02	8.40E+02	2.99E+02	4.74E+02	4.80E+02	9.58E+06	5.48E+04	4.20E+05	5.64E+02
F18	Mean	5.48E+05	**6.97E+03**	5.16E+05	3.34E+06	4.55E+04	1.08E+05	2.31E+06	3.66E+08	2.64E+07	9.23E+07	8.41E+03
	Std	2.30E+05	**4.06E+03**	1.74E+05	2.33E+06	2.40E+04	5.52E+04	9.79E+05	1.33E+08	1.15E+07	2.61E+07	6.63E+03
F19	Mean	3.33E+03	5.45E+03	1.43E+06	4.93E+06	1.26E+03	1.57E+03	6.64E+04	3.10E+10	5.10E+09	1.56E+10	**6.32E+02**
	Std	4.08E+03	7.18E+03	8.38E+05	1.42E+07	8.58E+02	2.65E+03	9.83E+04	2.37E+09	2.28E+09	2.24E+09	**1.23E+03**
F20	Mean	2.96E+03	**1.30E+03**	2.94E+03	3.44E+03	1.93E+03	2.67E+03	2.82E+03	4.82E+03	4.46E+03	5.12E+03	2.05E+03
	Std	4.45E+02	**5.97E+02**	2.81E+02	4.13E+02	3.62E+02	4.22E+02	4.23E+02	1.69E+02	4.25E+02	1.72E+02	5.08E+02
F21	Mean	6.25E+02	3.66E+02	7.44E+02	1.36E+03	5.82E+02	6.47E+02	7.61E+02	2.50E+03	2.11E+03	2.40E+03	**3.41E+02**
	Std	6.73E+01	2.07E+01	8.48E+01	1.58E+02	3.28E+01	1.21E+02	7.16E+01	2.53E+02	9.34E+01	7.55E+01	**1.82E+01**
F22	Mean	1.58E+04	8.96E+03	1.45E+04	1.58E+04	1.40E+04	2.42E+04	1.52E+04	2.78E+04	2.79E+04	3.09E+04	**8.19E+03**
	Std	1.59E+03	9.64E+02	8.90E+02	1.35E+03	1.12E+03	7.84E+02	1.06E+03	6.78E+02	1.20E+03	4.91E+02	**3.60E+03**
F23	Mean	9.31E+02	6.39E+02	1.14E+03	1.79E+03	8.11E+02	8.53E+02	9.19E+02	4.08E+03	3.20E+03	3.41E+03	**6.30E+02**
	Std	6.91E+01	2.20E+01	8.66E+01	2.20E+02	3.95E+01	8.36E+01	4.78E+01	4.35E+02	1.83E+02	8.76E+01	**1.90E+01**
F24	Mean	1.28E+03	1.07E+03	1.46E+03	2.57E+03	1.33E+03	1.28E+03	1.43E+03	1.17E+04	5.24E+03	5.86E+03	**1.03E+03**
	Std	7.52E+01	3.68E+01	8.36E+01	3.38E+02	6.47E+01	9.31E+01	6.76E+01	8.02E+02	5.05E+02	2.75E+02	**2.45E+01**
F25	Mean	7.84E+02	**7.72E+02**	7.78E+02	9.60E+02	8.04E+02	7.89E+02	8.43E+02	2.28E+04	1.78E+04	2.27E+04	7.85E+02
	Std	5.77E+01	**6.93E+01**	3.73E+01	2.46E+02	5.23E+01	6.81E+01	7.18E+01	1.73E+03	1.81E+03	8.88E+02	5.75E+01
F26	Mean	8.50E+03	5.19E+03	6.75E+03	1.88E+04	9.76E+03	7.94E+03	9.11E+03	5.21E+04	3.58E+04	4.42E+04	**5.16E+03**
	Std	1.28E+03	3.73E+02	4.15E+03	2.73E+03	1.57E+03	9.02E+02	8.97E+02	2.51E+03	3.41E+03	1.43E+03	**3.16E+02**
F27	Mean	8.13E+02	7.14E+02	1.06E+03	1.35E+03	8.18E+02	9.14E+02	7.71E+02	1.44E+04	4.29E+03	7.48E+03	**7.06E+02**
	Std	6.54E+01	4.43E+01	7.06E+01	2.59E+02	6.29E+01	7.01E+01	4.60E+01	1.46E+03	6.38E+02	5.05E+02	**3.23E+01**
F28	Mean	5.61E+02	5.63E+02	6.24E+02	9.49E+02	5.59E+02	**5.49E+02**	1.43E+03	3.00E+04	2.24E+04	2.22E+04	5.54E+02
	Std	3.25E+01	2.59E+01	2.01E+01	3.44E+02	3.15E+01	**2.59E+01**	3.08E+03	2.66E+03	1.94E+03	9.65E+02	3.07E+01
F29	Mean	3.42E+03	1.59E+03	5.35E+03	4.78E+03	2.99E+03	2.58E+03	3.40E+03	1.06E+06	3.09E+04	1.64E+05	**1.45E+03**
	Std	5.81E+02	3.73E+02	5.92E+02	8.71E+02	5.08E+02	4.48E+02	4.93E+02	4.41E+05	1.52E+04	6.30E+04	**3.21E+02**
F30	Mean	2.72E+04	3.08E+03	2.22E+07	2.31E+07	6.16E+03	3.07E+03	3.08E+04	4.78E+10	9.34E+09	3.02E+10	**2.94E+03**
	Std	2.98E+04	1.74E+03	6.40E+06	5.31E+07	1.98E+03	1.49E+03	1.42E+04	2.24E+09	3.75E+09	2.79E+09	**1.08E+03**

functions and dimensions, this research selects ten representative functions from the CEC 2017 test suite for this experiment. These functions are the unimodal functions F1, F2, and F3, the multimodal functions F6 and F9, the hybrid functions F13, F15, and F19, and the composition functions F21 and F27. Each function is tested at $D = 30$, 50, and 100. Fig. 4 describes the detailed convergence results, and the following will be discussed.

For the convergence curves on the unimodal functions F1, F2, and F3, QAGO seems to have advantages in convergence speed and accuracy. Especially on F1 and F3, it significantly alleviates the sensitivity of GO to parameters, weakening the impact of shift and rotation

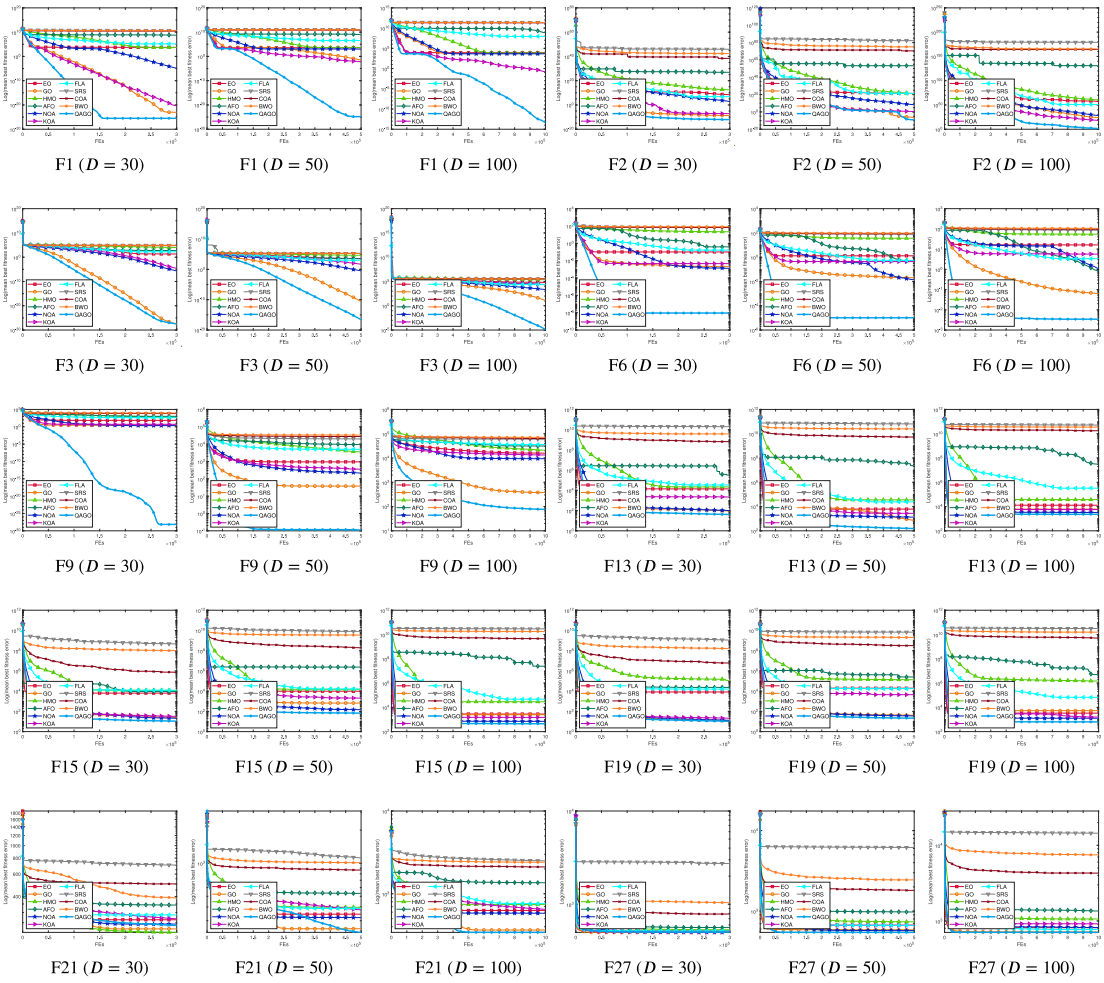


Fig. 4. Convergence curves of QAGO and novel metaheuristic algorithms on several representative functions (D = 30, 50, and 100).

characteristics on its search performance. Moreover, on F2, according to the vertical axis scale, the significance of QAGO's performance advantage is evident, especially as the problem dimension increases.

For the convergence curves on the multimodal function functions F6 and F9, QAGO achieves the fastest convergence speed and the best convergence accuracy on F6. As for other algorithms, the local optimal feature of the problem causes them to quickly fall into local optimum areas. Furthermore, only QAGO shows continuous convergence behavior on F9 when D = 30, while the convergence curve of QAGO still stands out among other algorithms at D = 50 and D = 100. This validates the effectiveness and competitiveness of QAGO's strategies, particularly in comparison to the performance of GO.

For the convergence curves on the hybrid functions F13, F15, and F19, the convergence curves of GO, NOA, and QAGO are similar, but the results of QAGO are relatively better. In addition, although the topologies of these functions are complex, the convergence curves of QAGO continue to decrease before the end of the iteration, indicating that it is still improving the quality of the current solution. Moreover, the convergence curves of other algorithms show similar convergence behavior and similar results.

For the convergence curves on the composition functions F21 and F27, QAGO does not show significant advantages on low-dimensional F21, but as the dimension increases, its performance seems to have improved compared to other algorithms. On F27, the results of GO, NOA, KOA, and QAGO are similar, but QAGO's relative advantage is more pronounced. Therefore, QAGO demonstrates good convergence on these issues.

To better verify the robustness and stability of the algorithms, we plotted box plots of each algorithm on representative functions, as shown in Fig. 7. These plots correspond to the convergence curves mentioned earlier. Based on the plots, it is apparent that the QAGO's data distribution is generally more concentrated, suggesting that its search function is both robust and stable. This can be attributed to its parameter adaptation mechanism, which continually adjusts the search state and lessens the effect of complicated problems on the search process. Additionally, although the data distributions of SRS, COA, and BWO are relatively stable, their box plot positions are high, indirectly reflecting that they may not have obvious advantages.

Fig. 5 shows the cumulative fitness errors of QAGO and the novel metaheuristic algorithms on 90 benchmark problems. Moreover, to show the comprehensive performance of the algorithm in solving problems, we also plot the average cumulative fitness of the algorithms on 90 problems and only show their logarithms in Fig. 6. As evident from the graphs, the proposed algorithm demonstrates a remarkable ability to balance its performance across various dimensions and problems. It has a competitive advantage compared to other competitors. Moreover, KOA, GO, and NOA can also solve these problems well. However, according to the NFL theorem, this situation may vary with changes in the problem.

The Wilcoxon rank-sum test was applied to compare the statistical significance of the proposed QAGO with each of the novel metaheuristic algorithms on each problem (Table 6). QAGO outperforms EO, GO, HMO, AFO, NOA, KOA, FLA, SRS, COA, and BWO on 28, 15, 26, 29, 20, 26, 29, 30, 30, and 30 problems when D = 30, respectively.

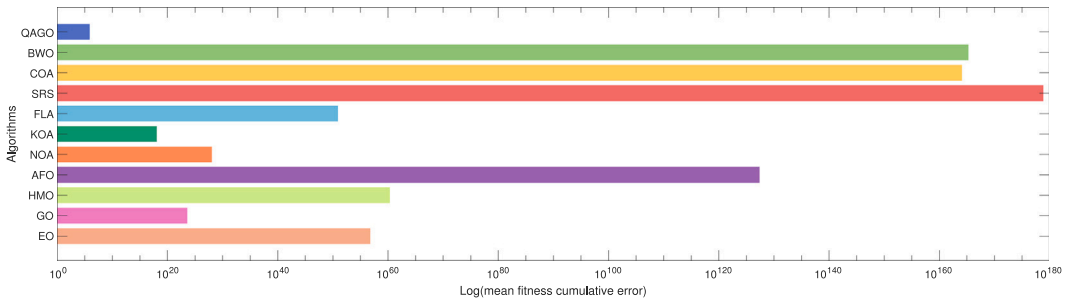


Fig. 5. The cumulative fitness error of QAGO and competitive algorithms on 90 benchmark problems (30 functions × 3 dimensions).

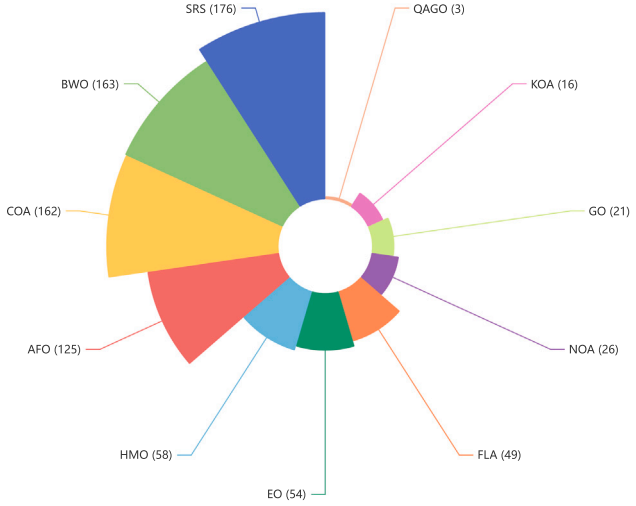


Fig. 6. The cumulative mean fitness logarithmic error of QAGO and competitive algorithms on 90 benchmark problems (30 functions × 3 dimensions).

Table 6
Significance achieved from the Wilcoxon rank-sum test comparing QAGO and novel metaheuristic algorithms.

Dimension	vs. QAGO	EO	GO	HMO	AFO	NOA	KOA	FLA	SRS	COA	BWO
$D = 30$	+	0	8	2	1	3	0	1	0	0	0
	≈	2	7	2	0	7	4	0	0	0	0
	−	28	15	26	29	20	26	29	30	30	30
$D = 50$	+	1	8	2	1	2	0	1	0	0	0
	≈	2	4	3	1	4	2	1	0	0	0
	−	27	18	25	28	24	28	28	30	30	30
$D = 100$	+	0	3	0	0	0	0	0	0	0	0
	≈	3	12	3	1	5	3	1	0	0	0
	−	27	15	27	29	25	27	29	30	30	30

QAGO outperforms the aforementioned algorithms on 27, 18, 25, 28, 24, 28, 28, 30, 30, and 30 problems when $D = 50$, respectively. QAGO outperforms the aforementioned algorithms on 27, 15, 27, 29, 25, 27, 29, 30, 30, and 30 problems when $D = 100$, respectively. Furthermore, the performance of QAGO is seldom inferior to or on par with other methods mentioned above.

The Friedman test and the Kruskal–Wallis test achieved the ranking of QAGO and other algorithms on the different dimensional problems, as shown in Table 7. According to two nonparametric statistical tests, it has been proven that QAGO outperforms other competitors across various dimensions. Moreover, Table 8 provides the numbers of the best and last positions of algorithms from the Kruskal–Wallis test. Obviously, QAGO achieved first place 13, 17, and 23 times in three distinct dimensions, respectively. Furthermore, there has not been a single instance of coming in last place. This once again confirms the effectiveness of QAGO.

4.4. Compare QAGO with the 50 classical algorithms

In the previous section, we compared several of the latest metaheuristic algorithms. To better demonstrate the optimization performance of QAGO, we selected 50 classical algorithms that have been proposed and widely used since the last century as the competing algorithms for QAGO. The parameter settings of all algorithms refer to relevant papers. The unified experimental setup is consistent with the previous section. In this section, only the statistical results based on the Kruskal–Wallis test and the Wilcoxon rank-sum test are shown.

Table 9 presents the statistical rankings of 50 algorithms in solving the CEC 2017 benchmark based on the Kruskal–Wallis test. It can be seen that QAGO achieved the first rank in all three dimensions. As a result, QAGO's adaptability allows it to cope with changes in dimensionality, resulting in better performance. Furthermore, Table 10 presents the number of times the compared algorithms were better, approximately the same, or worse than QAGO based on the Wilcoxon rank-sum test in different dimensions. The results indicate that QAGO is more competitive than all the compared algorithms.

4.5. Compare QAGO with advanced self-adaptive algorithms

In this section, the experiment further compared the optimization performance of QAGO with six advanced adaptive algorithms. These adaptive algorithms include self-adaptive differential evolution (SaDE) (Qin & Suganthan, 2005), composite differential evolution (CoDE) (Wang, Cai, & Zhang, 2011), adaptive differential evolution with optional external archive (JADE) (Zhang & Sanderson, 2009), success-history based parameter adaptation for differential evolution (SHADE) (Tanabe & Fukunaga, 2013), improved multi-operator differential evolution (IMODE) (Sallam, Elsayed, Chakrabortty, & Ryan, 2020), and adaptive inertia weight particle swarm optimization (AIW-PSO) (Nickabadi, Ebadzadeh, & Safabakhsh, 2011). The experiments were conducted under the following conditions: 30 problems from the CEC 2017 test suite, with dimensions $D = 30/50/100$, population size $N = 40$, maximum function evaluations $MaxFEs = 10,000 \times D$, and a total of 30 independent runs The remaining parameter settings of the algorithms were kept consistent with the relevant papers. The statistical results of this experiment are presented in Tables 11 and 12. Note that more detailed supplementary evidence is publicly available at https://github.com/tsingke/QAGO/tree/main/SupplementaryExperiments.

From Table 11, it can be observed that QAGO is slightly inferior in ranking compared to the top-ranked CoDE when $D = 30$, securing the second position. However, as the problem dimension increases, QAGO achieves the top statistical ranking among the adaptive algorithms for $D = 50$ and $D = 100$. Additionally, the "Mean rank" column indicates the average performance of each algorithm across the three different problem dimensions. Clearly, with a statistical ranking of 2.37, QAGO demonstrates a promising problem-solving capability

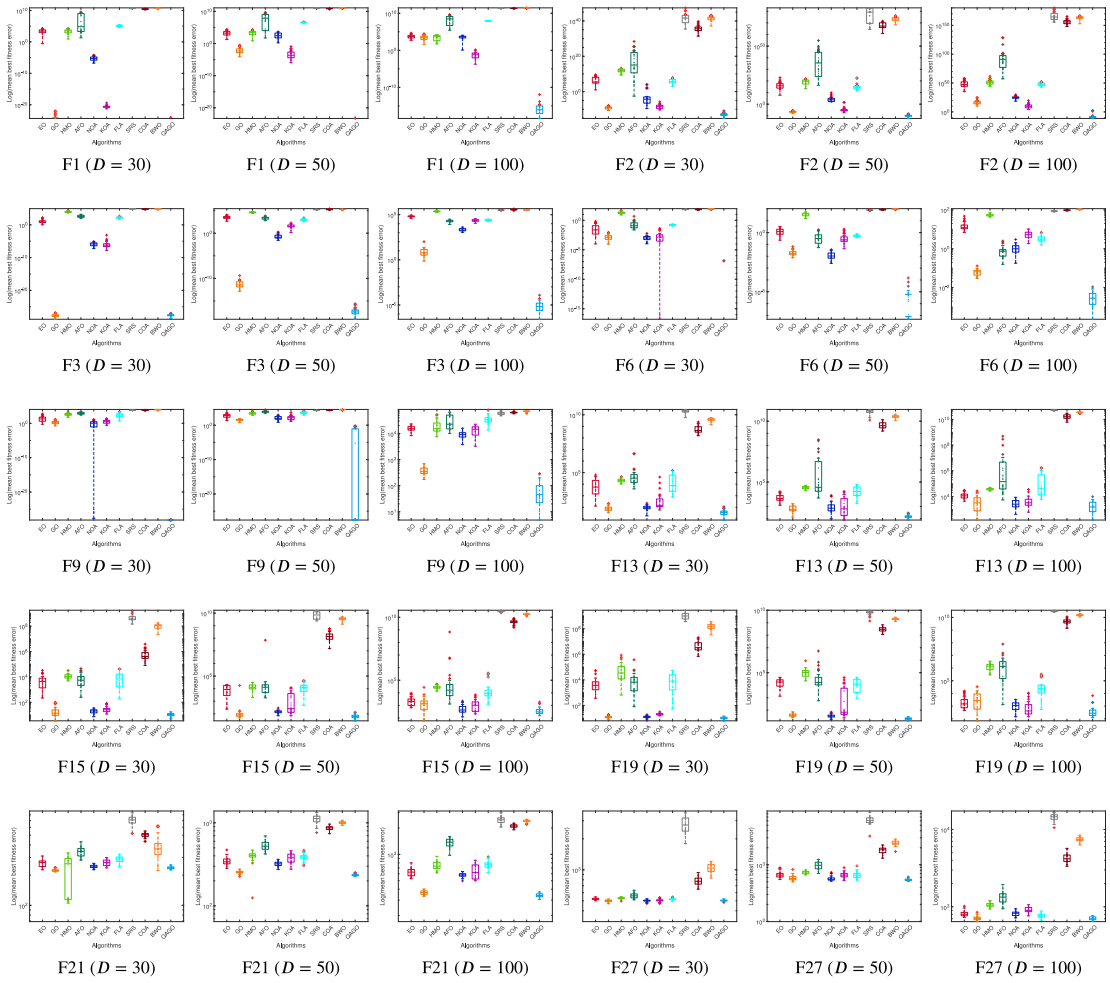


Fig. 7. Box plots of QAGO and novel metaheuristic algorithms on several representative functions (D = 30, 50, and 100).

Table 7
Ranking achieved from the Friedman test and the Kruskal–Wallis test comparing QAGO and novel metaheuristic algorithms.

Metric	Dimension	EO	GO	HMO	AFO	NOA	KOA	FLA	SRS	COA	BWO	QAGO
Friedman ranking	$D = 30$	5.00	2.17	5.93	7.57	2.60	4.33	6.37	10.77	9.07	10.13	**2.07**
	$D = 50$	5.47	1.93	5.83	7.50	3.17	4.43	6.00	10.70	9.10	10.20	**1.67**
	$D = 100$	5.37	2.27	6.00	7.43	3.60	3.83	6.00	10.50	9.23	10.27	**1.50**
Kruskal–Wallis ranking	$D = 30$	164.67	90.00	178.33	194.93	97.83	107.80	181.60	246.97	232.17	240.20	**86.00**
	$D = 50$	165.07	89.27	171.53	189.23	116.73	125.70	173.37	242.50	227.93	236.00	**83.17**
	$D = 100$	151.93	107.73	166.37	184.63	127.80	132.90	168.00	237.30	227.63	234.17	**82.03**

Table 8
Number of 1st and last mean rank of the algorithms from the Kruskal–Wallis test.

Metric	Rank	EO	GO	HMO	AFO	NOA	KOA	FLA	SRS	COA	BWO	QAGO
$D = 30$	1st	1	9	3	0	4	0	0	0	0	0	13
	last	0	0	0	0	0	0	0	25	0	5	0
$D = 50$	1st	0	10	3	0	0	0	0	0	0	0	17
	last	0	0	0	0	0	0	0	24	0	6	0
$D = 100$	1st	0	6	0	0	0	1	0	0	0	0	23
	last	0	0	0	0	0	4	0	20	1	9	0

compared to other adaptive algorithms. Furthermore, Table 12 presents the statistical differences in results between QAGO and other adaptive algorithms. From the table, it can be observed that CoDE outperforms QAGO on 10, 7, and 4 problems for D = 30, D = 50, and D = 100, respectively. However, other algorithms only surpass QAGO on a few problems. As the problem dimension increases, the problems become more complex, which can lead to performance degradation for algorithms. However, the number of times QAGO outperforms other adaptive algorithms also increases with this trend. Therefore, based on the above discussion, QAGO can still demonstrate its competitive performance compared to advanced adaptive algorithms.

Table 9
Statistical ranking of QAGO and 50 classical algorithms based on the Kruskal–Wallis test.

Position	Algorithms	$D = 30$	$D = 50$	$D = 100$	Mean rank
1	QAGO	323.80	330.33	339.23	331.12
2	GSK (Mohamed, Hadi, & Mohamed, 2020)	379.03	479.57	546.40	468.33
3	MPA (Faramarzi, Heidarinejad, Mirjalili, & Gandomi, 2020)	426.93	488.23	620.37	511.84
4	SFS (Salimi, 2015)	449.17	576.27	625.63	550.36
5	DE (Storn & Price, 1997)	578.83	639.27	619.27	612.46
6	INFO (Ahmadianfar et al., 2022)	595.00	663.50	663.50	640.67
7	ASO (Zhao, Wang, & Zhang, 2019a)	660.67	654.20	607.90	640.92
8	AEFA (Yadav et al., 2019)	722.90	632.80	573.90	643.20
9	SDO (Zhao, Wang, & Zhang, 2019b)	639.37	672.33	657.67	656.46
10	DS (Civicioglu, 2012)	623.83	684.80	698.73	669.12
11	CS (Yang & Deb, 2009)	623.30	669.27	729.70	674.09
12	COA (Pierezan & Coelho, 2018)	669.87	732.33	634.97	679.06
13	AEO (Zhao, Wang, & Zhang, 2020)	683.40	709.83	655.47	682.90
14	EO (Faramarzi, Heidarinejad, Stephens, & Mirjalili, 2020)	703.00	711.00	664.27	692.76
15	FA (Yang, 2009)	704.00	711.73	663.27	693.00
16	FPA (Yang, 2012)	694.63	694.10	698.00	695.58
17	HBO (Askari, Saeed, & Younas, 2020)	717.87	717.13	690.03	708.34
18	TLBO (Rao, Savsani, & Vakharia, 2011)	706.60	711.83	713.47	710.63
19	BBO (Simon, 2008)	731.07	729.37	684.30	714.91
20	MRFO (Zhao, Zhang, & Wang, 2020)	724.20	765.17	695.20	728.19
21	PSO (Kennedy & Eberhart, 1995)	761.90	722.67	709.30	731.29
22	MVO (Mirjalili, Mirjalili, & Hatamlou, 2016)	758.60	735.07	701.17	731.61
23	SO (Hashim & Hussien, 2022)	759.20	747.90	691.10	732.73
24	SSA (Mirjalili et al., 2017)	743.80	742.77	717.60	734.72
25	HBA (Hashim, Houssein, Hussain, Mabrouk, & Al-Atabany, 2022)	822.37	733.60	684.67	746.88
26	SA (Kirkpatrick et al., 1983)	753.27	728.23	790.20	757.23
27	HS (Geem, Kim, & Loganathan, 2001)	758.53	746.07	786.37	763.66
28	CMA-ES (Awad et al., 2017)	863.37	760.07	696.73	773.39
29	RUN (Ahmadianfar, Heidari, Gandomi, Chu, & Chen, 2021)	760.83	774.20	787.80	774.28
30	GSA (Rashedi, Nezamabadi-Pour, & Saryazdi, 2009)	825.93	784.87	713.93	774.91
31	SFLA (Eusuff, Lansey, & Pasha, 2006)	637.80	665.23	1030.47	777.83
32	GA (Zames, 1981)	805.50	793.10	735.00	777.87
33	CHIO (Al-Betar, Alyasseri, Awadallah, & Abu Doush, 2021)	801.17	778.50	762.67	780.78
34	ABC (Karaboga & Basturk, 2007)	787.20	765.67	792.00	781.62
35	WSO (Braik, Hammouri, Atwan, Al-Betar, & Awadallah, 2022)	762.53	807.23	776.00	781.92
36	DMOA (Agushaka, Ezugwu, & Abualigah, 2022)	792.50	767.00	791.30	783.60
37	ICA (Atashpaz-Gargari & Lucas, 2007)	836.73	790.87	735.10	787.57
38	LSA (Shareef, Ibrahim, & Mutlag, 2015)	789.83	811.93	768.77	790.18
39	ACO (Dorigo et al., 2006)	823.53	807.30	871.73	834.19
40	BSO (Shi, 2011)	853.70	831.90	836.57	840.72
41	GWO (Mirjalili, Mirjalili, & Lewis, 2014)	878.33	869.53	883.07	876.98
42	HHO (Heidari et al., 2019)	948.00	895.43	872.70	905.38
43	WOA (Mirjalili & Lewis, 2016)	970.50	909.93	893.97	924.80
44	CA (Reynolds, 1994)	955.33	947.50	985.13	962.65
45	Jaya (Rao, 2016)	954.40	959.17	1003.47	972.35
46	AFSA (Neshat, Sepidnam, Sargolzaei, & Toosi, 2014)	986.03	970.23	975.37	977.21
47	MFO (Mirjalili, 2015)	995.30	981.37	990.40	989.02
48	SCA (Mirjalili, 2016)	1013.63	1012.23	1025.23	1017.03
49	AOA (Abualigah, Diabat, Mirjalili, Abd Elaziz, & Gandomi, 2021)	1061.97	1057.07	1084.93	1067.99
50	BFO (Das, Biswas, Dasgupta, & Abraham, 2009)	1116.67	1073.40	1071.03	1087.03
51	BOA (Arora & Singh, 2019)	1104.57	1097.40	1095.47	1099.15

4.6. Compare QAGO with the CEC competition winners

To further verify the performance of QAGO, this paper used the winners from the CEC competition as competitors for the QAGO algorithm and conducted experiments on the latest CEC 2022 test suite. The winners involved are L-SHADE, L-SHADE-EpSin, UMOEAs-II, EBOwithCMAR, and HS-ES, with parameter settings referenced in Table 2.

From Table 13, it can be seen that the performance of QAGO is similar to that of winners. They can find the global optimal solution for F1, F3, F5, and F11. Furthermore, QAGO exhibits competitive performance compared to the winners in all twelve functions. The symbol "+/ ≈ /−" is recorded at the end of Table 13. The performance of HS-ES is the worst of all the winners. It does not perform as well as QAGO on these six functions but only outperforms QAGO on two functions. The performances of L-SHADE, L-SHADE-EpSin, UMOEAs-II, and EBOwithCMAR are similar to those of QAGO in about half the number of functions, but they outperform QAGO in only a few functions. Therefore, based on the above discussion, QAGO still demonstrates a certain level of competitiveness in terms of achieving solution quality when compared to the winners.

4.7. Convergence analysis of QAGO

This subsection will demonstrate the search behavior and effectiveness of QAGO in solving problems through the six qualitative metrics shown in Fig. 8. The first metric is the three-dimensional topology of the representative function, which shows the complexity of the problem. These functions are from the CEC 2017 test suite, where F1

Table 10
Statistical result of QAGO and 50 classical algorithms based on the Wilcoxon rank-sum test.

vs. QAGO	$D = 30$	$D = 50$	$D = 100$
GSK	7/11/12	3/4/23	1/5/24
MPA	3/7/20	3/2/25	0/3/27
SFS	2/4/24	2/0/28	0/4/26
DE	4/5/21	1/3/26	0/2/28
INFO	0/1/29	0/1/29	0/3/27
ASO	2/4/24	3/2/25	0/4/26
AEFA	7/1/22	7/6/17	8/2/20
SDO	0/1/29	1/1/28	0/1/29
DS	2/1/27	3/0/27	0/2/28
CS	2/1/27	3/1/26	1/1/28
COA	1/0/29	1/2/27	0/3/27
AEO	1/0/29	1/3/26	0/2/28
EO	0/2/28	1/2/27	0/3/27
FA	2/0/28	1/1/28	0/2/28
FPA	0/1/29	0/2/28	0/1/29
HBO	0/1/29	1/2/27	1/3/26
TLBO	0/0/30	0/0/30	0/1/29
BBO	0/1/29	1/1/28	0/1/29
MRFO	0/2/28	1/3/26	0/4/26
PSO	0/2/28	1/3/26	0/1/29
MVO	2/0/28	1/3/26	1/2/27
SO	1/1/28	2/2/26	1/2/27
SSA	0/2/28	1/3/26	0/2/28
HBA	0/1/29	1/1/28	0/2/28
SA	3/4/23	3/1/26	2/0/28
HS	4/1/25	0/0/30	0/0/30
CMA-ES	3/5/22	6/5/19	12/2/16
RUN	0/2/28	1/1/28	0/1/29
GSA	1/2/27	3/1/26	0/4/26
SFLA	1/2/27	0/3/27	0/0/30
GA	3/1/26	4/1/25	1/3/26
CHIO	2/1/27	1/1/28	0/2/28
ABC	0/1/29	1/1/28	0/1/29
WSO	1/1/28	1/1/28	0/2/28
DMOA	0/1/29	0/3/27	0/1/29
ICA	1/1/28	2/1/27	0/3/27
LSA	0/0/30	0/1/29	0/1/29
ACO	0/0/30	0/2/28	1/0/29
BSO	0/0/30	1/2/27	0/3/27
GWO	1/0/29	1/1/28	0/2/28
HHO	0/0/30	0/1/29	0/1/29
WOA	0/0/30	0/0/30	0/1/29
CA	1/0/29	1/2/27	0/1/29
Jaya	0/0/30	0/0/30	0/0/30
AFSA	0/1/29	1/1/28	0/1/29
MFO	0/0/30	0/1/29	0/1/29
SCA	0/0/30	0/0/30	0/0/30
AOA	0/0/30	0/0/30	0/0/30
BFO	0/0/30	0/0/30	0/1/29
BOA	0/0/30	0/0/30	0/0/30

Table 11
Statistical ranking of QAGO and 6 advanced self-adaptive algorithms based on the Friedman test.

Placement	Algorithm	$D = 30$	$D = 50$	$D = 100$	Mean rank
1	QAGO	2.67	2.10	2.33	2.37
2	CoDE	2.57	2.60	2.70	2.62
3	JADE	3.43	3.57	3.60	3.53
4	SHADE	3.90	3.97	4.10	3.99
5	SaDE	4.10	4.57	4.43	4.37
6	IMODE	4.77	5.00	4.87	4.88
7	AIWPSO	6.57	6.20	5.97	6.25

Table 12
Statistical result of QAGO and 6 advanced self-adaptive algorithms based on the Wilcoxon rank-sum test.

vs. QAGO	$D = 30$	$D = 50$	$D = 100$
CoDE	10/14/6	7/12/11	4/12/14
JADE	4/13/13	4/8/18	4/11/15
SHADE	2/10/18	4/4/22	1/7/22
SaDE	3/9/18	3/5/22	1/7/22
IMODE	3/1/26	3/3/24	4/5/21
AIWPSO	1/5/24	3/3/24	2/3/25

is a unimodal function, F4, F6, and F9 are multimodal functions, and F22 and F28 are composition functions.

The second metric is the search trajectory of an individual, which shows the searching behavior of the individual during the iterative evolution. On F1, F4, F6, and F9, the individual continuously approaches

Table 13
Mean and standard deviation of the fitness error for the CEC 2022 test suite on 30 runs (the best entries are in bold).

No.	Metric	L-SHADE (CEC 2014 winner)	L-SHADE-EpSin (CEC 2016 winner)	UMOEAs-II (CEC 2016 winner)	EBOwithCMAR (CEC 2017 winner)	HS-ES (CEC 2018 winner)	QAGO (the proposed algorithm)
CEC 2022 F1	Mean	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**
	Std	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**
CEC 2022 F2	Mean	6.45E+00	6.45E+00	2.39E+00	3.99E−01	**0.00E+00**	7.57E−03
	Std	2.60E+00	2.60E+00	2.06E+00	1.26E+00	**0.00E+00**	9.23E−03
CEC 2022 F3	Mean	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**
	Std	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**
CEC 2022 F4	Mean	2.69E+00	1.69E+00	2.79E+00	**0.00E+00**	4.97E−01	5.32E+00
	Std	9.44E−01	8.19E−01	9.14E−01	**0.00E+00**	5.24E−01	1.47E+00
CEC 2022 F5	Mean	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**
	Std	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**
CEC 2022 F6	Mean	**3.22E−01**	3.62E−01	4.02E−01	3.69E−01	1.10E+00	4.47E−01
	Std	**1.69E−01**	1.80E−01	7.40E−02	4.80E−01	1.23E+00	1.31E−01
CEC 2022 F7	Mean	2.17E−03	**2.29E−05**	5.26E−02	4.86E−05	2.58E+00	1.38E+00
	Std	5.22E−03	**2.57E−05**	6.87E−02	5.00E−05	6.12E+00	1.10E+00
CEC 2022 F8	Mean	1.62E+00	3.73E+00	2.63E+00	**1.78E−01**	2.08E+01	3.65E+00
	Std	1.85E+00	6.98E+00	2.99E+00	**2.31E−01**	3.50E−01	3.15E+00
CEC 2022 F9	Mean	**2.29E+02**	**2.29E+02**	**2.29E+02**	**2.29E+02**	**2.29E+02**	**2.29E+02**
	Std	**5.52E−14**	**5.76E−14**	**3.67E−14**	**5.27E−14**	**1.91E−04**	**3.27E−09**
CEC 2022 F10	Mean	**1.00E+02**	**1.00E+02**	**1.00E+02**	**1.00E+02**	1.36E+02	**1.00E+02**
	Std	**3.38E−02**	**5.25E−02**	**2.29E−02**	**2.13E−02**	4.75E+01	**3.75E−02**
CEC 2022 F11	Mean	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	6.00E+01	**0.00E+00**
	Std	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	6.00E+01	**0.00E+00**
CEC 2022 F12	Mean	**1.63E+02**	1.65E+02	**1.63E+02**	1.64E+02	1.67E+02	**1.63E+02**
	Std	**1.69E+00**	4.66E−01	**1.57E+00**	1.08E+00	5.54E−01	**7.91E−01**
	+/ ≈ /−	4/7/1	3/6/3	4/7/1	3/7/2	2/4/6	–

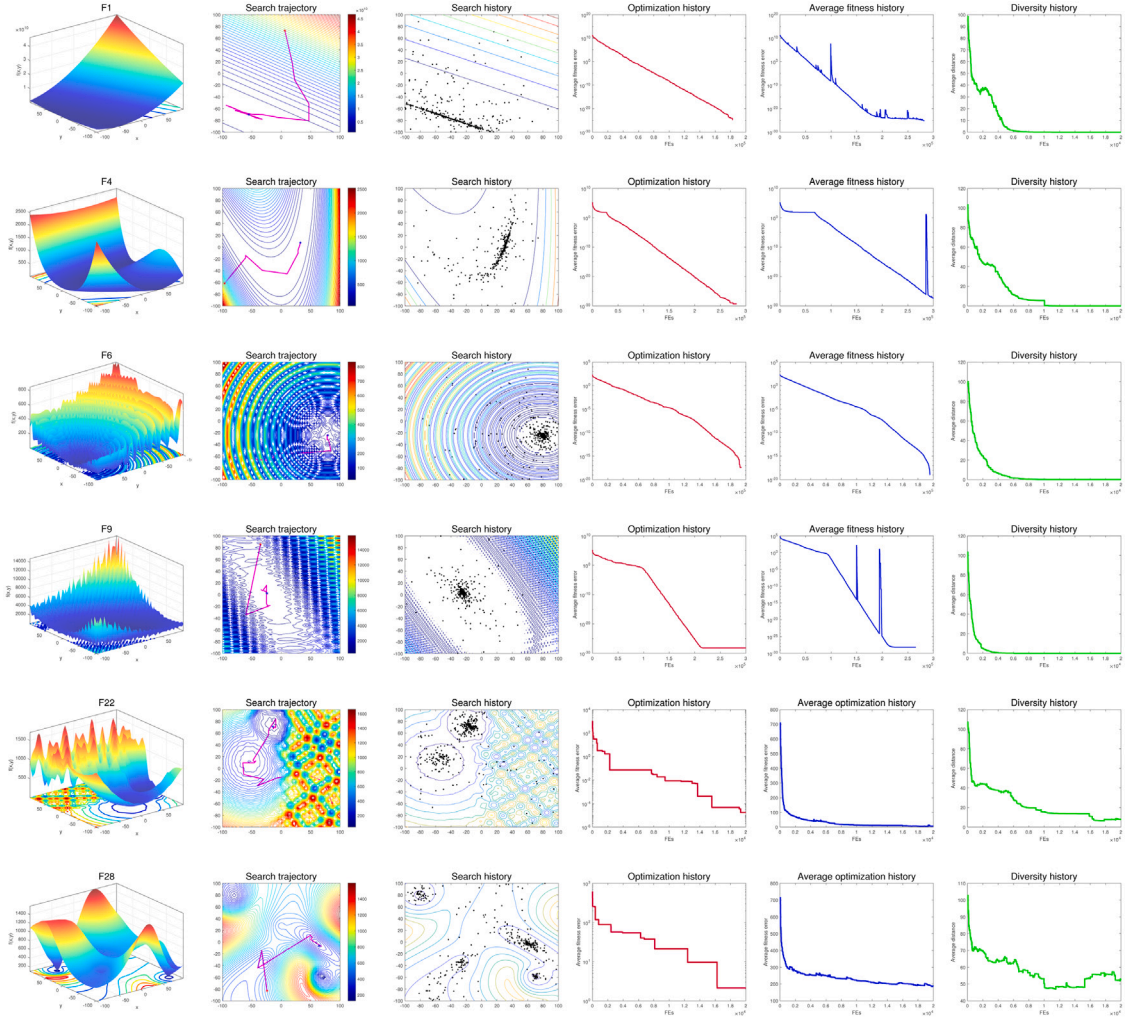


Fig. 8. Convergence analysis indicators: 3D topology of functions, search trajectory, search history, optimization history, average fitness history, and diversity history.

the global optimum and explores the local area around it. On F22 and F28, the individual is trapped in the local optimum and performs an escape behavior, which proves that the algorithm has the ability to prevent getting stuck in the local optimal region.

The third metric is the search history, which displays the evolutionary history of the entire population. On F1, F4, F6, F9, and F22, all individuals are updated towards the global optimal area and engage in a lot of local development activities near the global optimal point, demonstrating the QAGO algorithm's excellent convergence behavior. On F28, a large number of individuals are generated in the local optimum area, increasing the risk of the algorithm being trapped in the local optimum. However, the novel operators assist individuals

Table 14
Algorithm complexity of QAGO.

Dimension	$T0$	$T1$	$\overline{T2}$	$(\overline{T2}-T1)/T0$
$D = 30$	3.07E−02	1.18E+00	8.73E+00	2.46E+02
$D = 50$	3.07E−02	1.51E+00	9.24E+00	2.52E+02
$D = 100$	3.07E−02	3.64E+00	1.43E+01	3.47E+02

in escaping the local optimum and heading towards new, promising regions by incorporating more evolutionary information.

The fourth metric is the optimization history, which depicts the adaptability change of the best individual in the population. The downward trend of all optimization history curves indicates that the algorithm has good convergence capability. Moreover, on these functions, QAGO seems not to be troubled by the problem of local optima.

The fifth metric is the average fitness history, which represents the average fitness change of all individuals in the entire population. On F6, F22, and F28, the quality of the entire population is constantly improving, and there is no phenomenon of degradation. However, on F1, F4, and F9, the sudden shaking of the curve indicates that the quality of some individuals deteriorates rapidly, which is the result of the algorithm's forced update. This behavior hopes that individuals will explore more promising areas without being influenced by fitness.

The sixth metric is the diversity history, which represents the variation of distances between individuals. All curves continually decrease at the beginning, indicating a decrease in population diversity. However, as the iteration progresses, individuals generate a large number of individuals in promising areas to conduct exploitation, which slows the decay rate of the diversity curve.

4.8. Complexity analysis of QAGO

In terms of computational complexity, matrix-based operations can significantly improve the calculation speed of algorithms (Zhan et al., 2021). However, for practical problems, the objective function usually does not have the ability to handle matrices. Therefore, we do not rely on vectorized processing similar to that in Matlab for description. The computation criterion for QAGO algorithm complexity is described in Wu et al. (2017). This algorithm is implemented in MATLAB r2020b using a non-vectorized programming approach. The implementation runs on an Intel(R) Core(TM) i7-10700 CPU @ 2.90 GHz, with 16 GB RAM, and an X64 processor. Three complexity-related parameters $T0$, $T1$, and $T2$ are evaluated. $T0$ represents the runtime of the test program in problem definitions and evaluation criteria. $T1$ is the time to perform QAGO with 200,000 objective function evaluations of the function F18 in D dimensions. $T2$ is a complete computation time for 200,000 objective function evaluations of the same D dimension function F18, and this step is repeated five times to obtain $\overline{T2} = \text{mean}(T2)$. Finally, $T0$, $T1$, $\overline{T2}$, and $(\overline{T2}-T1)/T0$ are shown in Table 14.

Furthermore, the big O notation is also used to describe the computational complexity of the QAGO algorithm. Assuming that there are N individuals in the population, and the dimension of each individual is D. First, the time complexity of initializing the population is $O(N \times D)$. Second, the computational complexity of the sorting algorithm is $O(N \log(N))$. Third, the computational complexity of the improved learning phase is $O(N)$. Fourth, the computational complexity of the improved reflection phase is $O(N \times D)$ in the most extreme case. In addition, the evaluation of the objective function requires a total of $O(2N)$. Therefore, in the most extreme case, the overall computational complexity of QAGO is: $O(N \times D) + O(N \log(N)) + O(N) + O(N \times D) + O(2N) \approx O(N \times D + N \log(N))$.

It is worth mentioning that this is consistent with the computational complexity of GO. However, in practical computations, QAGO may incur slightly higher time costs compared to GO as it involves sampling more individuals and additional adaptive computations.

5. Application I: Multilevel threshold image segmentation

Image segmentation is an important basic task in image processing (Rather & Bala, 2021). It refers to dividing an image into several sub-regions, where each sub-region contains pixels with similar features or attributes and there are significant differences in pixel features or attributes between different sub-regions. Common image segmentation methods include threshold segmentation, region growth, edge detection, and clustering method (Houssein, Emam, & Ali, 2021). Due to the ease of use of threshold-based methods (Singh, Mittal, Nayyar, Singh, & Singh, 2023), we use a threshold-based method in this section to perform multi-level threshold image segmentation tasks. Threshold-based image segmentation methods are usually divided into two types: bi-level and multi-level. Multi-level threshold methods are more effective than bi-level threshold methods because when an image contains multiple homogeneous regions, bi-level threshold methods cannot separate them (Abdel-Basset, Chang, & Mohamed, 2021; Dinkar, Deep, Mirjalili, & Thapliyal, 2021).

5.1. Kapur's entropy method

To find a series of optimal thresholds, Kapur's entropy method (Kapur, Sahoo, & Wong, 1985) is used as the maximization objective function. It is based on the principle of information entropy, which determines multiple thresholds by calculating the distribution of image grayscale values and dividing the image into multiple regions (Bhandari, Singh, Kumar, & Singh, 2014; Sathya, Kalyani, & Sakthivel, 2021). For an 8-bit grayscale image, the threshold (t) takes values between 0 and 255, and the number of thresholds (K) takes values between 1 and 255. When the threshold number is $K-1$, the image is segmented into $S_1, S_2, S_3, \ldots, S_K$ parts. Eq. (28) provides the mathematical model of Kapur's objective function:

$$F\left(t_1, t_2, \ldots, t_{K-1}\right) = S_1 + S_2 + S_3 + \cdots + S_K \tag{28}$$

$$S_1 = -\sum_{i=0}^{t_1-1} \frac{X_i}{T_1} \cdot \ln \frac{X_i}{T_1}, \quad T_1 = \sum_{i=0}^{t_1-1} X_i, X_i = \frac{N_i}{T_1} \tag{29}$$

$$S_2 = -\sum_{i=t_1}^{t_2-1} \frac{X_i}{T_2} \cdot \ln \frac{X_i}{T_2}, \quad T_2 = \sum_{i=t_1}^{t_2-1} X_i, X_i = \frac{N_i}{T_2} \tag{30}$$

$$S_3 = -\sum_{i=t_2}^{t_3-1} \frac{X_i}{T_3} \cdot \ln \frac{X_i}{T_3}, \quad T_3 = \sum_{i=t_2}^{t_3-1} X_i, X_i = \frac{N_i}{T_3} \tag{31}$$

$$S_K = -\sum_{i=t_{K-1}}^{255} \frac{X_i}{T_K} \cdot \ln \frac{X_i}{T_K}, \quad T_K = \sum_{i=t_{K-1}}^{255} X_i, X_i = \frac{N_i}{T_K} \tag{32}$$

where N_i represents the number of pixels owned by the ith grayscale value, T_K represents the total number of pixels in the Kth part image, and S_K denotes the objective function used to evaluate the Kth part image.

5.2. Comparison experiment of QAGO, GO, and winners

To verify the practicality of QAGO proposed in this article, we selected four images from the USC-SIPI image database as benchmarks, and Fig. 9 shows the corresponding test images and their grayscale histograms. GO, L-SHADE, L-SHADE-EpSin, UMOEAs-II, EBOwithCMAR, and HS-ES were used as comparative algorithms for QAGO in the summary of numerical optimization experiments, with their parameter settings consistent with those described in their respective papers. Each image was segmented under four different threshold numbers ($K = 5$, 10, 15, and 20), and for each segmentation test, the algorithm had to be independently run 30 times. The termination criteria for each run of each algorithm was set to $MaxFEs = 4,500$. The test results were summarized in Tables 15 and 16, with the relevant data including the average fitness value (F_{ave}), standard deviation (Std), average running

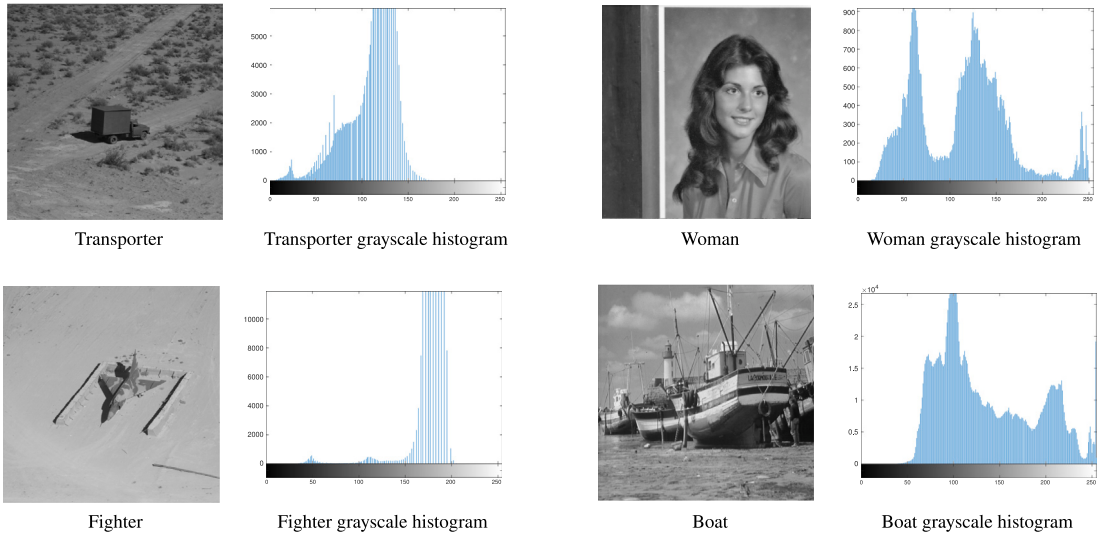


Fig. 9. Test images and corresponding grayscale histograms.

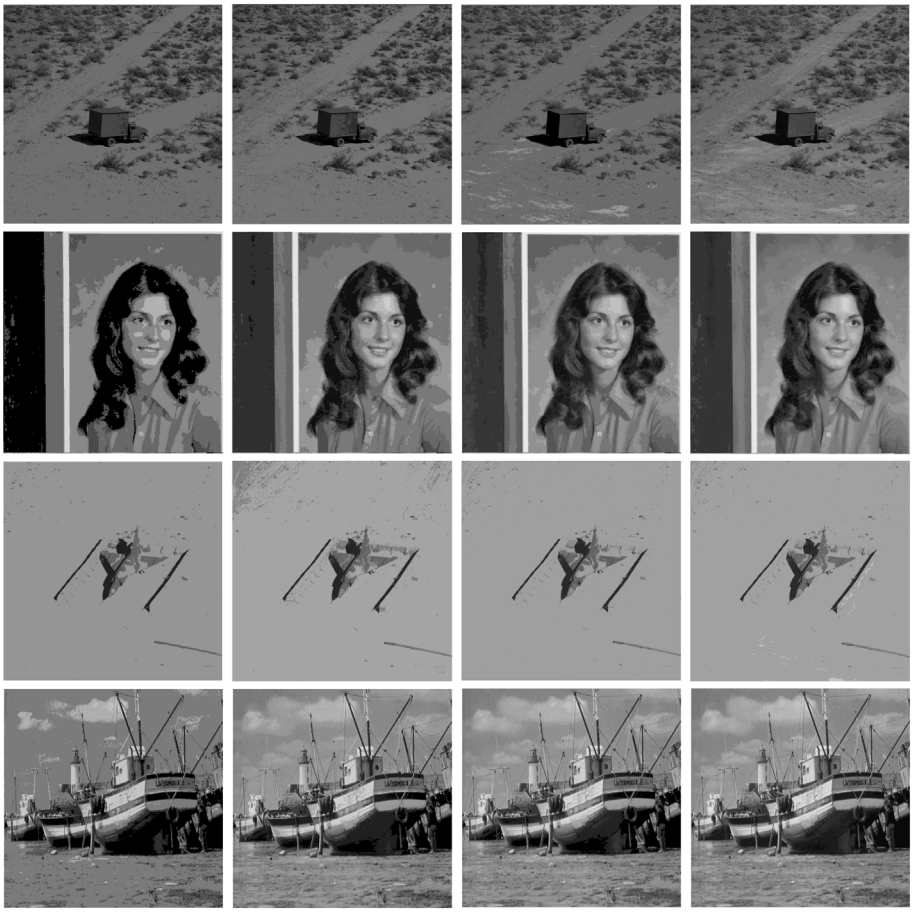


Fig. 10. The image segmentation results were obtained through QAGO.

time (T_{ave}), peak signal-to-noise ratio (PSNR), mean squared error (MSE), maximum absolute error (MAE), structured similarity index metric (SSIM), and feature similarity index measure (FSIM).

Figs. 10 and 11 illustrate the image segmentation results of QAGO using different thresholds. It is evident that as the number of thresholds increases, the segmented image produced by QAGO exhibits more detail. Tables 15 and 16 summarize nine indicators used to evaluate algorithm performance, and all algorithms involved in the test can effectively solve this problem. The tables reveal that the algorithms exhibit a minor performance disparity when dealing with small threshold numbers. However, as the number of thresholds increases, the performance gap between the algorithms gradually widens. Compared

Table 15
Comparison of the performance of the algorithms on the transporter image and the woman image.

Test image	K	Algorithms	F_{ave}	Std	T_{ave}	PSNR	MSE	MAE	SSIM	FSIM
Transporter	5	GO	17.28	0.02	1.89	22.78	342.94	144	0.86	0.92
		L-SHADE	17.28	0.02	**1.29**	23.32	302.68	142	0.86	0.93
		L-SHADE-EpSin	17.26	0.05	1.71	21.93	417.28	147	0.85	0.91
		UMOEAs-II	17.33	0.02	1.41	**24.10**	252.81	137	0.88	0.93
		HS-ES	17.25	0.03	1.52	23.51	289.56	138	0.87	0.93
		QAGO	**17.35**	0.15	1.47	23.95	**262.14**	**90**	**0.89**	**0.94**
	10	GO	27.53	0.10	2.42	20.17	625.24	50	0.80	0.87
		L-SHADE	26.56	0.16	**1.80**	25.28	192.67	**31**	0.91	0.95
		L-SHADE-EpSin	26.82	0.14	2.35	21.54	456.15	91	0.83	0.90
		UMOEAs-II	**27.75**	**0.08**	1.79	23.19	312.03	42	0.87	0.92
		EBOwithCMAR	27.57	0.16	1.82	21.64	446.18	91	0.84	0.90
		HS-ES	27.41	0.18	1.87	25.02	204.52	46	0.91	0.94
		QAGO	25.74	0.23	2.40	**26.56**	**143.68**	102	**0.94**	**0.97**
	15	GO	35.25	**0.18**	2.96	25.34	190.02	48	0.91	0.95
		L-SHADE	33.74	0.29	**2.34**	26.70	138.91	34	0.92	0.96
		L-SHADE-EpSin	34.45	0.55	4.19	27.99	103.22	34	0.94	0.97
		UMOEAs-II	**35.52**	0.22	2.41	25.49	183.67	39	0.91	0.95
		EBOwithCMAR	35.23	0.28	2.46	23.47	292.68	43	0.88	0.93
		HS-ES	33.02	0.73	2.90	**28.41**	**93.78**	**28**	**0.96**	**0.98**
		QAGO	35.39	0.28	2.45	27.78	108.29	38	0.93	0.96
	20	GO	42.10	0.33	3.53	29.24	77.50	34	0.95	0.97
		L-SHADE	39.36	0.75	2.85	27.44	117.32	36	0.94	0.97
		L-SHADE-EpSin	40.59	0.55	3.90	27.79	108.05	29	0.95	**0.98**
		UMOEAs-II	41.03	0.52	**2.77**	27.34	120.05	22	0.94	0.97
		EBOwithCMAR	41.12	0.53	2.85	29.83	67.56	31	**0.97**	**0.98**
		HS-ES	38.78	0.58	3.64	26.27	153.37	24	0.93	0.95
		QAGO	**42.11**	**0.31**	2.84	**30.00**	**64.97**	**21**	**0.97**	**0.98**
Woman	5	GO	20.75	0.03	1.66	17.05	1282.85	71	0.56	0.76
		L-SHADE	20.62	0.07	1.40	**20.26**	**612.41**	82	**0.75**	**0.79**
		L-SHADE-EpSin	20.75	0.05	1.65	16.87	1336.83	**70**	0.55	0.75
		UMOEAs-II	20.80	**0.01**	1.39	16.94	1314.52	73	0.56	0.77
		EBOwithCMAR	20.80	**0.01**	1.35	17.01	1294.01	74	0.56	0.77
		HS-ES	20.70	0.02	1.51	17.07	1276.39	73	0.56	0.77
		QAGO	**20.86**	**0.01**	**1.32**	17.10	1267.20	**70**	0.56	0.77
	10	GO	32.53	0.12	2.25	24.37	237.48	32	**0.83**	0.87
		L-SHADE	31.96	0.26	1.95	24.26	244.04	41	0.81	0.86
		L-SHADE-EpSin	32.26	0.18	2.46	24.68	221.57	32	0.81	**0.88**
		UMOEAs-II	**32.71**	0.13	1.85	24.78	216.23	34	**0.83**	**0.88**
		EBOwithCMAR	32.67	0.16	**1.81**	**24.90**	**210.45**	**31**	**0.83**	**0.88**
		HS-ES	31.94	0.18	2.07	22.42	372.16	55	0.73	0.85
		QAGO	32.59	**0.10**	1.91	24.38	237.07	37	0.80	0.86
	15	GO	41.68	0.19	2.42	27.78	108.39	25	0.85	0.92
		L-SHADE	40.46	0.36	2.33	25.96	165.00	35	0.84	0.89
		L-SHADE-EpSin	40.92	0.26	3.20	27.20	123.77	27	0.88	0.92
		UMOEAs-II	41.64	0.32	2.38	**28.84**	**84.97**	30	0.90	0.92
		EBOwithCMAR	41.57	0.37	**2.29**	28.40	93.96	**23**	0.89	0.93
		HS-ES	40.49	0.25	2.68	28.83	85.15	45	**0.92**	**0.93**
		QAGO	**41.87**	**0.18**	3.06	28.64	88.94	26	0.89	**0.93**
	20	GO	49.00	0.29	2.88	29.08	80.34	24	0.89	0.93
		L-SHADE	47.36	0.43	2.87	27.08	127.36	30	0.89	0.91
		L-SHADE-EpSin	47.98	0.47	3.93	29.24	77.49	22	0.90	**0.95**
		UMOEAs-II	48.82	0.31	**2.83**	28.61	89.57	22	0.90	0.93
		EBOwithCMAR	48.50	0.51	2.84	28.51	91.60	22	0.90	0.94
		HS-ES	46.72	0.45	3.27	29.90	66.56	39	**0.92**	**0.95**
		QAGO	**49.23**	**0.19**	3.72	**31.17**	**49.72**	**20**	**0.92**	**0.95**

with the winners and GO, QAGO showed competitive optimization results on all four test images, which were confirmed by indicators such as PSNR, MSE, MAE, SIM, and FSIM.

6. Application II: wireless sensor network node deployment

Wireless sensor network (WSN) is widely regarded as one of the most crucial technologies of the 21st century (Liao, Kao, & Wu, 2011). They assume a fundamental role in various fields, such as physical security, traffic monitoring, military sensing, industrial automation, and environmental monitoring (Chong & Kumar, 2003; Sheikh-Hosseini & Hashemi, 2022). The deployment problem of WSN is a challenge, as it is aimed at ensuring the effective completion of monitoring tasks (Khedr, Osamy, & Agrawal, 2009). The coverage range in WSN needs to be ensured in order to reliably monitor the area (Liao, Kao, & Li, 2011; Wang, Ghosh, & Das, 2010). Therefore, the coverage optimization

Table 16
Comparison of the performance of the algorithms on the fighter image and the boat image.

Test image	K	Algorithms	F_{ave}	Std	T_{ave}	PSNR	MSE	MAE	SSIM	FSIM
Fighter	5	GO	16.68	0.06	1.68	18.72	872.42	99	0.92	0.86
		L-SHADE	16.32	0.11	1.73	18.93	831.30	**68**	0.92	0.87
		L-SHADE-EpSin	16.74	0.08	1.53	19.54	722.62	96	0.93	0.87
		UMOEAs-II	**16.78**	**0.02**	**1.28**	18.71	875.55	99	0.92	0.86
		EBOwithCMAR	16.77	0.04	1.32	19.24	773.92	97	0.93	0.87
		HS-ES	16.62	0.09	1.44	20.09	636.47	94	0.93	**0.88**
		QAGO	16.77	0.05	1.30	**20.39**	**593.94**	93	**0.95**	**0.88**
	10	GO	25.48	0.15	2.50	18.80	857.51	58	0.92	0.85
		L-SHADE	24.17	0.29	1.87	23.37	299.12	**39**	0.92	0.89
		L-SHADE-EpSin	24.97	0.20	2.45	18.01	1027.68	76	0.91	0.84
		UMOEAs-II	25.45	**0.11**	1.87	23.57	286.12	51	0.95	0.91
		EBOwithCMAR	**25.54**	**0.11**	1.87	21.21	491.90	46	0.94	0.89
		HS-ES	24.88	0.24	2.14	24.04	256.31	43	0.95	0.91
		QAGO	25.44	0.20	**1.78**	**24.45**	**233.53**	46	**0.96**	**0.92**
	15	GO	31.60	0.18	2.97	24.51	230.21	38	0.94	0.91
		L-SHADE	29.80	0.45	2.43	28.56	90.50	37	0.93	0.94
		L-SHADE-EpSin	30.49	0.34	3.33	27.67	111.31	**35**	0.93	0.92
		UMOEAs-II	31.40	0.30	**2.35**	22.87	335.55	36	0.89	0.88
		EBOwithCMAR	31.19	0.41	2.40	**30.06**	**64.08**	36	**0.96**	**0.96**
		HS-ES	30.43	0.34	2.75	18.29	963.71	56	0.91	0.83
		QAGO	**31.77**	**0.16**	2.54	23.25	307.57	45	0.94	0.90
	20	GO	36.53	**0.25**	3.34	29.48	73.37	37	**0.97**	**0.95**
		L-SHADE	34.03	0.77	2.65	26.92	132.08	32	0.93	0.93
		L-SHADE-EpSin	34.92	0.54	3.57	24.40	236.02	32	0.90	0.87
		UMOEAs-II	36.13	0.55	2.66	27.35	119.62	**28**	0.94	0.93
		EBOwithCMAR	35.62	0.37	**2.64**	26.05	161.45	35	0.96	0.94
		HS-ES	34.34	0.42	2.95	27.30	120.95	34	0.94	0.94
		QAGO	**36.59**	0.39	2.70	**29.92**	**66.28**	32	0.95	**0.95**
Boat	5	GO	21.37	0.02	1.53	**21.02**	**513.55**	57	**0.82**	**0.91**
		L-SHADE	21.27	0.05	1.41	20.98	518.47	56	**0.82**	0.90
		L-SHADE-EpSin	21.36	0.02	1.49	20.58	568.77	**52**	0.80	0.90
		UMOEAs-II	**21.40**	**0.01**	1.28	20.16	626.94	53	0.80	0.89
		EBOwithCMAR	21.39	**0.01**	1.29	20.45	585.90	53	0.80	0.89
		HS-ES	21.34	0.03	1.36	19.85	672.64	59	0.78	0.88
		QAGO	21.37	**0.01**	**1.17**	20.09	636.39	55	0.80	0.89
	10	GO	32.90	**0.07**	2.10	25.78	171.75	32	0.92	**0.96**
		L-SHADE	32.64	0.19	1.81	26.10	159.44	48	0.93	**0.96**
		L-SHADE-EpSin	32.81	0.13	2.17	24.73	218.63	**31**	0.90	0.95
		UMOEAs-II	33.09	0.11	1.74	**26.12**	**158.76**	35	0.93	**0.96**
		EBOwithCMAR	33.06	0.08	1.76	24.62	224.37	35	0.90	0.95
		HS-ES	32.57	0.19	1.97	25.71	174.43	38	0.92	**0.96**
		QAGO	**33.10**	0.08	**1.72**	25.93	165.83	**31**	**0.94**	**0.96**
	15	GO	42.19	0.13	2.22	26.85	134.19	**24**	0.95	**0.98**
		L-SHADE	41.18	0.32	2.10	26.97	130.53	34	0.95	0.96
		L-SHADE-EpSin	41.75	0.15	2.95	25.18	197.09	27	0.93	0.97
		UMOEAs-II	42.11	0.18	2.10	27.13	125.83	**24**	0.95	0.97
		EBOwithCMAR	42.04	0.23	2.32	26.12	158.85	29	0.94	0.97
		HS-ES	41.11	0.33	2.41	27.12	126.24	28	0.94	0.96
		QAGO	**42.26**	**0.12**	**2.09**	**27.97**	**103.88**	**24**	**0.96**	**0.98**
	20	GO	49.76	0.16	3.48	30.52	57.73	**20**	**0.97**	0.98
		L-SHADE	48.04	0.35	2.69	25.95	165.19	30	0.92	0.95
		L-SHADE-EpSin	48.62	0.30	3.95	28.53	91.21	30	0.96	0.98
		UMOEAs-II	49.36	0.42	2.80	30.24	57.47	21	0.96	0.98
		EBOwithCMAR	49.26	0.28	2.61	28.40	94.08	29	**0.97**	0.96
		HS-ES	47.76	0.49	2.96	28.57	90.37	26	0.95	0.97
		QAGO	**49.77**	**0.10**	**2.59**	**30.57**	**52.50**	**20**	**0.97**	**0.99**

problem of WSN has been a hot research topic for a long time as an important component of wireless sensor network node deployment.

Wireless sensor nodes are crucial components of network systems, serving as terminal structures for sensing and monitoring external objects. The extent of area coverage achieved through the deployment of these nodes has a direct impact on the overall service quality of the network (Khedr, Osamy, & Salim, 2018). The regional coverage of the network is an evaluation indicator for coverage optimization, which is usually treated as the maximization goal to be achieved in optimization tasks.

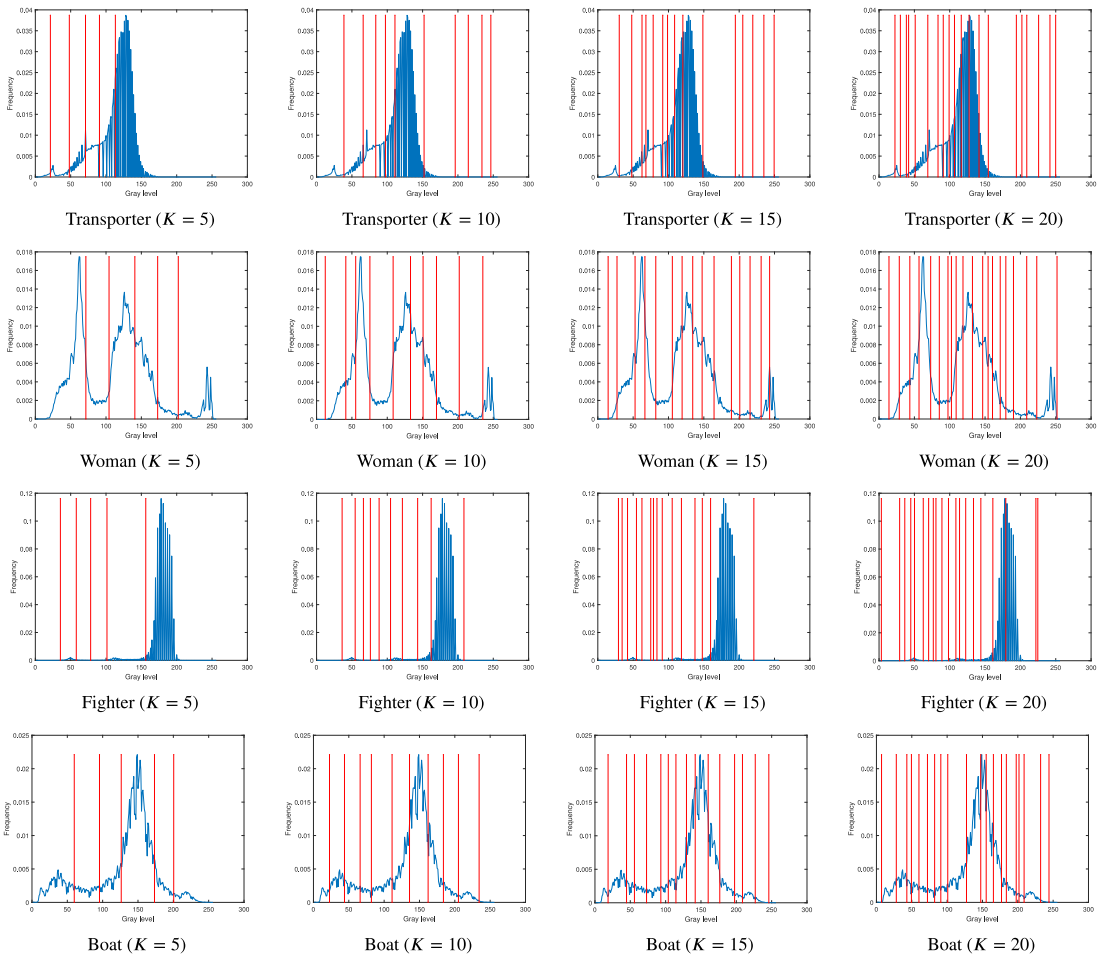


Fig. 11. The corresponding threshold distributions were obtained through QAGO.

6.1. Mathematical model of WSN coverage rate

Assuming v identical wireless sensor nodes are randomly distributed on a designated two-dimensional plane, the set of these nodes is denoted as $C = c_1, c_2, \ldots, c_v$. Each node has the same perception radius R_s and communication radius R_c, satisfying $R_c = 2R_s$. If a plane region is discretized into $m \times n$ pixels, then the Euclidean distance from node i to any pixel $z(x, y)$ is described by Eq. (33):

$$d\left(c_i, z\right) = \sqrt{\left(x - x_i\right)^2 + \left(y - y_i\right)^2} \tag{33}$$

where the coordinate of the node is $c_i = (x_i, y_i), i = 1, 2, \ldots, v$. If a pixel point is located within the perception radius of node i, as determined by the Euclidean distance between the two, the pixel point is considered to be in a covered state. The likelihood of this occurrence is represented by P in Eq. (34):

$$P\left(c_i, z\right) = \begin{cases} 1, & d\left(c_i, z\right) \le R_s - R_e \\ exp(\frac{-\lambda_1 \alpha_1^{\beta_1}}{\lambda_2 + \alpha_2^{\beta_2}}), & R_s - R_e < d\left(c_i, z\right) < R_s + R_e \\ 0, & else \end{cases} \tag{34}$$

where R_e is the reliability parameter of node measurement, and $\alpha_1 = R_e - R_s + d(c_i, z)$, $\alpha_2 = R_e + R_s - d(c_i, z)$. λ_1, λ_2, β_1, and β_2 are measurement parameters related to nodes, and $\lambda_1 = 1$, $\lambda_2 = 0$, $\beta_1 = 1$, and $\beta_2 = 1.5$ are selected in this research. After obtaining the probability of pixel detection, Eq. (35) is used to calculate the joint detection probability of all nodes in the region for the target point:

$$P\left(C, z\right) = 1 - \prod_{i=1}^{v} \left(1 - P\left(c_i, z\right)\right) \tag{35}$$

where $P(C, z)$ represents the joint detection probability. Here, C represents the set of sensor nodes, and v corresponds to the number of nodes in the region.

In the experiment, a grid-based approach is used to select the detection point z, which discretizes the detection area into $m \times n$ pixels. The network coverage (P_{cov}) of this area is then calculated using Eq. (36):

$$P_{cov} = \frac{\sum_{x=1}^{m} \sum_{y=1}^{n} P\left(C, z\right)}{m \times n} \tag{36}$$

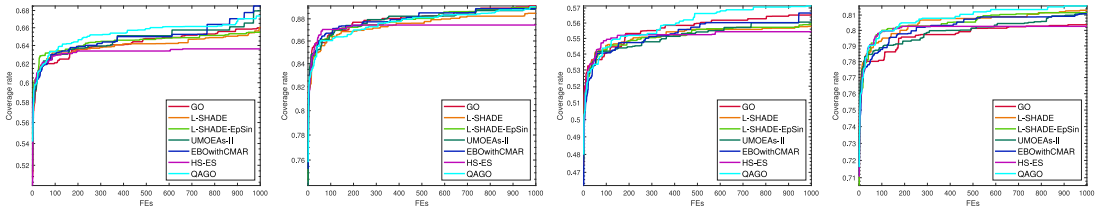
6.2. Comparison experiment of QAGO, GO, and winners

In this section, we will compare QAGO with standard GO and IEEE CEC competition winners. To do this, we have designed four experimental cases. The first two cases are based on the deployment of network nodes in a 30 × 30 space with $R_s = 5$, while the last two cases are based on the deployment of network nodes in a 100 × 100 space with $R_s = 10$. The population size N for all algorithms is consistent with the value set in the numerical optimization experiment. Additionally, the dimension of individuals in each algorithm is set to $D = 30/40/60/80$, with a node position determined by every two dimensions, resulting in $v = 15/20/30/40$ nodes. Each algorithm is independently executed 30 times, with $MaxFEs$ set to 1000.

Table 17, Fig. 12 and Fig. 13 show the experimental results, in which EBOwithCMAR achieves the best deployment coverage in cases

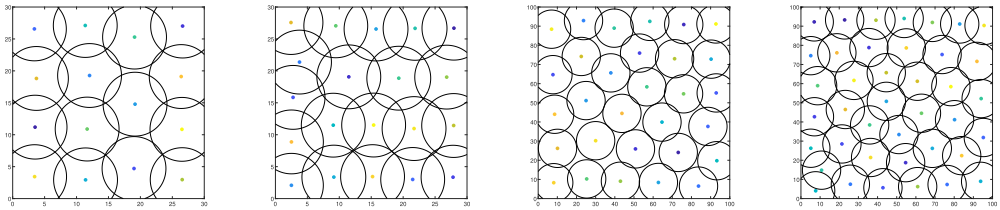
Table 17
Comparison results of QAGO and competitive algorithms on four WSN node deployment cases.

Case	Metric	GO	L-SHADE	L-SHADE-EpSin	UMOEAs-II	EBOwithCMAR	HS-ES	QAGO
Case 1 ($v = 15, D = 30$)	Mean	6.60E−01	6.60E−01	6.55E−01	6.85E−01	**6.92E−01**	6.36E−01	6.74E−01
	Std	1.10E−02	9.48E−03	1.13E−02	3.21E−02	**3.97E−02**	7.56E−03	1.12E−02
Case 2 ($v = 20, D = 40$)	Mean	8.92E−01	8.85E−01	8.90E−01	8.92E−01	**8.94E−01**	8.80E−01	8.91E−01
	Std	7.15E−03	8.04E−03	9.81E−03	1.68E−02	**2.03E−02**	8.11E−03	7.43E−03
Case 3 ($v = 30, D = 60$)	Mean	7.17E−01	7.13E−01	7.14E−01	7.16E−01	7.15E−01	7.08E−01	**7.23E−01**
	Std	9.58E−03	7.28E−03	9.16E−03	7.89E−03	1.06E−02	1.41E−02	**1.17E−02**
Case 4 ($v = 40, D = 80$)	Mean	8.04E−01	8.13E−01	8.16E−01	8.14E−01	8.17E−01	8.14E−01	**8.19E−01**
	Std	8.24E−03	7.03E−03	7.42E−03	7.36E−03	6.21E−03	1.34E−02	**1.09E−02**



Case 1 Case 2 Case 3 Case 4

Fig. 12. Convergence curves of QAGO and competitive algorithms in different cases.



Case 1 Case 2 Case 3 Case 4

Fig. 13. Optimal wireless sensor deployment in different cases.

1 and 2, while the proposed algorithm achieves the best deployment coverage in cases 3 and 4. Therefore, the application of QAGO in WSN deployment demonstrates that compared with the winners and GO, it has competitive optimization ability and practical application value. This further validates the effectiveness of QAGO in improving GO from an alternative perspective.

7. Discussion

To clearly present the effectiveness and competitiveness of the QAGO algorithm proposed in this paper, all the experimental results in this work are summarized and discussed in this section. In comparison results (Tables 3, 4, and 5) with the novel metaheuristic algorithms, QAGO's convergence accuracy results outperform the original standard algorithms on most of the benchmark functions. In particular, this advantage becomes increasingly evident as the problem dimension increases. This also indirectly confirms the effectiveness of the strategies of QAGO. According to the results of the convergence curves (Fig. 4), compared with other algorithms, QAGO shows more potential optimization ability, especially on F1, F3, F6, and F9. Its convergence curves still show obvious continued convergence behavior in the late iteration, which suggests that QAGO is able to alleviate the problems caused by the local optimum and thus obtain more promising solutions. The box plots (Fig. 7) show that the distribution of QAGO's results is more centralized. Its adaptive strategy can adjust its search according to different problems, making its performance more stable and robust. Statistical results (Tables 7 and 8) demonstrate that QAGO performs well on benchmarks of different dimensions and has comprehensive and effective problem optimization capabilities. Moreover, this capability of QAGO is better supported by the cumulative error results (Figs. 5 and 6) on a total of 90 problems and the comparison results (Tables 9 and 10) with 50 classical algorithms. In further experiments, QAGO obtains expected numerical optimization results (Tables 11, 12, and 13) even though the comparison involves six advanced self-adaptive algorithms and five powerful winner algorithms. This proves the effectiveness and competitiveness of its adaptive strategy compared to similar or advanced strategies.

In the search behavior analysis results (Fig. 8), QAGO demonstrates good convergence, swarm intelligence characteristics, avoidance of falling into local optimal behavior, and the ability to balance exploration and exploitation. In the complexity analysis results, QAGO keeps the computational complexity similar to GO while ensuring good performance. Moreover, based on the results (Tables 15, 16, and 17) of QAGO on two real-world application problems, it is shown that QAGO not only solves these problems well but also provides evidence that QAGO is more competitive than GO and winners in terms of Mean, Std, PSNR, MSE, MAS, SSIM, and FSIM metrics.

Based on the above discussion, it is shown that under the experimental conditions controlled in this study, the proposed method demonstrates more promising and competitive optimization capabilities than the existing studies or other studies.

8. Conclusion

This paper proposes a quadruple parameter adaptation growth optimizer (QAGO) with integrated distribution, confrontation, and balance features. The goal is to address the issue of imbalanced searching in engineering problems and benchmarks caused by the parameter

sensitivity and operator limitations of GO. The quadruple parameter adaptation mechanism includes parameter adaptation for tuning hyperparameters and triple-parameter self-adaptation for tuning operator parameters. The former is used to overcome the limitations of fixed hyperparameter settings in GO. The latter achieves a highly adaptive algorithm search by utilizing one-dimensional mapping of vectors, fitness difference, and Jensen–Shannon divergence methods to refine the operators. Furthermore, the refined operator collects extensive information relevant to the search space, enabling a more effective evolution of the population.

In the numerical optimization experiments conducted using the CEC 2017 and CEC 2022 boundary constraint test suites, QAGO was evaluated against 10 newly developed metaheuristic algorithms, 50 classical algorithms, 6 advanced self-adaptive algorithms, and five winners of the IEEE CEC competition. The results obtained were verified by the Wilcoxon rank-sum, Friedman, and Kruskal–Wallis tests, which showed that QAGO effectively solves the challenge of GO and provides more promising solutions than other competing algorithms for most problems. Furthermore, to showcase the practical application value of QAGO, it is applied to address multilevel threshold image segmentation and wireless sensor network node deployment problems. In the experiments conducted using various test cases, QAGO has demonstrated its effectiveness and competitiveness in solving these problems when compared to GO and the competition winners. This further validates the effectiveness of the QAGO strategy from an alternative perspective.

However, note that QAGO may not always find the optimal solution for optimization problems. Moreover, to achieve better performance, the computational complexity of QAGO may be slightly higher compared to GO. Further, QAGO typically requires a significant number of function evaluations to perform the adaptive process. As a result, when there are a limited number of function calls, the performance of QAGO may deteriorate. In the future, research plans will focus on exploring the performance of QAGO in high-dimensional spaces and extending the universality of methods for QAGO.

CRediT authorship contribution statement

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

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), the Major Fundamental Research Project of Shandong, China (No. ZR2019ZD03), and the Taishan Scholar Project of Shandong, China (No. ts20190924).

References

Abdel-Basset, M., Chang, V., & Mohamed, R. (2021). A novel equilibrium optimization algorithm for multi-thresholding image segmentation problems. *Neural Computing and Applications*, *33*, 10685–10718. http://dx.doi.org/10.1007/s00521-020-04820-y.

Abdel-Basset, M., Mohamed, R., Azeem, S. A. A., Jameel, M., & Abouhawwash, M. (2023). Kepler optimization algorithm: A new metaheuristic algorithm inspired by Kepler's laws of planetary motion. *Knowledge-Based Systems*, *268*, Article 110454. http://dx.doi.org/10.1016/j.knosys.2023.110454.

Abdel-Basset, M., Mohamed, R., Jameel, M., & Abouhawwash, M. (2023). Nutcracker optimizer: A novel nature-inspired metaheuristic algorithm for global optimization and engineering design problems. *Knowledge-Based Systems*, Article 110248. http://dx.doi.org/10.1016/j.knosys.2022.110248.

Abdelhamid, M., Houssein, E. H., Mahdy, M. A., Selim, A., & Kamel, S. (2022). An improved seagull optimization algorithm for optimal coordination of distance and directional over-current relays. *Expert Systems with Applications*, *200*, Article 116931. http://dx.doi.org/10.1016/j.eswa.2022.116931.

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 Systems with Applications*, *191*, Article 116158. http://dx.doi.org/10.1016/j.eswa.2021.116158.

Abualigah, L., Diabat, A., Mirjalili, S., Abd Elaziz, M., & Gandomi, A. H. (2021). The arithmetic optimization algorithm. *Computer Methods in Applied Mechanics and Engineering*, *376*, Article 113609. http://dx.doi.org/10.1016/j.cma.2020.113609.

Agushaka, J. O., Ezugwu, A. E., & Abualigah, L. (2022). Dwarf mongoose optimization algorithm. *Computer Methods in Applied Mechanics and Engineering*, *391*, Article 114570. http://dx.doi.org/10.1016/j.cma.2022.114570.

Ahmad, M. F., Isa, N. A. M., Lim, W. H., & Ang, K. M. (2022). Differential evolution: A recent review based on state-of-the-art works. *Alexandria Engineering Journal*, *61*(5), 3831–3872. http://dx.doi.org/10.1016/j.aej.2021.09.013.

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 Systems with Applications*, *181*, Article 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 Systems with Applications*, *195*, Article 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 Computing and Applications*, *33*, 5011–5042. http://dx.doi.org/10.1007/s00521-020-05296-6.

Aras, S., Gedikli, E., & Kahraman, H. T. (2021). A novel stochastic fractal search algorithm with fitness-distance balance for global numerical optimization. *Swarm and Evolutionary Computation*, *61*, Article 100821. http://dx.doi.org/10.1016/j.swevo.2020.100821.

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

Askari, Q., Saeed, M., & Younas, I. (2020). Heap-based optimizer inspired by corporate rank hierarchy for global optimization. *Expert Systems with Applications*, *161*, Article 113702. http://dx.doi.org/10.1016/j.eswa.2020.113702.

Atashpaz-Gargari, E., & Lucas, C. (2007). Imperialist competitive algorithm: An algorithm for optimization inspired by imperialistic competition. In *2007 IEEE congress on evolutionary computation* (pp. 4661–4667). IEEE, http://dx.doi.org/10.1109/CEC.2007.4425083.

Awad, N. H., Ali, M. Z., & Suganthan, P. N. (2017). Ensemble sinusoidal differential covariance matrix adaptation with euclidean neighborhood for solving CEC2017 benchmark problems. In *2017 IEEE congress on evolutionary computation* (pp. 372–379). IEEE, http://dx.doi.org/10.1109/CEC.2017.7969336.

Balochian, S., & Baloochian, H. (2019). Social mimic optimization algorithm and engineering applications. *Expert Systems with Applications*, *134*, 178–191. http://dx.doi.org/10.1016/j.eswa.2019.05.035.

Bhandari, A. K., Singh, V. K., Kumar, A., & Singh, G. K. (2014). Cuckoo search algorithm and wind driven optimization based study of satellite image segmentation for multilevel thresholding using Kapur's entropy. *Expert Systems with Applications*, *41*(7), 3538–3560. http://dx.doi.org/10.1016/j.eswa.2013.10.059.

Braik, M., Hammouri, A., Atwan, J., Al-Betar, M. A., & Awadallah, M. A. (2022). White shark optimizer: A novel bio-inspired meta-heuristic algorithm for global optimization problems. *Knowledge-Based Systems*, *243*, Article 108457. http://dx.doi.org/10.1016/j.knosys.2022.108457.

Caraffini, F., & Neri, F. (2019). A study on rotation invariance in differential evolution. *Swarm and Evolutionary Computation*, *50*, Article 100436. http://dx.doi.org/10.1016/j.swevo.2018.08.013.

Carrasco, J., García, S., Rueda, M., Das, S., & Herrera, F. (2020). Recent trends in the use of statistical tests for comparing swarm and evolutionary computing algorithms: Practical guidelines and a critical review. *Swarm and Evolutionary Computation*, *54*, Article 100665. http://dx.doi.org/10.1016/j.swevo.2020.100665.

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 Systems with Applications*, *200*, Article 117118. http://dx.doi.org/10.1016/j.eswa.2022.117118.

Chong, C.-Y., & Kumar, S. P. (2003). Sensor networks: Evolution, opportunities, and challenges. *Proceedings of the IEEE*, *91*(8), 1247–1256. http://dx.doi.org/10.1109/JPROC.2003.814918.

Civicioglu, P. (2012). Transforming geocentric cartesian coordinates to geodetic coordinates by using differential search algorithm. *Computers & Geosciences*, *46*, 229–247. http://dx.doi.org/10.1016/j.cageo.2011.12.011.

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

Das, S., & Suganthan, P. N. (2011). Differential evolution: A survey of the state-of-the-art. *IEEE Transactions on Evolutionary Computation*, *15*(1), 4–31. http://dx.doi.org/10.1109/TEVC.2010.2059031.

Dash, S. P., Subhashini, K., & Chinta, P. (2022). Development of a boundary assigned animal migration optimization algorithm and its application to optimal power flow study. *Expert Systems with Applications*, *200*, Article 116776. http://dx.doi.org/10.1016/j.eswa.2022.116776.

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 and Evolutionary Computation*, *75*, Article 101193. http://dx.doi.org/10.1016/j.swevo.2022.101193.

de Lacerda, M. G. P., de Araujo Pessoa, L. F., de Lima Neto, F. B., Ludermir, T. B., & Kuchen, H. (2021). A systematic literature review on general parameter control for evolutionary and swarm-based algorithms. *Swarm and Evolutionary Computation*, *60*, Article 100777. http://dx.doi.org/10.1016/j.swevo.2020.100777.

Dehghani, M., Montazeri, Z., Trojovská, E., & Trojovskỳ, P. (2023). Coati optimization algorithm: A new bio-inspired metaheuristic algorithm for solving optimization problems. *Knowledge-Based Systems*, *259*, Article 110011. http://dx.doi.org/10.1016/j.knosys.2022.110011.

Del Ser, J., Osaba, E., Molina, D., Yang, X.-S., Salcedo-Sanz, S., Camacho, D., et al. (2019). Bio-inspired computation: Where we stand and what's next. *Swarm and Evolutionary Computation*, *48*, 220–250. http://dx.doi.org/10.1016/j.swevo.2019.04.008.

Ding, L., Bai, Y.-L., Fan, M.-H., Yu, Q.-H., Zhu, Y.-J., & Chen, X.-Y. (2023). Serial-parallel dynamic echo state network: A hybrid dynamic model based on a chaotic coyote optimization algorithm for wind speed prediction. *Expert Systems with Applications*, *212*, Article 118789. http://dx.doi.org/10.1016/j.eswa.2022.118789.

Dinkar, S. K., Deep, K., Mirjalili, S., & Thapliyal, S. (2021). Opposition-based Laplacian equilibrium optimizer with application in image segmentation using multilevel thresholding. *Expert Systems with Applications*, *174*, Article 114766. http://dx.doi.org/10.1016/j.eswa.2021.114766.

Dokeroglu, T., Sevinc, E., Kucukyilmaz, T., & Cosar, A. (2019). A survey on new generation metaheuristic algorithms. *Computers & Industrial Engineering*, *137*, Article 106040. http://dx.doi.org/10.1016/j.cie.2019.106040.

Dorigo, M., Birattari, M., & Stutzle, T. (2006). Ant colony optimization. *IEEE Computational Intelligence Magazine*, *1*(4), 28–39. http://dx.doi.org/10.1109/MCI.2006.329691.

Du, S., Zhou, W., Wu, D., & Fei, M. (2023). An effective discrete monarch butterfly optimization algorithm for distributed blocking flow shop scheduling with an assembly machine. *Expert Systems with Applications*, *225*, Article 120113. http://dx.doi.org/10.1016/j.eswa.2023.120113.

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.

Elsayed, S., Hamza, N., & Sarker, R. (2016). Testing united multi-operator evolutionary algorithms-II on single objective optimization problems. In *2016 IEEE congress on evolutionary computation* (pp. 2966–2973). IEEE, http://dx.doi.org/10.1109/CEC.2016.7744164.

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

Ezugwu, A. E., Shukla, A. K., Nath, R., Akinyelu, A. A., Agushaka, J. O., Chiroma, H., et al. (2021). Metaheuristics: A comprehensive overview and classification along with bibliometric analysis. *Artificial Intelligence Review*, *54*, 4237–4316. http://dx.doi.org/10.1007/s10462-020-09952-0.

Faramarzi, A., Heidarinejad, M., Mirjalili, S., & Gandomi, A. H. (2020). Marine predators algorithm: A nature-inspired metaheuristic. *Expert Systems with Applications*, *152*, Article 113377. http://dx.doi.org/10.1016/j.eswa.2020.113377.

Faramarzi, A., Heidarinejad, M., Stephens, B., & Mirjalili, S. (2020). Equilibrium optimizer: A novel optimization algorithm. *Knowledge-Based Systems*, *191*, Article 105190. http://dx.doi.org/10.1016/j.knosys.2019.105190.

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

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. *Knowledge-Based Systems*, *257*, Article 109484. http://dx.doi.org/10.1016/j.knosys.2022.109484.

Hansen, N. (2006). The CMA evolution strategy: A comparing review. *Towards a New Evolutionary Computation*, 75–102. http://dx.doi.org/10.1007/3-540-32494-1_4.

Hansen, N. (2016). The CMA evolution strategy: A tutorial. http://dx.doi.org/10.48550/arXiv.1604.00772, arXiv preprint arXiv:1604.00772.

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. *Mathematics and Computers in 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. *Knowledge-Based Systems*, *242*, Article 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. *Knowledge-Based Systems*, *260*, Article 110146. http://dx.doi.org/10.1016/j.knosys.2022.110146.

Heidari, A. A., Mirjalili, S., Faris, H., Aljarah, I., Mafarja, M., & Chen, H. (2019). Harris hawks optimization: Algorithm and applications. *Future Generation Computer Systems*, *97*, 849–872. http://dx.doi.org/10.1016/j.future.2019.02.028.

Holland, J. H. (1992). *Adaptation in natural and artificial systems: An introductory analysis with applications to biology, control, and artificial intelligence*. MIT Press: http://dx.doi.org/10.1086/418447.

Houssein, E. H., Emam, M. M., & Ali, A. A. (2021). An efficient multilevel thresholding segmentation method for thermography breast cancer imaging based on improved chimp optimization algorithm. *Expert Systems with Applications*, *185*, Article 115651. http://dx.doi.org/10.1016/j.eswa.2021.115651.

Houssein, E. H., Oliva, D., Çelik, E., Emam, M. M., & Ghoniem, R. M. (2023). Boosted sooty tern optimization algorithm for global optimization and feature selection. *Expert Systems with Applications*, *213*, Article 119015. http://dx.doi.org/10.1016/j.eswa.2022.119015.

Huang, C., Li, Y., & Yao, X. (2020). A survey of automatic parameter tuning methods for metaheuristics. *IEEE Transactions on Evolutionary Computation*, *24*(2), 201–216. http://dx.doi.org/10.1109/TEVC.2019.2921598.

Jia, H., Peng, X., & Lang, C. (2021). Remora optimization algorithm. *Expert Systems with Applications*, *185*, Article 115665. http://dx.doi.org/10.1016/j.eswa.2021.115665.

Kapur, J. N., Sahoo, P. K., & Wong, A. K. (1985). A new method for gray-level picture thresholding using the entropy of the histogram. *Computer Vision, Graphics, and Image Processing*, *29*(3), 273–285. http://dx.doi.org/10.1016/0734-189X(85)90125-2.

Karaboga, D., & Basturk, B. (2007). A powerful and efficient algorithm for numerical function optimization: Artificial bee colony (ABC) algorithm. *Journal of Global Optimization*, *39*, 459–471. http://dx.doi.org/10.1007/s10898-007-9149-x.

Kennedy, J., & Eberhart, R. (1995). Particle swarm optimization. In *Proceedings of ICNN'95-international conference on neural networks, Vol. 4* (pp. 1942–1948). IEEE, http://dx.doi.org/10.1109/ICNN.1995.488968.

Khedr, A. M., Osamy, W., & Agrawal, D. P. (2009). Perimeter discovery in wireless sensor networks. *Journal of Parallel and Distributed Computing*, *69*(11), 922–929. http://dx.doi.org/10.1016/j.jpdc.2009.08.002.

Khedr, A. M., Osamy, W., & Salim, A. (2018). Distributed coverage hole detection and recovery scheme for heterogeneous wireless sensor networks. *Computer Communications*, *124*, 61–75. http://dx.doi.org/10.1016/j.comcom.2018.04.002.

Kirkpatrick, S., Gelatt, 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.

Kreischer, V., Magalhaes, T. T., Barbosa, H., & Krempser, E. (2017). Evaluation of bound constraints handling methods in differential evolution using the CEC 2017 benchmark. In *XIII Brazilian congress on computational intelligence*. http://dx.doi.org/10.21528/cbic2017-68.

Kumar, A., Misra, R. K., & Singh, D. (2017). Improving the local search capability of effective butterfly optimizer using covariance matrix adapted retreat phase. In *2017 IEEE Congress on Evolutionary Computation* (pp. 1835–1842). IEEE, http://dx.doi.org/10.1109/CEC.2017.7969524.

Lamberti, P. W., Majtey, A. P., Borras, A., Casas, M., & Plastino, A. (2008). Metric character of the quantum Jensen-Shannon divergence. *Physical Review A*, *77*(5), Article 052311. http://dx.doi.org/10.1103/PhysRevA.77.052311.

Liao, W.-H., Kao, Y., & Li, Y.-S. (2011). A sensor deployment approach using glowworm swarm optimization algorithm in wireless sensor networks. *Expert Systems with Applications*, *38*(10), 12180–12188. http://dx.doi.org/10.1016/j.eswa.2011.03.053.

Liao, W.-H., Kao, Y., & Wu, R.-T. (2011). Ant colony optimization based sensor deployment protocol for wireless sensor networks. *Expert Systems with Applications*, *38*(6), 6599–6605. http://dx.doi.org/10.1016/j.eswa.2010.11.079.

Liu, X.-F., Zhan, Z.-H., Lin, Y., Chen, W.-N., Gong, Y.-J., Gu, T.-L., et al. (2018). Historical and heuristic-based adaptive differential evolution. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, *49*(12), 2623–2635. http://dx.doi.org/10.1109/TSMC.2018.2855155.

Liu, H., Zhang, X., Zhang, H., Li, C., & Chen, Z. (2023). A reinforcement learning-based hybrid aquila optimizer and improved arithmetic optimization algorithm for global optimization. *Expert Systems with Applications*, *224*, Article 119898. http://dx.doi.org/10.1016/j.eswa.2023.119898.

Ma, Z., Wu, G., Suganthan, P. N., Song, A., & Luo, Q. (2023). Performance assessment and exhaustive listing of 500+ nature-inspired metaheuristic algorithms. *Swarm and Evolutionary Computation*, *77*, Article 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. *Knowledge-Based Systems*, *258*, Article 110032. http://dx.doi.org/10.1016/j.knosys.2022.110032.

McKight, P. E., & Najab, J. (2010). Kruskal-wallis test. *The Corsini Encyclopedia of Psychology*, 1. http://dx.doi.org/10.1002/9780470479216.corpsy0491.

Mirjalili, S. (2015). Moth-flame optimization algorithm: A novel nature-inspired heuristic paradigm. *Knowledge-Based Systems*, *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. *Knowledge-Based Systems*, *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. *Advances in Engineering Software*, *114*, 163–191. http://dx.doi.org/10.1016/j.advengsoft.2017.07.002.

Mirjalili, S., & Lewis, A. (2016). The whale optimization algorithm. *Advances in Engineering Software*, *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 Computing and Applications*, *27*, 495–513. http://dx.doi.org/10.1007/s00521-015-1870-7.

Mirjalili, S., Mirjalili, S. M., & Lewis, A. (2014). Grey wolf optimizer. *Advances in Engineering Software*, *69*, 46–61. http://dx.doi.org/10.1016/j.advengsoft.2013.12.007.

Mohamed, A. W., Hadi, A. A., Fattouh, A. M., & Jambi, K. M. (2017). LSHADE with semi-parameter adaptation hybrid with CMA-ES for solving CEC 2017 benchmark problems. In *2017 IEEE congress on evolutionary computation* (pp. 145–152). IEEE, http://dx.doi.org/10.1109/CEC.2017.7969307.

Mohamed, A. W., Hadi, A. A., & Mohamed, A. K. (2020). Gaining-sharing knowledge based algorithm for solving optimization problems: A novel nature-inspired algorithm. *International Journal of Machine Learning and Cybernetics*, *11*(7), 1501–1529. http://dx.doi.org/10.1007/s13042-019-01053-x.

Naderipour, A., Abdullah, A., Marzbali, M. H., & Nowdeh, S. A. (2022). An improved corona-virus herd immunity optimizer algorithm for network reconfiguration based on fuzzy multi-criteria approach. *Expert Systems with Applications*, *187*, Article 115914. http://dx.doi.org/10.1016/j.eswa.2021.115914.

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. *Artificial Intelligence Review*, *42*(4), 965–997. http://dx.doi.org/10.1007/s10462-012-9342-2.

Nickabadi, A., Ebadzadeh, M. M., & Safabakhsh, R. (2011). A novel particle swarm optimization algorithm with adaptive inertia weight. *Applied Soft Computing*, *11*(4), 3658–3670. http://dx.doi.org/10.1016/j.asoc.2011.01.037.

Opara, K. R., & Arabas, J. (2019). Differential evolution: A survey of theoretical analyses. *Swarm and Evolutionary Computation*, *44*, 546–558. http://dx.doi.org/10.1016/j.swevo.2018.06.010.

Pierezan, J., & Coelho, L. D. S. (2018). Coyote optimization algorithm: A new metaheuristic for global optimization problems. In *2018 IEEE congress on evolutionary computation* (pp. 1–8). IEEE, http://dx.doi.org/10.1109/CEC.2018.8477769.

Qin, A. K., & Suganthan, P. N. (2005). Self-adaptive differential evolution algorithm for numerical optimization. In *2005 IEEE congress on evolutionary computation, vol. 2* (pp. 1785–1791). IEEE, http://dx.doi.org/10.1109/CEC.2005.1554904.

Rao, R. (2016). Jaya: A simple and new optimization algorithm for solving constrained and unconstrained optimization problems. *International Journal of Industrial Engineering Computations*, *7*(1), 19–34. http://dx.doi.org/10.5267/j.ijiec.2015.8.004.

Rao, R. V., Savsani, V. J., & Vakharia, D. (2011). Teaching–learning-based optimization: A novel method for constrained mechanical design optimization problems. *Computer-Aided Design*, *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. *Information Sciences*, *179*(13), 2232–2248. http://dx.doi.org/10.1016/j.ins.2009.03.004.

Rather, S. A., & Bala, P. S. (2021). Constriction coefficient based particle swarm optimization and gravitational search algorithm for multilevel image thresholding. *Expert Systems*, *38*(7), Article e12717. http://dx.doi.org/10.1111/exsy.12717.

Reynolds, R. G. (1994). An introduction to cultural algorithms. In *Proceedings of the 3rd annual conference on evolutionary programming, world scientific publishing* (pp. 131–139). World Scientific, http://dx.doi.org/10.1142/9789814534116.

Salimi, H. (2015). Stochastic fractal search: A powerful metaheuristic algorithm. *Knowledge-Based Systems*, *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* (pp. 1–8). IEEE, http://dx.doi.org/10.1109/CEC48606.2020.9185577.

Sathya, P., Kalyani, R., & Sakthivel, V. (2021). Color image segmentation using Kapur, Otsu and minimum cross entropy functions based on exchange market algorithm. *Expert Systems with Applications*, *172*, Article 114636. http://dx.doi.org/10.1016/j.eswa.2021.114636.

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 Systems with Applications*, *161*, Article 113698. http://dx.doi.org/10.1016/j.eswa.2020.113698.

Shareef, H., Ibrahim, A. A., & Mutlag, A. H. (2015). Lightning search algorithm. *Applied Soft Computing*, *36*, 315–333. http://dx.doi.org/10.1016/j.asoc.2015.07.028.

Sheikh-Hosseini, M., & Hashemi, S. R. S. (2022). Connectivity and coverage constrained wireless sensor nodes deployment using steepest descent and genetic algorithms. *Expert Systems with Applications*, *190*, Article 116164. http://dx.doi.org/10.1016/j.eswa.2021.116164.

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* (pp. 303–309). Springer, http://dx.doi.org/10.1007/978-3-642-21515-5_36.

Simon, D. (2008). Biogeography-based optimization. *IEEE Transactions on Evolutionary Computation*, *12*(6), 702–713. http://dx.doi.org/10.1109/TEVC.2008.919004.

Singh, S., Mittal, N., Nayyar, A., Singh, U., & Singh, S. (2023). A hybrid transient search naked mole-rat optimizer for image segmentation using multilevel thresholding. *Expert Systems with Applications*, *213*, Article 119021. http://dx.doi.org/10.1016/j.eswa.2022.119021.

Storn, R., & Price, K. (1997). Differential evolution–a simple and efficient heuristic for global optimization over continuous spaces. *Journal of Global Optimization*, *11*(4), 341–359. http://dx.doi.org/10.1023/A:1008202821328.

Sun, J., Liu, X., Bäck, T., & Xu, Z. (2021). Learning adaptive differential evolution algorithm from optimization experiences by policy gradient. *IEEE Transactions on Evolutionary Computation*, *25*(4), 666–680. http://dx.doi.org/10.1109/TEVC.2021.3060811.

Tanabe, R., & Fukunaga, A. (2013). Success-history based parameter adaptation for differential evolution. In *2013 IEEE congress on evolutionary computation* (pp. 71–78). IEEE, http://dx.doi.org/10.1109/CEC.2013.6557555.

Tanabe, R., & Fukunaga, A. S. (2014). Improving the search performance of SHADE using linear population size reduction. In *2014 IEEE congress on evolutionary computation* (pp. 1658–1665). IEEE, http://dx.doi.org/10.1109/CEC.2014.6900380.

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

Wang, Y., Cai, Z., & Zhang, Q. (2011). Differential evolution with composite trial vector generation strategies and control parameters. *IEEE Transactions on Evolutionary Computation*, *15*(1), 55–66. http://dx.doi.org/10.1109/TEVC.2010.2087271.

Wang, J., Ghosh, R. K., & Das, S. K. (2010). A survey on sensor localization. *Journal of Control Theory and Applications*, *8*(1), 2–11. http://dx.doi.org/10.1007/s11768-010-9187-7.

Wang, Z.-J., Jian, J.-R., Zhan, Z.-H., Li, Y., Kwong, S., & Zhang, J. (2022). Gene targeting differential evolution: A simple and efficient method for large scale optimization. *IEEE Transactions on Evolutionary Computation*, http://dx.doi.org/10.1109/TEVC.2022.3185665.

Wang, C., Jiao, S., Li, Y., & Zhang, Q. (2023). Capacity optimization of a hybrid energy storage system considering wind-solar reliability evaluation based on a novel multi-strategy snake optimization algorithm. *Expert Systems with Applications*, Article 120602. http://dx.doi.org/10.1016/j.eswa.2023.120602.

Wang, X., Xu, J., & Huang, C. (2023). Fans optimizer: A human-inspired optimizer for mechanical design problems optimization. *Expert Systems with Applications*, *228*, Article 120242. http://dx.doi.org/10.1016/j.eswa.2023.120242.

Wolpert, D. H., & Macready, W. G. (1997). No free lunch theorems for optimization. *IEEE Transactions on Evolutionary Computation*, *1*(1), 67–82. http://dx.doi.org/10.1109/4235.585893.

Wu, L., Huang, X., Cui, J., Liu, C., & Xiao, W. (2023). Modified adaptive ant colony optimization algorithm and its application for solving path planning of mobile robot. *Expert Systems with Applications*, *215*, Article 119410. http://dx.doi.org/10.1016/j.eswa.2022.119410.

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://raw.githubusercontent.com/P-N-Suganthan/CEC2017-BoundContrained/master/Bound-Constrained-Comparisons.pdf.

Wu, G., Mallipeddi, R., & Suganthan, P. N. (2019). Ensemble strategies for population-based optimization algorithms–A survey. *Swarm and Evolutionary Computation*, *44*, 695–711. http://dx.doi.org/10.1016/j.swevo.2018.08.015.

Xian, S., & Feng, X. (2023). Meerkat optimization algorithm: A new meta-heuristic optimization algorithm for solving constrained engineering problems. *Expert Systems with Applications*, Article 120482. http://dx.doi.org/10.1016/j.eswa.2023.120482.

Xin-gang, Z., Ji, L., Jin, M., & Ying, Z. (2020). An improved quantum particle swarm optimization algorithm for environmental economic dispatch. *Expert Systems with Applications*, *152*, Article 113370. http://dx.doi.org/10.1016/j.eswa.2020.113370.

Yadav, A., et al. (2019). AEFA: Artificial electric field algorithm for global optimization. *Swarm and Evolutionary Computation*, *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* (pp. 169–178). Springer, 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* (pp. 240–249). Springer, http://dx.doi.org/10.1007/978-3-642-32894-7_27.

Yang, X.-S., & Deb, S. (2009). Cuckoo search via Lévy flights. In *2009 world congress on nature & biologically inspired computing* (pp. 210–214). IEEE, http://dx.doi.org/10.1109/NABIC.2009.5393690.

Yang, Z., Deng, L., Wang, Y., & Liu, J. (2021). Aptenodytes forsteri optimization: Algorithm and applications. *Knowledge-Based Systems*, *232*, Article 107483. http://dx.doi.org/10.1016/j.knosys.2021.107483.

Zames, G. (1981). Genetic algorithms in search, optimization and machine learning. *Information Technology Journal*, *3*(1), 301. http://dx.doi.org/10.5860/choice.27-0936.

Zhan, Z.-H., Zhang, J., Lin, Y., Li, J.-Y., Huang, T., Guo, X.-Q., et al. (2021). Matrix-based evolutionary computation. *IEEE Transactions on Emerging Topics in Computational Intelligence*, *6*(2), 315–328. http://dx.doi.org/10.1109/TETCI.2020.3047410.

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. *Knowledge-Based Systems*, *261*, Article 110206. http://dx.doi.org/10.1016/j.knosys.2022.110206.

Zhang, Y.-H., Gong, Y.-J., Zhang, H.-X., Gu, T.-L., & Zhang, J. (2016). Toward fast niching evolutionary algorithms: A locality sensitive hashing-based approach. *IEEE Transactions on Evolutionary Computation*, *21*(3), 347–362. http://dx.doi.org/10.1109/TEVC.2016.2604362.

Zhang, J., & Sanderson, A. C. (2009). JADE: Adaptive differential evolution with optional external archive. *IEEE Transactions on Evolutionary Computation*, *13*(5), 945–958. http://dx.doi.org/10.1109/TEVC.2009.2014613.

Zhang, G., & Shi, Y. (2018). Hybrid sampling evolution strategy for solving single objective bound constrained problems. In *2018 IEEE congress on evolutionary computation* (pp. 1–7). IEEE, http://dx.doi.org/10.1109/CEC.2018.8477908.

Zhang, C., Zhou, W., Qin, W., & Tang, W. (2023). A novel UAV path planning approach: Heuristic crossing search and rescue optimization algorithm. *Expert Systems with Applications*, *215*, Article 119243. http://dx.doi.org/10.1016/j.eswa.2022.119243.

Zhao, F., Jiang, T., Xu, T., Zhu, N., et al. (2023). A co-evolutionary migrating birds optimization algorithm based on online learning policy gradient. *Expert Systems with Applications*, *228*, Article 120261. http://dx.doi.org/10.1016/j.eswa.2023.120261.

Zhao, W., Wang, L., & Zhang, Z. (2019a). Atom search optimization and its application to solve a hydrogeologic parameter estimation problem. *Knowledge-Based Systems*, *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. (2020). Artificial ecosystem-based optimization: A novel nature-inspired meta-heuristic algorithm. *Neural Computing and Applications*, *32*, 9383–9425. http://dx.doi.org/10.1007/s00521-019-04452-x.

Zhao, W., Zhang, Z., & Wang, L. (2020). Manta ray foraging optimization: An effective bio-inspired optimizer for engineering applications. *Engineering Applications of Artificial Intelligence*, *87*, Article 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. *Knowledge-Based Systems*, *251*, Article 109215. http://dx.doi.org/10.1016/j.knosys.2022.109215.

Zhong, X., You, Z., & Cheng, P. (2023). A hybrid optimization algorithm and its application in flight trajectory prediction. *Expert Systems with Applications*, *213*, Article 119082. http://dx.doi.org/10.1016/j.eswa.2022.119082.

Zhou, X., Ma, H., Gu, J., Chen, H., & Deng, W. (2022). Parameter adaptation-based ant colony optimization with dynamic hybrid mechanism. *Engineering Applications of Artificial Intelligence*, *114*, Article 105139. http://dx.doi.org/10.1016/j.engappai.2022.105139.