Applied Intelligence (2024) 54:959–984
https://doi.org/10.1007/s10489-023-05179-y

A variable population size opposition-based learning for differential evolution algorithm and its applications on feature selection

Le Wang[1] · Jiahang Li[2] · Xuefeng Yan[1]

Accepted: 12 November 2023 / Published online: 27 December 2023

Abstract
The opposition-based differential evolution (ODE) cannot adaptively adjust the number of individuals partake opposition-based learning, which makes it difficult to solve complex optimization problems. In this manuscript, we present an innovative approach for the treatment of variable population ODE (SASODE) by leveraging on adaptive parameters. The core idea of SASODE is to assign a jumping rate to each individual in the population, which is the key parameter that determines whether an individual enters a subpopulation or not. The initial rate assignment relies on the empirical mean of a normal distribution. During the iterative process, the mean is adjusted adaptively by taking into account the historical information of the individuals retained from the preceding generation. At the same time, the variation of this mean directly lead to changing the jumping rate of individuals and thus to adjusting the subpopulation size. In addition, the constant c and the Lehmer mean together maintain a balance between exploration and exploitation of SASODE. Experimental results show that the algorithm ranks first in the Wilcoxon test on 61 benchmarks and three optimization problems in three dimensions. Then, we confirm that SASODE can achieve an accuracy of 96% or even higher on the feature selection problem. Therefore, SASODE outperforms the other state-of-the-art algorithms compared in terms of convergence rate and accuracy.

Keywords Differential evolution · Parameter control · Opposition-based learning · Feature selection

1 Introduction

Nowadays, as the complexity of actual optimization problem increases, efficient and easy implementation algorithms are essential in solving these problems. Inspired by the various principles of nature, metaheuristic algorithms are widely used in various fields such as scheduling problems [1, 2], routing problems [3], data mining [4], feature selection [5, 6] and deep learning [7]. Based on different principles, the existing metaheuristic algorithms are divided into four categories [8–10]: Evolutionary computation, Swarm intelligence, Human-based algorithms and Physics-based algorithms. Swarm intelligence algorithms inspired by swarm behavior, and the collaborative effect of a population is greater than the sum of the effect of individuals. Evolutionary computation is derived from Darwinism, where algorithms are designed from an evolutionary perspective and perform genetic operators (crossover, mutation and selection) on the population to find global optimal. Evolutionary computation mainly include Genetic Algorithm (GA) [11] and Differential Evolution (DE) [12]. Algorithm such as Brain Storm Optimization (BSO) [13] is inspired by human social behavior. Sine Cosine Algorithm (SCA) [14] and Equilibrium Optimizer (EO) [15] are proposed based on the laws of natural physics.

DE is a classical evolutionary algorithm, it was proposed by Storn [12] and was inspired by the GA. The Differential Evolution (DE) algorithm is characterized by three fundamental parameters, namely the population size NP, the scaling factor F, and the crossover rate CR. The effectiveness of DE is contingent upon the utilization of three genetic operators and three crucial parameters. In [16], Li et al. introduced a variant of DE that integrates an elite preservation and mutation strategy. The utilization of elite preservation strengthens the exploitation capability while mutation strategies were employed to maintain a balance between exploration and

✉ Xuefeng Yan
xfyan@ecust.edu.cn

[1] Key Laboratory of Smart Manufacturing in Energy Chemical Process, East China University of Science and Technology, Shanghai 200237, China

[2] School of Mechanical Science & Engineering, Huazhong University of Science and Technology, Wuhan 430074, People's Republic of China

exploitation (EE) [17]. Zeng et al. [18] introduced a selection operator in DE with a novel approach that intended to minimize the effect of stagnation. The three candidate vectors may survive to the next generation if the algorithm is in a state of stagnation. Rosic et al. [19] proposed a hybrid firefly algorithm for DE (AHFADE). The AHFADE has the capability of adaptively adjusting parameter settings in order to choose a suitable mutation operator that can ensure a stable balance between the processes of diversification and intensification. According to Deng et al. [20], an adaptive mechanism for dimensional adjustment based on DE was incorporated to address the issue of premature convergence or stagnation, this mechanism greatly reduces the impact of dimensionality on algorithm performance.

Opposition-based learning (OBL) was proposed by Tizhoosh to improve the probability of identifying superior individuals in a population [21, 22]. In 2008, Rahnamayan et al. [23] proposed a new OBL-based DE variant, ODE, which for the first time used OBL in the DE initialization phase to accelerate the convergence of DE. Choi et al. [24] proposed a fast and efficient stochastic OBL (BetaCOBL) to control the degree of OBL solution, but the excessive computational complexity made it difficult to solve cost-sensitive optimization problems, so a second generation method iBetaCOBL was generated based on this method, using a linear diversity time metric to reduce the computational cost, and experimental results showed that the complexity of the algorithm can be reduced from $O(NP^2 \cdot D)$ to $O(NP \cdot D)$. iBetaCOBL-eig [25], a new version of iBetaCOBL, was developed in 2023, which improves DE from a dimensionality point of view by using multiple crossover operators based on eigenvectors, and similarly proved that the advantage of the algorithm performance. Of course, in addition to the combination with DE, OBL is now integrated with multiple approaches. Mohapatra and Mohapatra [26] combined OBL and random OBL (ROBL) with the Golden Jackal optimization algorithm, and statistical tests showed that the algorithm was optimal on benchmark functions and engineering optimization problems. Wang et al. [27] combined both OBL and the Q-learning with the heuristic algorithm, which not only improved the neighbourhood search capability of the algorithm, but also increased the probability of finding the optimal and greatly reduced the time cost.

However, research into OBL at this stage has shortcomings. First, the direct manipulation of the population in initialization phase or during iteration can ignore the diversity of individuals in the population. Second, as the algorithm iterates, the individuals that can produce the optimal solution will become more and more concentrated, and the population size for performing OBL operations theoretically needs to become smaller and smaller, which requires providing an adaptive subpopulation strategy to ensure the equilibrium of EE and find the optimal of the algorithm faster.

In the age of information, there has been a significant surge in the volume of data available for analysis, posing a formidable challenge for classification tasks due to the substantial increase in samples and features. Redundant features in the samples not only reduce the classification accuracy, but also increase the training time and computational complexity of the classification model [28], and the process of selecting a specific number of features for classification training is called data dimensionality reduction. There are two ways of data dimensionality reduction, one is feature extraction and the other is feature selection. The former is spatially scoped to produce a new dimensional mapping of features, and the latter is a reduction in the number of features to obtain a new feature subset [29]. As the complexity of the problem and the quantity of data dimensions augment, the number of features within the search space will exponentially expand, which makes it a challenging optimization problem. To solve this problem, most of the early studies used traditional greedy algorithms such as SFS and SBS to solve the problem, but as the difficulty of the problem increases, these methods tend to fall into local optimal and fail to find an effective and efficient solution. In recent years, heuristic algorithms have been widely used to solve it due to their powerful global search capability and robustness [30, 31]. There is also a lot of work on combining OBL with heuristic algorithms applied to feature selection [32, 33], but there is still a lot of room for development in the field of combining DE with OBL variants to jointly solve the feature selection problem. Therefore, in this paper, we improve the ODE and validate it on the feature selection problem.

Although there are many performance investigations for ODE, there are still some drawbacks in practice. According to our research, the traditional ODE may not obtain better performance because it ignores the individual differences.This paper presents an approach aimed at overcoming the constraints that characterize traditional OBL. The proposed strategy seeks to enhance OBL by shifting the focus from population-based changes to individual-based changes. The main principal contributions of this paper can be summarized as:

1. A strategic proposal is being presented to enhance the utilization of the OBL population to their utmost potential by emphasizing the strengths of each individual member of the population.
2. Adaptive parameter control is introduced in ODE to decide the size of the subpopulation according to the jumping rate of individuals and accelerate the convergence speed of evolution.
3. The concept of subpopulation survived individuals and historical information is proposed to guide the direction of SASODE optimization search.

4. Validation on several benchmarks and feature selection optimization problem and comparison with several competing algorithms.

The present article is organized as follows: The preliminary review of the DE and OBL is provided in Section 2. Section 3 presents the proposed SASODE. The experimental verification is discussed in detail in Section 4. The application of feature selection is demonstrated in Section 5. Lastly, the conclusion is drawn in Section 6.

2 Fundamentals

2.1 Differential evolution

As a population-based algorithm, the population undergoes three genetic operators (mutation, crossover and selection) in each iteration [12]. Here, the population is initialized as follow:

$$x_i^j = (x_{\max}^j - x_{\min}^j) \cdot rand(0,1) + x_{\min}^j \tag{1}$$

where $x_{\min}^j$ is the lower boundary. $x_{\max}^j$ is the upper boundary.

Then, DE employ mutation operator on the target vector. In SASODE, we use the most popular mutation DE/rand/1, it is defined as:

$$\mathbf{v}_i = \mathbf{x}_{r_1} + F \cdot (\mathbf{x}_{r_2} - \mathbf{x}_{r_3}) \tag{2}$$

where the indices r_1, r_2, r_3 are numbers randomly selected from $[1, NP]$. F is a positive parameter and scales the difference vector to control the size of search step.

After mutation, DE will perform the cross operator. CR is the crossover probability and the vectors $\mathbf{v}_i$ and $\mathbf{x}_i$ are crossed based on CR. The detail is outlined as follows:

$$\mathbf{u_i(j)} = \begin{cases} \mathbf{v_i}(j) & \text{if rand}(0,1) \le CR, or, j = j_{rand} \\ \mathbf{x_i}(j) & \text{otherwise} \end{cases} \tag{3}$$

Where the variable j_{rand} denotes an integer that is randomly chosen from the interval 1 to D, D is the dimension size.

The last step in each iteration of the DE is selection. A one-to-one selection between $\mathbf{x}_i$ and $\mathbf{u}_i$ takes place, guided by the fitness values. The selection approach can be elucidated as follows:

$$\mathbf{x}_i^{G+1} = \begin{cases} \mathbf{u}_i^G & \text{if fitness}(\mathbf{u}_i^G) < \text{fitness}(\mathbf{x}_i^G) \\ \mathbf{x}_i^G & \text{otherwise} \end{cases} \tag{4}$$

where $fitness(x)$ denotes the fitness function.

2.2 Opposition-based learning

Assume the set $P = \{x(1), x(2), ..., x(D)\}$, where $x(j) \in [a(j), b(j)]$, where $j = 1, 2, ..., D$. This set contains all points in a D-dimensional space. The set of opposite points can be defined as $\breve{P} = \{\breve{x}(1), \breve{x}(2), ..., \breve{x}(D)\}$. The specific formula is as follows.

$$\breve{x}(j) = a(j) + b(j) - x(j) \tag{5}$$

3 Proposed algorithm

Based on the characteristics of OBL and the evolutionary process of ODE, the advantages of individuals in the population are maximized in the proposed SASODE. Furthermore, the NFL demonstrates that no single algorithm can solve all problems [34], various problems necessitate distinct parameter configurations, which can be effectively resolved through the utilization of a self-adaptive approach to parameter setting in the iterative process of the algorithm.

3.1 Motivations

In ODE, the population jump in each generation is contingent upon the jumping rate, and provided that the requisite jumping condition is fulfilled, all members of the population produce their opposite individuals. This is a population-based strategy, however its utilization for the opposite population shows weakness during the later stages of evolution. Therefore, how to use the characteristics of individual information plays a key role in enhancing algorithm performance.

The research of meta-heuristic algorithms places significant emphasis on the aspect of parameter control [35], which has been discussed in Section 1. The primary parameter in ODE is the jumping rate. To regulate the number of individuals participating in OBL, subpopulation and adaptive parameter control mechanisms have been proposed for the attainment of SASODE via data exchange.

3.2 Subpopulation-based strategy

The most important difference between population-based and subpopulation-based is the number of opposite individuals in each generation. The detailed description is shown on Algorithm 1. From line 2 to line 5, we employ (1) to produce an initialized population that improves the diversity. From line 8 to line 10, the entire population participates in the opposite operation if a random number is less than Jr (Jr is generally taken as 0.3). Assume population size is 50, iteration is 100, the total number of individuals required to perform the opposition operation is approximately $0.3 \times 1000 \times 50$. From line 16 to line 22, the subpopulation is crated by using the

Algorithm 1 The difference between initialization, population-based and subpopulation-based.

Step 1 Population Initialization
Initialize $\mathbf{P}$ use (1)
$\mathbf{O}_{i,j} = a_j + b_j - \mathbf{P}_{i,j}, \quad i = 1, 2, ..., NP, \ j = 1, 2, ..., D.$
Select first NP individuals from $\{\mathbf{P} \cup \mathbf{O}\}$ to form the initial population.

Step 2 Population-based Opposition
while do
  if $rand < Jr$ **then**
    $\mathbf{O}_{i,j} = L_j + U_j - \mathbf{P}_{i,j}, \quad i = 1, 2, ..., NP, \ j = 1, 2, ..., D.$
    Select first NP individuals from $\{\mathbf{P} \cup \mathbf{O}\}$ enter the next generation.
  end if
end while

Step 3 Subpopulation-based Opposition
while do
  $\mathbf{Jr} = Gauss(\mu_J, 0.1)$
  $Jind = find(rand(size(\mathbf{Jr})) \leq \mathbf{Jr})$
  $\mathbf{SP} = \mathbf{P}(Jind, :)$
  for $k = 1 : length(Jind)$ **do**
    $\mathbf{OSP}(k, :) = min(\mathbf{SP}(k, :)) + max(\mathbf{SP}(k, :)) - \mathbf{SP}(k, :)$
  end for
  Select first NP fittest individuals from $\{\mathbf{P} \cup \mathbf{OSP}\}$ enter the next generation.
end while

probability. Specifically, each individual is accompanied by a jumping rate and the assignment of jumping rate follows a Gaussian distribution which the mean is μ_J and the variance is 0.1 (μ_J is generally taken as 0.3). The condition for an individual to enter a subpopulation is whether its jumping rate is greater than a random number. In each iteration, if the population size is 50, approximately 0.3×50 individuals perform the opposite operation. In this way, when the iteration reaches 1000, the total number of individuals involved in OBL is approximately $0.3 \times 50 \times 1000$. Obviously, population-based and subpopulation-based are very different in the methods of selecting the opposite individuals, but the number of times the opposite individuals are calculated is approximately the same. It indicates that the proposed subpopulation strategy leads to no extra computational effort.

The approach of subpopulation is motivated by the dependence mechanism of individuals, which manages the mutation operators and parameters in accordance with the fitness value of each individual, with the ultimate objective of enhancing the convergence and accuracy of DE [36]. The purpose of this paper is to solve complex single-objective optimization problems, and how to select suitable individuals to create subpopulations is the most important concern at present. In [37], subpopulations are created according to the size of fitness value. In [38], NBC uses a neighborhood subpopulation strategy to obtain clustering centers. To embed parameter control into the subpopulation strategy, SASODE creates subpopulations based on the size of individual jumping rate. The outlined steps are specified as follows:

1. $P = \{\mathbf{x}_i | i = 1, 2, ..., NP\}$ is the initialized population. The jumping rate corresponding to each individual $\mathbf{Jr} = \{j_1, j_2, ..., j_{NP}\}$ is generated according to the Gaussian distribution. More detailed description of this step is shown in Fig. 1.
2. A random number r_i between [0, 1] for each individual is generated according to the uniform distribution, and each random number with the jumping rate j_i is compared. If the former is smaller than the latter, the individual $\mathbf{x}_i$ is a member of the subpopulation SP. The individuals of

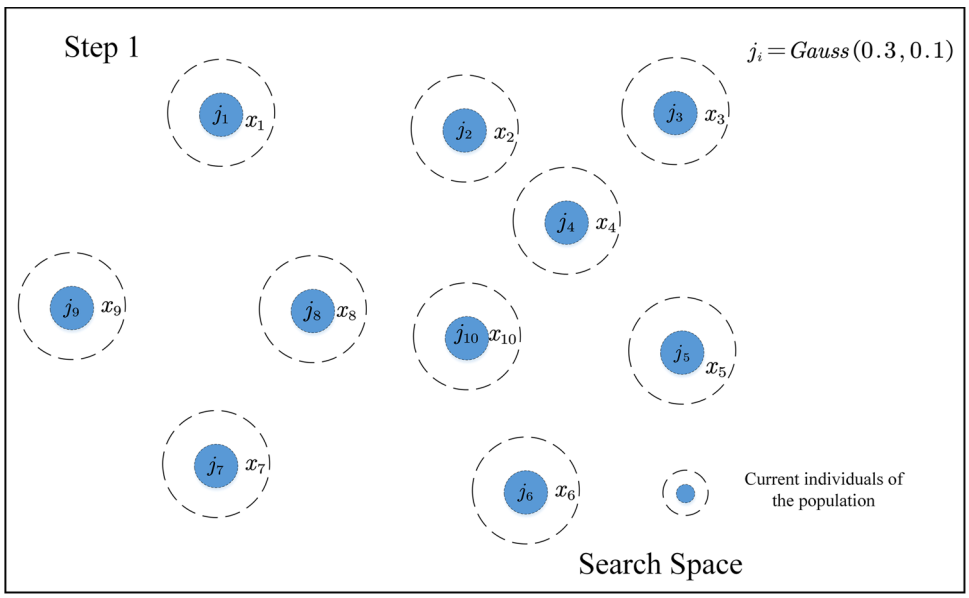


Fig. 1 Illustration of step 1 of the subpopulation strategy

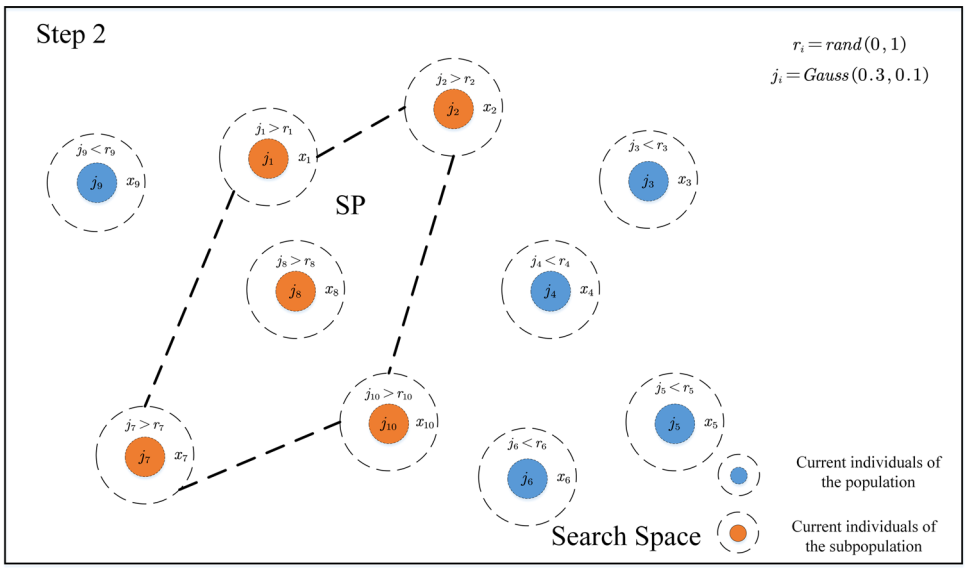


Fig. 2 Illustration of step 2 of the subpopulation strategy

the subpopulation are represented by different colored symbols in Fig. 2.

3. Create the opposite subpopulation OSP according to (5), combine the original population P with OSP. The surviving individuals are recorded as green yellow circles in Fig. 3, and select the top NP individuals with the best fitness into the next generation. S_J represents the jumping rate of individuals retained in the OSP and serves as a crucial parameter for regulating subpopulation size. S_J is a significant parameter that enables the adjustment of subpopulation size, and its values can be modified based on historical data from the preceding generation, it details in Section 3.3.

3.3 Self-adaptive parameter control

The jump rate of OBL is a random number between 0 and 1, but the jumping rate based on subpopulation is a vector. The vector $\mathbf{Jr} = \{j_1, j_2, ..., j_{NP}\}$ is generated by a Gaussian distribution, and the mean value of μ_J determines the location of the distribution of the jumping rate j_i for each individual. The larger the μ_J, the larger the value of j_i, and the greater the probability that an individual corresponding to j_i will enter the subpopulation at that time. It can be seen that μ_J can control the size of the subpopulation. In the late evolutionary stage, the parameter control mechanism determines the balance of algorithm about EE.

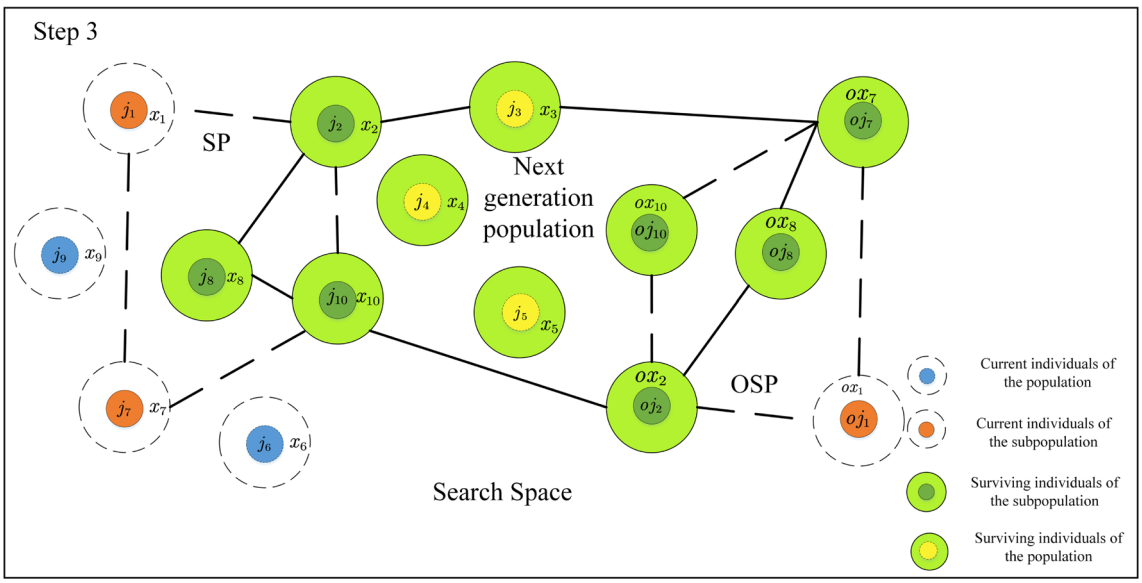


Fig. 3 Illustration of step 3 of the subpopulation strategy

According to the previous analysis, it is clear that there is a corresponding relationship between the size of subpopulations and μ_J. The literature [39] found that an increase in convergence rate was associated with a reduction in population diversity. To attain a more equitable equilibrium concerning diversity and convergence rate, Zhang et al. augmented the arithmetic mean employed for adaptive parameter computation by integrating the Lehmer mean paradigm [40]. The difference between the two is that the calculated value of the Lehmer is larger than the value of the arithmetic mean when the variables have the same value. The SASODE algorithm also extends this method. The specific calculation formula is as follows:

$$\text{Lehmer}(x_1, x_2, ...x_n) = \frac{x_1^2 + x_2^2 + ... + x_n^2}{x_1 + x_2 + ... + x_n} \tag{6}$$

where n represents the size of S_J, $x_i, i = 1, 2, ..., n$ are all the elements of S_J in each generation.

During the iteration, the size of j_i directly determines whether the individual performs the opposite operation or not. As mentioned before, the jumping rate j_i of an individual is generated by Gaussian random numbers, and a Gaussian distribution with mean μ_J and variance 0.1 can generate the jumping rate randomly and without outliers according to the location parameter μ_J, as shown in the formula for (7). The same operation as SASODE can also be found in the algorithms JADE [40], SHADE [41], LSHADE [42] and so on.

$$j_i = Gauss(\mu_J, 0.1) \tag{7}$$

In the above formula, the initial μ_J is 0.3, which is updated in the evolutionary process using the following formula [23]:

$$\mu_J^{G+1} = (1 - c) \cdot \mu_J^G + c \cdot \text{Lehmer}(S_J) \tag{8}$$

where S_J indicates the set of individual jumping rate j_i of surviving individuals in the opposite subpopulation OSP. c is a positive number between 0 and 1 that makes a linear combination of μ_J and Lehmer(S_J) to reach an equilibrium state.

The (8) consists of two parts, μ_J^G and Lehmer(S_J). The former is the historical information left by the previous iterations, which represents the global information in the previous generations. The latter is the surviving individuals in the opposite population, and these surviving individuals represent the local information that survived the current generation to the next iteration. In order to avoid that μ_J^{G+1}can have a violent oscillation, which causes the individuals performing the backward learning to lose control, making μ_J^{G+1} infinitely larger or smaller affecting the performance of the algorithm. The introduction of a constant c in this formula and assigning computational weights to the two components enables a stable state to be reached for both global and local information. Algorithm 2 is the pseudo code of SASODE.

Algorithm 2 SASODE.

Step 1 Initialization
Set up $F, CR, NP, MaxFEs, G, c, \mu_J$. Generate the population **P0** of NP individuals, $\mathbf{P0} = \{\mathbf{x}_1^G, \mathbf{x}_2^G, ..., \mathbf{x}_{NP}^G\}$, with $\mathbf{x}_i^G = \{x_{i,1}^G, x_{i,2}^G, ..., x_{i,D}^G\}$, each individual is in range $[a, b]$.
for $i = 1 : NP$ **do**
  $\mathbf{OP}(i, :) = a + b - \mathbf{P0}(i, :)$
end for
Select first NP fittest individuals from {**P0**, **OP**} as **P**.

Step 2 Evolutionary Process
while The termination criterion is satisfied **do**
  Step 2.1 Mutation
  for $i = 1 : NP$ **do**
    $\mathbf{Jr} = Gaussian(\mu_J, 0.1)$, randomly choose $i \neq r_1 \neq r_2 \neq r_3$ from $[1, NP]$.
    $\mathbf{V}^G(i, :) = \mathbf{P}(r_1, :) + F \cdot (\mathbf{P}(r_2, :) - \mathbf{P}(r_3, :))$
    Step 2.2 Crossover
    Generate $j_{rand} = randint(1, D)$
    for $j = 1 : D$ **do**
      if $j = j_{rand} || rand(0, 1) < CR$ **then**
        $\mathbf{U}^G(i, j) = \mathbf{V}^G(i, j)$
      else
        $\mathbf{U}^G(i, j) = \mathbf{P}^G(i, j)$
      end if
    end for
    Step 2.3 Selection
    if $f(\mathbf{U}^G(i, :)) < f(\mathbf{P}^G(i, :))$ **then**
      $\mathbf{P}(i, :)' = \mathbf{U}^G(i, :)$
    else
      $\mathbf{P}(i, :)' = \mathbf{P}^G(i, :)$
    end if
  end for
  $\mathbf{Pt} = \mathbf{P}'$
  Step 3 Opposite Operation
  $Jind = find(rand(size(\mathbf{Jr})) \leq \mathbf{Jr})$
  $\mathbf{SP} = \mathbf{Pt}(Jind, :)$
  for $k = 1 : length(Jind)$ **do**
    $\mathbf{OSP}(k, :) = min(\mathbf{SP}(k, :)) + max(\mathbf{SP}(k, :)) - \mathbf{SP}(k, :)$
  end for
  Select first NP fittest individuals from {**Pt**, **OSP**}, record survive individuals as $\mathbf{P}^{G+1}$, record jumping rate **Jr** of surviving individuals from **OSP** as S_J.
  Step 4 Parameter Control
  $\mu_J^{G+1} = (1 - c) \cdot \mu_J^G + c \cdot \text{Lehmer}(S_J)$
  $G = G + 1$
end while

4 Experimental verification

To further validate SASODE's performance, we compared different test functions, unimodal and multimodal functions, verified the validation algorithm on the CEC2017 optimization problem, and tested three engineering problems.

4.1 Experimental on unimodal and multimodal test functions

To validate the effectiveness of SASODE, a total of 32 widely-recognized benchmark functions have been meticulously chosen for the purpose of conducting a comprehensive numerical experiment. Specifically, 15 unimodal functions and 17 multimodal functions pertaining to the optimization of real-valued data sets have been carefully selected, as referenced in works by Yao et al. [43] and Askari et al. [44]. It is noteworthy that the aforementioned benchmark functions, i.e., $f_1 - f_{15}$ and $f_{16} - f_{32}$ respectively represent unimodal and multimodal functions, the specific details regarding the aforementioned benchmark functions are presented in Table 1.

4.1.1 Parameter setting

This paper aims to compare SASODE with six swarm intelligence algorithms, namely GWO [45], WOA [46], MFO [47], SCA [14], SSA [48], and HBO [44], for the aforementioned functions. To ensure the reliability of the algorithmic results, the population size and maximum number of iterations of the comparison algorithm have been fixed at 50 and 1000, respectively. However, the rest of the parameter settings have been retained from its original research. All experiments were done on the Windows 10 operating system, MATLAB R2018b.

4.1.2 Comparison metrics

The Wilcoxon test for pairwise comparison was utilized to compare the performance of SASODE and the comparison algorithm. This statistical analysis was conducted with a significant level of $\alpha = 0.05$ according to [49]. To denote the correlation between the algorithm under consideration and the comparative algorithms, we employed the symbols $+$, $-$, and $=$. A detailed explanation of the specific application of these symbols is provided below.

1. $+$: The solutions of SASODE performs better than the comparison algorithm.
2. $=$: The solutions of SASODE performs approximately than the comparison algorithm.
3. $-$: The solutions of SASODE performs worse than the comparison algorithm.

4.1.3 The results of unimodal functions

This paper sets four dimensions of 10, 30, 50 and 100 for experimental comparison, and the 30D results are displayed visually by iteration curves.

From the results of testing functions f1-f15 in Tables 2, 3, and 4, SASODE has the best performance on 100D, especially on functions f4, f5, f6, f8, f9, f11, f12, f13, and f14, with better results for both mean and standard deviation than other algorithms. The SASODE algorithm may have certain advantages in solving real-world high-dimensional

Table 1 Unimodal and multimodal test functions

Unimodal				Multimodal			
f.no	Name	Range	Optimum	f.no	Name	Range	Optimum
f1	Sphere	[-100,100]	0	f16	Schwefel′s 2.26	[-500,500]	0
f2	Powell Sum	[-1,1]	0	f17	Rastrigin	[-5.12,5.12]	0
f3	Schwefel′s 2.20	[-100,100]	0	f18	Periodic	[-10,10]	0.9
f4	Schwefel′s 2.21	[-100,100]	0	f19	Qing	[-500,500]	0
f5	Step	[-100,100]	0	f20	Alpine N. 1	[-10,10]	0
f6	Stepint	[-5.12,5.12]	-155	f21	Xin-She Yang	[-5,5]	0
f7	Schwefel′s 2.22	[-100,100]	0	f22	Ackley	[-32,32]	0
f8	Schwefel′s 2.23	[-10,10]	0	f23	Trignometric 2	[-500,500]	0
f9	Rosenbrock	[-30,30]	0	f24	Salomon	[-100,100]	0
f10	Brown	[-1,4]	0	f25	Styblinski-Tang	[-5,5]	-1174.98
f11	Dixon and Price	[-10,10]	0	f26	Griewank	[-100,100]	0
f12	Powell Singular	[-4,5]	0	f27	Xin-She Yang N. 4	[-10,10]	-1
f13	Xin-She Yang	[-20,20]	0	f28	Xin-She Yang N. 2	[-2pi,2pi]	0
f14	Perm 0,D,Beta	[-5,5]	0	f29	Gen. Penalized	[-50,50]	0
f15	Sum Squares	[-10,10]	0	f30	Penalized	[-50,50]	0
				f31	Michalewics	[0,pi]	-29.6309
				f32	Quartic Noise	[-1.28,1.28]	0

Table 2 Results of the unimodal functions and multimodal functions from SASODE and six other metaheuristic algorithms on 10D

Function	Stats	SASODE	WOA	GWO	HBO	SSA	SCA	MFO
f1	Mean	2.07E-100	**6.91E-163**	2.33E-116	1.51E-59	6.04E-10	3.39E-23	2.87E-30
	Std	1.13E-99	**3.14E-162**	8.49E-116	2.62E-59	2.70E-10	1.86E-22	5.47E-30
f2	Mean	1.09E-88	1.56E-230	**2.04E-234**	4.72E-117	1.34E-07	1.14E-52	1.10E-63
	Std	5.80E-88	0.00E+00	0.00E+00	**1.97E-116**	7.89E-08	3.86E-52	5.41E-63
f3	Mean	1.43E-12	**3.00E-104**	5.64E-66	1.59E-36	3.61E-02	9.17E-18	5.10E-18
	Std	7.82E-12	**1.25E-103**	1.20E-65	1.85E-36	1.97E-01	2.86E-17	8.82E-18
f4	Mean	2.12E-26	4.00E-01	**2.15E-37**	2.43E-09	1.61E-05	2.05E-07	4.23E-01
	Std	1.04E-25	1.08E+00	**6.57E-37**	8.70E-09	5.56E-06	4.98E-07	6.77E-01
f5	Mean	0.00E+00	3.36E-05	9.02E-07	**1.03E-34**	6.99E-10	3.13E-01	1.33E-30
	Std	0.00E+00	3.04E-05	2.99E-07	**5.63E-34**	2.70E-10	1.30E-01	2.38E-30
f6	Mean	**-3.50E+01**	**-3.50E+01**	-3.40E+01	**-3.50E+01**	**-3.50E+01**	-3.41E+01	**-3.50E+01**
	Std	**0.00E+00**	**0.00E+00**	2.48E+00	**0.00E+00**	**0.00E+00**	7.76E-01	**0.00E+00**
f7	Mean	6.15E-22	**8.31E-108**	1.17E-65	1.36E-36	1.84E+02	3.82E-18	4.67E+01
	Std	3.37E-21	**4.07E-107**	2.94E-65	1.88E-36	4.51E+02	8.87E-18	6.29E+01
f8	Mean	**1.24E-208**	0.00E+00	0.00E+00	4.16E-181	9.62E-58	3.13E-88	1.27E-77
	Std	0.00E+00	0.00E+00	0.00E+00	0.00E+00	2.70E-57	**1.39E-87**	6.98E-77
f9	Mean	**1.98E+00**	1.07E+01	6.55E+00	1.64E+00	1.15E+02	7.18E+00	2.35E+02
	Std	2.40E+00	2.68E+01	**6.53E-01**	1.72E+00	2.66E+02	3.35E-01	7.63E+02
f10	Mean	1.09E-77	**1.46E-167**	3.27E-119	1.26E-62	1.55E-12	2.29E-29	3.48E-32
	Std	5.99E-77	0.00E+00	**1.60E-118**	3.76E-62	6.24E-13	7.21E-29	1.67E-31
f11	Mean	1.78E-01	**6.67E-01**	**6.67E-01**	6.47E-02	6.24E-01	**6.67E-01**	4.39E+01
	Std	3.00E-01	1.97E-03	2.49E-05	1.98E-01	1.69E-01	**4.22E-05**	9.84E+01
f12	Mean	**1.07E-30**	1.18E-05	3.35E-07	2.23E-05	1.44E-03	3.50E-06	1.39E+01
	Std	**5.89E-30**	8.29E-06	5.53E-07	2.00E-05	1.05E-03	1.04E-05	2.77E+01
f13	Mean	-1.00E+00	**-9.33E-01**	1.83E-70	7.57E-78	7.57E-78	7.57E-78	7.57E-78
	Std	0.00E+00	2.54E-01	1.98E-70	**1.95E-93**	**1.95E-93**	**1.95E-93**	**1.95E-93**
f14	Mean	3.61E+00	5.48E+02	8.27E+02	**2.62E+00**	2.14E+01	4.50E+01	1.77E+01
	Std	1.53E+01	9.77E+02	1.66E+03	**3.16E+00**	3.56E+01	4.42E+01	5.87E+01
f15	Mean	1.33E-47	**4.32E-168**	1.32E-117	1.64E-61	4.45E-11	3.37E-27	3.33E+00
	Std	7.29E-47	0.00E+00	**6.95E-117**	3.62E-61	1.98E-11	1.20E-26	1.83E+01
f16	Mean	**1.27E-05**	9.46E+01	1.45E+02	1.27E-05	1.48E+02	1.94E+02	8.86E+01
	Std	**0.00E+00**	5.58E+01	3.64E+01	0.00E+00	3.03E+01	1.35E+01	3.18E+01
f17	Mean	0.00E+00	0.00E+00	1.40E-01	0.00E+00	1.76E+01	**1.93E-01**	1.99E+01
	Std	0.00E+00	0.00E+00	7.65E-01	0.00E+00	5.50E+00	**9.40E-01**	1.02E+01
f18	Mean	9.00E-01	**9.84E-01**	1.10E+00	1.01E+00	1.00E+00	1.08E+00	1.52E+00
	Std	**4.52E-16**	4.73E-02	2.76E-01	2.97E-03	3.04E-12	2.84E-01	3.62E-01
f19	Mean	2.90E-17	3.26E-01	7.62E+00	3.36E-06	1.30E-03	5.40E+01	**5.98E-27**
	Std	1.58E-16	4.47E-01	1.13E+01	1.84E-05	5.91E-03	2.60E+01	**2.30E-26**
f20	Mean	3.13E-05	3.60E-01	4.15E-05	**6.48E-09**	2.04E-01	5.37E-08	1.48E-01
	Std	1.29E-04	7.38E-01	9.31E-05	**3.12E-08**	4.00E-01	2.90E-07	8.11E-01
f21	Mean	**6.23E-49**	2.04E-04	1.88E-61	3.64E-38	2.52E-02	1.30E-09	5.58E-03
	Std	**1.42E-48**	1.09E-03	9.26E-61	1.82E-37	4.29E-02	6.98E-09	1.86E-02
f22	Mean	1.78E-12	2.19E-15	**3.38E-15**	2.66E-15	6.55E-01	1.90E-07	3.85E-02
	Std	9.75E-12	**1.80E-15**	1.45E-15	0.00E+00	8.72E-01	1.04E-06	2.11E-01
f23	Mean	**1.00E+00**	8.65E+00	3.84E+00	1.00E+00	1.38E+01	1.58E+01	6.79E+00
	Std	**0.00E+00**	6.54E+00	1.43E+00	0.00E+00	7.80E+00	2.95E+00	5.31E+00

Table 2 continued

Function	Stats	SASODE	WOA	GWO	HBO	SSA	SCA	MFO
f24	Mean	**6.00E-54**	1.27E-01	9.99E-02	9.99E-02	1.97E-01	9.99E-02	4.97E-01
	Std	**3.27E-53**	5.83E-02	7.58E-11	2.00E-08	5.56E-02	1.97E-07	2.36E-01
f25	Mean	-3.91E+02	-3.87E+02	-3.59E+02	**-3.92E+02**	-3.50E+02	-3.10E+02	-3.59E+02
	Std	3.59E+00	1.01E+01	1.76E+01	**0.00E+00**	1.93E+01	2.40E+01	1.93E+01
f26	Mean	**0.00E+00**	5.85E-02	1.45E-02	0.00E+00	2.48E-01	5.26E-02	1.50E-01
	Std	**0.00E+00**	1.09E-01	1.75E-02	0.00E+00	1.40E-01	1.23E-01	9.97E-02
f27	Mean	**-1.00E+00**	-6.67E-02	3.48E-07	6.54E-12	7.43E-16	2.31E-04	3.46E-06
	Std	**0.00E+00**	2.54E-01	1.33E-06	2.74E-11	4.51E-16	7.52E-05	1.32E-05
f28	Mean	**0.00E+00**	6.46E-04	1.20E-03	5.66E-04	2.14E-03	2.57E-03	2.72E-03
	Std	**0.00E+00**	2.41E-04	1.27E-03	1.36E-10	2.42E-04	1.76E-04	2.18E-04
f29	Mean	7.60E-28	1.11E-03	6.65E-03	**1.99E-32**	1.83E-03	2.50E-01	3.66E-03
	Std	4.00E-27	2.81E-03	2.53E-02	**3.47E-32**	4.16E-03	6.44E-02	5.27E-03
f30	Mean	9.42E-31	3.21E-03	2.19E-03	**4.73E-32**	1.38E-01	7.91E-02	9.33E-02
	Std	4.90E-30	1.01E-02	5.80E-03	**3.34E-34**	3.77E-01	4.08E-02	1.85E-01
f31	Mean	-9.08E+00	-5.78E+00	-7.80E+00	**-9.66E+00**	-7.23E+00	-4.04E+00	-7.91E+00
	Std	5.84E-01	6.66E-01	1.05E+00	8.54E-03	**8.98E-01**	5.56E-01	8.07E-01
f32	Mean	4.94E-03	7.61E-04	4.21E-04	2.26E-03	**6.39E-03**	1.91E-03	6.58E-03
	Std	**1.64E-03**	8.25E-04	3.15E-04	9.06E-04	5.15E-03	1.75E-03	3.46E-03

Table 3 Results of the unimodal functions and multimodal functions from SASODE and six other metaheuristic algorithms on 50D

Function	Stats	SASODE	WOA	GWO	HBO	SSA	SCA	MFO
f1	Mean	5.49E-89	**6.37E-158**	8.89E-44	2.16E-10	8.98E-08	1.85E+02	6.03E+03
	Std	2.95E-88	**3.33E-157**	1.14E-43	4.48E-10	2.46E-08	2.93E+02	8.95E+03
f2	Mean	3.95E-19	**1.52E-228**	4.24E-182	1.31E-34	4.82E-07	6.04E-04	4.18E-09
	Std	1.61E-18	**0.00E+00**	0.00E+00	3.03E-34	2.91E-07	1.06E-03	1.01E-08
f3	Mean	6.74E-29	**1.85E-104**	4.62E-25	1.85E-07	3.89E+01	1.55E-01	2.47E+02
	Std	3.55E-28	**9.72E-104**	4.48E-25	3.96E-07	2.03E+01	3.36E-01	1.86E+02
f4	Mean	**4.06E-24**	5.69E+01	2.28E-09	1.78E+01	1.70E+01	5.92E+01	8.32E+01
	Std	**2.22E-23**	2.79E+01	3.54E-09	3.61E+00	3.40E+00	7.18E+00	3.66E+00
f5	Mean	**0.00E+00**	2.92E-01	2.34E+00	1.12E-10	9.17E-08	7.40E+01	9.04E+03
	Std	**0.00E+00**	2.02E-01	5.67E-01	1.29E-10	2.96E-08	9.07E+01	8.85E+03
f6	Mean	**-2.75E+02**	-2.75E+02	-2.15E+02	-2.75E+02	-2.17E+02	-1.50E+02	-2.75E+02
	Std	**0.00E+00**	0.00E+00	1.29E+01	0.00E+00	1.57E+01	7.98E+00	2.01E+00
f7	Mean	2.96E-27	**4.42E-101**	7.10E-25	4.20E-07	2.67E+39	1.28E-01	7.83E+02
	Std	7.43E-27	**2.21E-100**	6.22E-25	9.71E-07	1.45E+40	2.28E-01	2.88E+02
f8	Mean	7.56E-294	**0.00E+00**	5.18E-135	1.04E-16	9.32E-07	4.52E+07	3.61E+03
	Std	**0.00E+00**	0.00E+00	2.44E-134	3.15E-16	3.42E-06	7.61E+07	6.15E+03
f9	Mean	**3.71E+01**	4.75E+01	4.70E+01	1.18E+02	1.76E+02	1.31E+06	2.70E+06
	Std	1.89E+01	5.47E-01	**7.59E-01**	5.52E+01	2.90E+02	2.27E+06	1.46E+07
f10	Mean	4.02E-87	**1.12E-159**	2.00E-46	3.97E-13	2.78E-10	6.54E-02	1.52E+02
	Std	1.53E-86	**6.10E-159**	2.57E-46	7.47E-13	1.08E-10	1.32E-01	1.69E+02
f11	Mean	6.44E-01	**6.67E-01**	6.67E-01	2.13E+00	7.80E+00	4.22E+03	3.69E+05
	Std	1.22E-01	3.57E-05	**7.13E-07**	2.19E+00	9.33E+00	7.53E+03	4.30E+05
f12	Mean	1.60E-06	**5.94E-08**	9.31E-06	1.68E-01	6.95E+00	1.50E+02	3.08E+03
	Std	3.73E-06	**3.25E-07**	1.15E-05	2.42E-01	4.19E+00	2.90E+02	2.60E+03

Table 3 continued

Function	Stats	SASODE	WOA	GWO	HBO	SSA	SCA	MFO
f13	Mean	5.99E-266	-1.33E-01	3.73E-169	0.00E+00	8.32E-280	**2.64E-289**	0.00E+00
	Std	**0.00E+00**	3.46E-01	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00
f14	Mean	3.49E+00	6.51E+02	4.13E+02	**1.39E+00**	2.05E+01	4.20E+01	5.81E+00
	Std	1.87E+01	6.62E+02	1.14E+03	**1.48E+00**	3.60E+01	2.91E+01	1.83E+01
f15	Mean	4.28E-42	**4.22E-158**	1.38E-44	4.31E-11	4.41E+00	2.14E+01	2.21E+03
	Std	2.34E-41	**1.53E-157**	1.92E-44	1.01E-10	4.50E+00	5.27E+01	1.38E+03
f16	Mean	**2.03E+01**	9.94E+01	2.34E+02	2.61E+01	1.69E+02	3.18E+02	1.49E+02
	Std	3.94E+01	5.47E+01	**1.69E+01**	9.26E+00	1.94E+01	7.41E+00	2.50E+01
f17	Mean	**0.00E+00**	3.79E-15	7.77E-01	2.12E+01	8.08E+01	7.86E+01	3.17E+02
	Std	**0.00E+00**	2.08E-14	2.50E+00	3.31E+00	2.39E+01	6.76E+01	6.00E+01
f18	Mean	**9.07E-01**	1.03E+00	2.32E+00	4.20E+00	1.00E+00	1.09E+01	7.05E+00
	Std	**2.60E-02**	3.29E-01	2.98E+00	9.02E-01	1.72E-10	2.41E+00	1.43E+00
f19	Mean	4.52E+02	3.86E+03	1.07E+04	**3.37E-02**	7.60E+01	6.08E+08	6.29E+09
	Std	3.52E+02	1.67E+03	2.59E+03	**1.73E-01**	1.04E+02	1.09E+09	1.91E+10
f20	Mean	5.43E-03	**1.56E-101**	7.18E-05	1.25E-06	7.27E+00	2.56E+00	1.23E+01
	Std	6.29E-03	**8.56E-101**	2.26E-04	1.48E-06	2.58E+00	4.50E+00	8.43E+00
f21	Mean	1.01E-19	1.21E-02	**2.83E-43**	2.23E+01	1.47E+05	2.75E+07	1.58E+20
	Std	3.62E-19	5.90E-02	**1.55E-42**	1.19E+02	5.30E+05	1.51E+08	6.48E+20
f22	Mean	1.48E-15	**1.60E-15**	3.01E-14	1.76E-06	3.12E+00	1.55E+01	1.95E+01
	Std	1.70E-15	2.31E-15	**4.06E-15**	1.64E-06	8.19E-01	8.02E+00	7.43E-01
f23	Mean	**1.00E+00**	1.22E+02	6.08E+01	2.75E+00	4.29E+02	2.07E+03	1.68E+05
	Std	**0.00E+00**	3.01E+01	7.61E+00	1.57E+00	1.11E+02	3.35E+03	2.20E+05
f24	Mean	**3.24E-02**	1.27E-01	1.97E-01	5.20E-01	2.71E+00	1.70E+00	1.48E+01
	Std	4.69E-02	6.40E-02	1.83E-02	**7.13E-02**	4.33E-01	9.28E-01	4.47E+00
f25	Mean	-1.88E+03	-1.91E+03	-1.40E+03	**-1.96E+03**	-1.67E+03	-8.73E+02	-1.68E+03
	Std	**1.07E+02**	1.03E+02	8.97E+01	5.16E+00	4.67E+01	6.18E+01	6.73E+01
f26	Mean	**0.00E+00**	0.00E+00	1.10E-03	3.29E-04	3.60E-03	7.05E-01	2.82E+00
	Std	**0.00E+00**	0.00E+00	4.25E-03	1.80E-03	5.12E-03	3.36E-01	2.48E+00
f27	Mean	**-7.55E-01**	-2.00E-01	1.36E-23	6.44E-23	2.10E-24	9.13E-16	3.65E-19
	Std	4.28E-01	4.07E-01	**1.66E-23**	1.95E-22	1.15E-23	1.00E-15	8.76E-19
f28	Mean	4.88E-18	**1.52E-20**	4.74E-12	3.40E-19	1.65E-18	9.80E-16	1.19E-19
	Std	2.55E-17	**6.18E-21**	1.14E-11	1.21E-19	4.84E-18	1.23E-15	1.65E-20
f29	Mean	**1.35E-32**	3.86E-01	1.80E+00	7.32E-04	5.85E+01	5.12E+06	8.21E+07
	Std	**5.57E-48**	1.82E-01	3.25E-01	2.79E-03	1.96E+01	1.25E+07	1.99E+08
f30	Mean	**9.42E-33**	6.77E-03	9.02E-02	2.07E-03	9.86E+00	1.20E+06	2.56E+07
	Std	**2.78E-48**	4.36E-03	3.18E-02	1.14E-02	3.84E+00	2.76E+06	7.81E+07
f31	Mean	-2.49E+01	-1.70E+01	-1.96E+01	-2.74E+01	-2.62E+01	-1.08E+01	**-3.39E+01**
	Std	2.52E+00	2.08E+00	5.85E+00	1.58E+00	2.66E+00	**1.03E+00**	2.83E+00
f32	Mean	**9.80E-03**	2.04E-03	1.45E-03	3.14E-02	3.09E-01	6.90E-01	1.35E+01
	Std	3.11E-03	2.11E-03	**7.13E-04**	7.80E-03	9.60E-02	5.77E-01	3.14E+01

Table 4 Results of the unimodal functions and multimodal functions from SASODE and six other metaheuristic algorithms on 100D

Function	Stats	SASODE	WOA	GWO	HBO	SSA	SCA	MFO
f1	Mean	9.11E-65	**2.58E-156**	1.98E-29	1.66E-02	2.29E+00	6.84E+03	3.09E+04
	Std	4.99E-64	**8.51E-156**	2.12E-29	3.00E-02	2.12E+00	5.69E+03	1.39E+04
f2	Mean	3.80E-16	**1.53E-224**	4.83E-137	3.64E-18	8.81E-07	6.39E-02	5.35E-03
	Std	2.06E-15	**0.00E+00**	2.64E-136	1.17E-17	5.15E-07	4.18E-02	2.24E-02
f3	Mean	1.22E-20	**1.57E-102**	5.23E-17	4.66E-02	2.35E+02	1.73E+01	8.20E+02
	Std	3.80E-20	**7.12E-102**	2.42E-17	4.33E-02	5.53E+01	1.62E+01	2.22E+02
f4	Mean	**2.37E-42**	6.69E+01	2.46E-03	5.25E+01	2.60E+01	8.70E+01	9.25E+01
	Std	**1.30E-41**	2.80E+01	3.68E-03	5.30E+00	3.60E+00	2.48E+00	2.09E+00
f5	Mean	**0.00E+00**	1.27E+00	9.08E+00	2.92E-02	2.56E+00	4.84E+03	3.26E+04
	Std	**0.00E+00**	3.14E-01	8.92E-01	4.81E-02	1.98E+00	3.80E+03	1.29E+04
f6	Mean	**-5.75E+02**	-5.75E+02	-3.54E+02	-5.75E+02	-3.77E+02	-2.41E+02	-5.68E+02
	Std	**0.00E+00**	0.00E+00	1.40E+01	0.00E+00	3.22E+01	1.23E+01	9.29E+00
f7	Mean	1.13E-17	**9.04E-103**	5.92E-17	7.28E-02	9.28E+76	1.60E+01	2.52E+03
	Std	6.13E-17	**3.37E-102**	3.48E-17	6.92E-02	3.68E+77	1.76E+01	3.62E+02
f8	Mean	**0.00E+00**	0.00E+00	4.42E-84	1.79E-01	1.22E+01	3.73E+09	1.52E+09
	Std	**0.00E+00**	0.00E+00	2.34E-83	2.19E-01	3.14E+01	2.31E+09	2.80E+09
f9	Mean	**8.74E+01**	9.75E+01	9.73E+01	6.14E+02	2.34E+03	6.91E+07	6.02E+07
	Std	**2.61E+01**	4.43E-01	7.51E-01	5.58E+02	1.93E+03	4.11E+07	5.24E+07
f10	Mean	3.57E-73	**1.44E-158**	5.19E-32	6.32E-05	1.67E+01	3.64E+00	2.14E+03
	Std	1.95E-72	**6.65E-158**	6.37E-32	6.12E-05	2.02E+01	2.04E+00	8.71E+02
f11	Mean	**6.67E-01**	6.67E-01	6.67E-01	4.06E+01	1.28E+02	8.09E+05	2.25E+06
	Std	4.40E-05	5.16E-05	**1.94E-06**	9.61E+00	4.31E+01	4.34E+05	1.64E+06
f12	Mean	**3.46E-23**	8.13E-19	8.21E-06	7.80E+01	3.67E+01	2.11E+03	1.13E+04
	Std	**1.62E-22**	4.45E-18	1.19E-05	1.15E+02	1.55E+01	1.57E+03	5.19E+03
f13	Mean	**0.00E+00**	0.00E+00	2.43E-245	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	0.00E+00
f14	Mean	**1.77E+00**	5.76E+02	3.55E+02	1.85E+00	1.36E+01	4.42E+01	4.33E+02
	Std	**7.28E+00**	1.01E+03	1.08E+03	1.76E+00	2.51E+01	3.05E+01	1.62E+03
f15	Mean	1.94E-47	**8.36E-154**	7.43E-30	7.90E-03	1.17E+02	1.86E+03	1.69E+04
	Std	1.04E-46	**4.43E-153**	9.08E-30	1.28E-02	3.78E+01	1.54E+03	7.08E+03
f16	Mean	**5.70E+01**	9.60E+01	2.63E+02	7.43E+01	1.75E+02	3.49E+02	1.71E+02
	Std	7.63E+01	5.16E+01	3.64E+01	**1.53E+01**	1.96E+01	4.71E+00	2.61E+01
f17	Mean	**0.00E+00**	3.79E-15	3.01E-01	9.60E+01	1.76E+02	2.36E+02	7.57E+02
	Std	**0.00E+00**	2.08E-14	1.16E+00	1.60E+01	4.84E+01	1.35E+02	6.71E+01
f18	Mean	**9.12E-01**	1.07E+00	1.87E+00	1.12E+01	1.11E+00	2.78E+01	1.44E+01
	Std	**4.84E-02**	6.67E-01	6.72E-01	4.29E+00	2.80E-01	3.12E+00	1.74E+00
f19	Mean	2.67E+04	7.40E+04	1.47E+05	**6.39E+03**	6.59E+04	4.05E+10	6.84E+10
	Std	1.90E+04	1.56E+04	1.73E+04	**7.24E+03**	6.13E+04	2.15E+10	5.52E+10
f20	Mean	1.92E-02	**5.15E-105**	3.72E-04	5.63E-02	2.30E+01	1.40E+01	3.97E+01
	Std	2.39E-02	**2.02E-104**	6.94E-04	8.26E-02	5.25E+00	9.50E+00	1.05E+01
f21	Mean	7.83E-16	6.12E-04	**3.35E-27**	2.00E+18	1.85E+15	3.42E+33	9.62E+50
	Std	4.06E-15	1.84E-03	**1.83E-26**	1.09E+19	6.10E+15	1.87E+34	5.00E+51
f22	Mean	**2.66E-15**	2.66E-15	1.11E-13	6.74E-01	7.01E+00	1.97E+01	1.98E+01
	Std	**1.87E-15**	2.64E-15	9.35E-15	5.76E-01	1.39E+00	3.47E+00	2.55E-01
f23	Mean	**1.00E+00**	2.52E+02	1.79E+02	1.35E+02	3.30E+03	1.72E+05	7.96E+05
	Std	**0.00E+00**	5.59E+01	1.36E+01	4.96E+01	4.04E+02	1.49E+05	2.70E+05

Table 4 continued

Function	Stats	SASODE	WOA	GWO	HBO	SSA	SCA	MFO
f24	Mean	**6.13E-02**	1.07E-01	2.27E-01	1.91E+00	8.43E+00	8.97E+00	3.36E+01
	Std	**4.75E-02**	7.39E-02	4.50E-02	2.63E-01	1.06E+00	3.51E+00	3.95E+00
f25	Mean	-3.52E+03	-3.83E+03	-2.36E+03	**-3.87E+03**	-3.24E+03	-1.43E+03	-3.17E+03
	Std	4.67E+02	1.92E+02	1.23E+02	**2.64E+01**	5.57E+01	7.17E+01	1.23E+02
f26	Mean	**0.00E+00**	9.63E-03	1.19E-03	2.03E-03	1.88E-01	2.27E+00	8.95E+00
	Std	**0.00E+00**	3.74E-02	4.61E-03	4.15E-03	4.45E-02	9.49E-01	4.02E+00
f27	Mean	-1.00E-01	**-3.00E-01**	9.00E-41	1.20E-42	2.44E-42	2.72E-28	6.28E-38
	Std	**3.05E-01**	4.66E-01	2.00E-40	8.31E-44	3.68E-42	4.78E-28	1.31E-37
f28	Mean	1.33E-37	**5.53E-42**	8.86E-20	7.11E-36	1.16E-37	1.05E-29	9.57E-41
	Std	3.90E-37	**1.85E-42**	1.69E-19	6.78E-36	2.08E-37	1.44E-29	2.51E-41
f29	Mean	**1.35E-32**	1.36E+00	6.26E+00	1.37E+01	1.87E+02	2.79E+08	2.74E+08
	Std	**5.57E-48**	5.06E-01	5.02E-01	9.49E+00	1.77E+01	1.54E+08	2.34E+08
f30	Mean	**4.71E-33**	1.27E-02	2.42E-01	8.23E-01	1.66E+01	1.53E+08	9.24E+07
	Std	**1.39E-48**	5.86E-03	5.40E-02	5.28E-01	4.08E+00	9.90E+07	1.38E+08
f31	Mean	-4.05E+01	-2.68E+01	-2.55E+01	-3.53E+01	-4.56E+01	-1.79E+01	**-6.01E+01**
	Std	3.79E+00	2.63E+00	6.27E+00	**1.15E+00**	3.45E+00	1.30E+00	3.01E+00
f32	Mean	1.10E-02	1.75E-03	**2.79E-03**	1.63E-01	1.33E+00	6.62E+01	1.38E+02
	Std	3.71E-03	2.21E-03	**9.09E-04**	3.31E-02	2.90E-01	4.65E+01	6.55E+01

parameter optimization problems, industrial engineering and manufacturing optimization problems, and the algorithm is relatively stable. Making full use of the characteristics and advantages of each individual in the optimization process and deciding the surviving individuals based on the size of their jumping rate is a gap area in the existing ODE research.

In the results of 10D and 50D, the comparison algorithms perform best on f6, f8, f9, f12, and f4, f5, f6, f8, f13, respectively. The SSA, SCA, and MFO algorithms do not have an advantage in unimodal functions, except for the algorithm WOA, which is competitive with SASODE. The analysis depicted in Fig. 4 indicates that SASODE exhibits a superior convergence speed and accuracy relative to the other algorithms assessed over a period of 1000 iterations. For f12, although the convergence accuracy is slightly worse than WOA, it has a significant advantage compared with the accuracy of the remaining five algorithms. Based on the characteristics of the unimodal function, we conclude that the SASODE algorithm exhibits remarkable exploitation capabilities, enhances convergence speed, and demonstrates robustness as evidenced by the values of the standard deviation.

4.1.4 The results of multimodal functions

Multimodal functions possess multiple local optima, which distinguishes them from unimodal functions. Moreover, the count of local optima rises exponentially as the dimensionality increases. The resulting multimodal function serves as a reliable metric to judge the exploration capacity of the algorithm. The results of SASODE on the multimodal functions f16-f32 are given in Tables 2, 3, and 4.

Based on the results, it is evident that SASODE exhibits superior performance in multimodal functions as compared to unimodal ones. Out of the 17 functions, the algorithm has secured the first position with regard to both the mean and standard deviation values for f16, f28, f23, f24, f26, and f27. In addition, the SASODE function performs better on more than half of the functions, and in the 10D, the algorithm performs best on f18 and f32 in terms of standard deviation, ranking first on nine functions. In the 50D results, the algorithm has the best standard deviation on f25 and the best on 11 functions. And the 100D results conclude that SASODE performs better on f27, ranking highest on 10 functions. All the above experimental results can demonstrate the advantages of the adaptive parameter mechanism proposed in SASODE. As the iteration proceeds, the jumping rate of each individual in the subpopulation is linearly assigned according to the jumping rate of the surviving individuals in the previous generation and the size of the mean of the Gaussian distribution. This allows historical information to be used directly to guide the search direction, and also allows the algorithm to jump out of the local optimum to search in the global domain as the jumping rate changes. As the complexity of the optimization problem increases, this algorithm based on the adaptive mechanism can be applied to as many different industrial and engineering optimization problems as possible, such as the rocket engine design problem [50], which also shows that

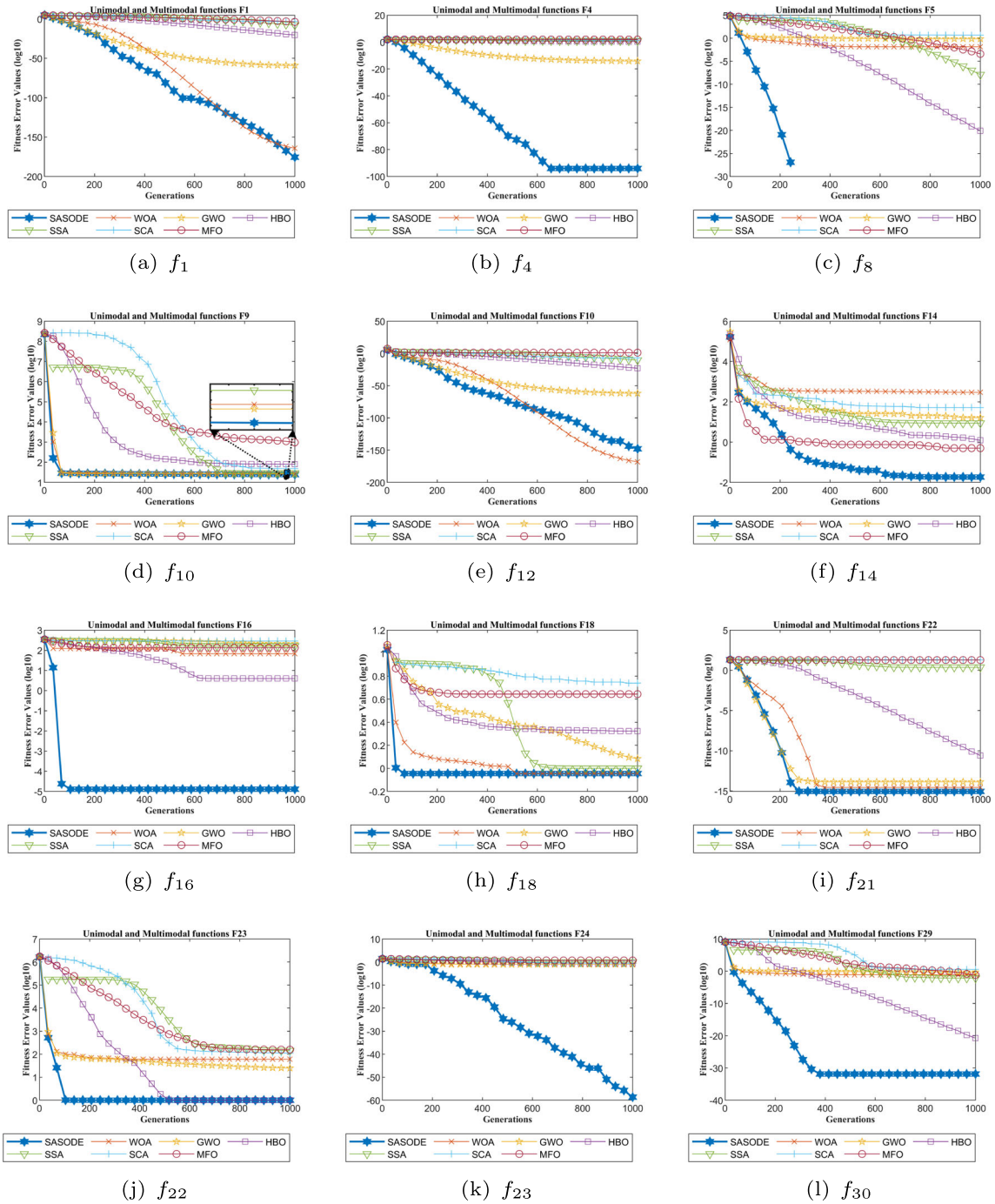


Fig. 4 Convergence graphs of the SASODE and six other optimizers for the selected unimodal functions and multimodal functions at 30D

SASODE has the ideal exploratory ability and advantages in multimodal problems.

4.1.5 Statistical analysis

This section presents the outcomes of the pair-wise Wilcoxon test and multiple-wise Friedman test conducted on SASODE and other algorithms. SASODE is selected as the control algorithm. The significance outcomes of various algorithms on distinct benchmark functions are illustrated in Table 5, which showcases the results of the Wilcoxon test.

1. For unimodal functions f1-f15, SASODE clearly finds better solutions on more than half of the functions, and the algorithm in this paper is significant on all 15 functions compared to SSA and SCA. Compared with HBO and

Table 5 Wilcoxon test for unimodal functions and multimodal functions on 30D(SASODE is the control algorithm)

Function	vs. WOA pvalue (+/=/-)	vs. GWO pvalue (+/=/-)	vs. HBO pvalue (+/=/-)	vs. SSA pvalue (+/=/-)	vs. SCA pvalue (+/=/-)	vs. MFO pvalue (+/=/-)
f1	2.77E-01 (=)	5.57E-10 (-)	3.02E-11 (+)	3.02E-11 (+)	3.02E-11 (+)	3.02E-11 (+)
f2	3.02E-11 (-)	3.02E-11 (-)	3.02E-11 (-)	3.02E-11 (+)	4.08E-11 (+)	7.38E-10 (+)
f3	3.02E-11 (-)	5.57E-10 (-)	3.02E-11 (+)	3.02E-11 (+)	3.02E-11 (+)	3.02E-11 (+)
f4	3.02E-11 (+)	5.57E-10 (-)	3.02E-11 (+)	3.02E-11 (+)	3.02E-11 (+)	3.02E-11 (+)
f5	1.21E-12 (+)	1.21E-12 (+)	1.21E-12 (+)	1.21E-12 (+)	1.21E-12 (+)	1.21E-12 (+)
f6	1.00E+00 (=)	1.19E-12 (+)	1.00E+00 (=)	1.19E-12 (+)	1.13E-12 (+)	1.00E+00 (=)
f7	3.02E-11 (-)	8.48E-09 (-)	5.57E-10 (-)	3.02E-11 (+)	3.02E-11 (+)	3.02E-11 (+)
f8	8.15E-02 (=)	7.76E-11 (-)	3.16E-12 (+)	3.16E-12 (+)	3.16E-12 (+)	3.16E-12 (+)
f9	2.72E-11 (+)	7.38E-11 (+)	4.18E-07 (+)	2.86E-08 (+)	2.72E-11 (+)	6.75E-10 (+)
f10	2.51E-02 (-)	8.48E-09 (-)	3.02E-11 (+)	3.02E-11 (+)	3.02E-11 (+)	3.02E-11 (+)
f11	3.02E-11 (+)	3.02E-11 (+)	1.69E-09 (+)	3.02E-11 (+)	3.02E-11 (+)	3.02E-11 (+)
f12	4.38E-01 (=)	9.88E-03 (+)	3.02E-11 (+)	3.02E-11 (+)	3.02E-11 (+)	3.02E-11 (+)
f13	6.49E-10 (-)	9.40E-12 (+)	2.79E-03 (-)	9.40E-12 (+)	9.40E-12 (+)	2.79E-03 (-)
f14	4.98E-11 (+)	1.41E-09 (+)	2.68E-06 (+)	7.12E-09 (+)	1.61E-10 (+)	4.94E-05 (+)
f15	1.04E-04 (-)	8.48E-09 (-)	3.02E-11 (+)	3.02E-11 (+)	3.02E-11 (+)	3.02E-11 (+)
f16	1.86E-09 (+)	5.22E-12 (+)	2.36E-06 (-)	5.22E-12 (+)	5.22E-12 (+)	7.21E-12 (+)
f17	1.00E+00 (=)	6.47E-04 (+)	1.21E-12 (+)	1.21E-12 (+)	1.21E-12 (+)	1.21E-12 (+)
f18	1.97E-05 (+)	1.72E-12 (+)	1.72E-12 (+)	4.56E-11 (+)	1.72E-12 (+)	1.68E-12 (+)
f19	3.02E-11 (+)	3.02E-11 (+)	3.02E-11 (-)	2.87E-10 (-)	3.02E-11 (+)	4.64E-05 (+)
f20	7.77E-09 (+)	1.95E-03 (-)	4.80E-07 (-)	3.02E-11 (+)	1.36E-07 (+)	1.17E-03 (+)
f21	7.60E-07 (+)	3.02E-11 (-)	3.02E-11 (+)	3.02E-11 (+)	3.02E-11 (+)	3.02E-11 (+)
f22	1.65E-02 (-)	1.69E-10 (-)	2.84E-10 (+)	1.40E-11 (+)	1.40E-11 (+)	1.40E-11 (+)
f23	1.21E-12 (+)	1.21E-12 (+)	2.20E-06 (+)	1.21E-12 (+)	1.21E-12 (+)	1.21E-12 (+)
f24	4.23E-08 (+)	5.57E-10 (+)	3.02E-11 (+)	3.00E-11 (+)	3.02E-11 (+)	3.01E-11 (+)
f25	4.20E-01 (=)	3.00E-11 (+)	1.52E-06 (-)	3.00E-11 (+)	3.00E-11 (+)	3.00E-11 (+)
f26	8.15E-02 (=)	8.15E-02 (=)	2.15E-02 (+)	1.21E-12 (+)	1.21E-12 (+)	1.21E-12 (+)
f27	3.25E-07 (+)	2.23E-10 (+)	2.23E-10 (+)	2.23E-10 (+)	1.06E-11 (+)	1.01E-10 (+)
f28	9.36E-10 (+)	7.21E-12 (+)	6.48E-12 (+)	1.23E-11 (+)	6.48E-12 (+)	7.21E-12 (+)
f29	1.72E-12 (+)	1.72E-12 (+)	4.56E-11 (-)	1.72E-12 (+)	1.72E-12 (+)	1.72E-12 (+)
f30	1.21E-12 (+)	1.21E-12 (+)	1.21E-12 (+)	1.21E-12 (+)	1.21E-12 (+)	1.21E-12 (+)
f31	4.50E-11 (+)	1.44E-03 (+)	3.82E-09 (-)	3.18E-01 (=)	3.02E-11 (+)	1.02E-05 (-)
f32	1.86E-09 (-)	4.50E-11 (-)	1.73E-07 (+)	3.02E-11 (+)	1.29E-06 (+)	3.02E-11 (+)
total (+/=/-)	17/7/8	19/1/12	22/1/9	30/1/1	32/0/0	29/1/2

MFO, SASODE has a greater advantage on f11 and f13, respectively. Compared with WOA and GWO, f5 and f7 are dominant, respectively. In other words, SASODE performs much better on single-peaked functions than all other algorithms.

2. For multimodal functions f16-f32, SASODE finds better solutions compared to all six comparison algorithms. For example, compared to WOA, GWO, SSA, SCA and MFO, the results are significant on f12, f15, f17 and f16, respectively. Although, SASODE does not perform significantly on six functions compared to HBO, the algorithm employed in this paper continues to exhibit exceptional performance when compared to other algorithms. Hence, SASODE surpasses other algorithms on multimodal functions.

The comparison of the convergence curves of SASODE with benchmark functions on 30D has been conducted in conjunction with WOA, GWO, HBO, SSA, SCA, and MFO, as illustrated in Fig. 4. The horizontal axis of Fig. 4 represents the number of iterations, while the vertical axis represents the fitness. Each algorithm was independently run 30 times with a fixed number of iterations of 1000. In Fig. 4, (a)-

Table 6 Wilcoxon test for benchmark functions at different dimensions (SASODE is the control algorithm)

SASODE vs Algorithms	D=10			D=30			D=50			D=100		
	+	=	-	+	=	-	+	=	-	+	=	-
WOA	20	6	6	17	7	8	14	7	11	13	9	10
GWO	20	2	10	19	1	12	26	1	5	26	1	5
HBO	12	10	10	22	1	9	23	1	8	26	4	2
SSA	30	2	0	30	1	1	28	2	2	28	2	2
SCA	29	0	3	32	0	0	31	0	1	31	1	0
MFO	28	2	2	29	1	2	28	1	3	29	1	2

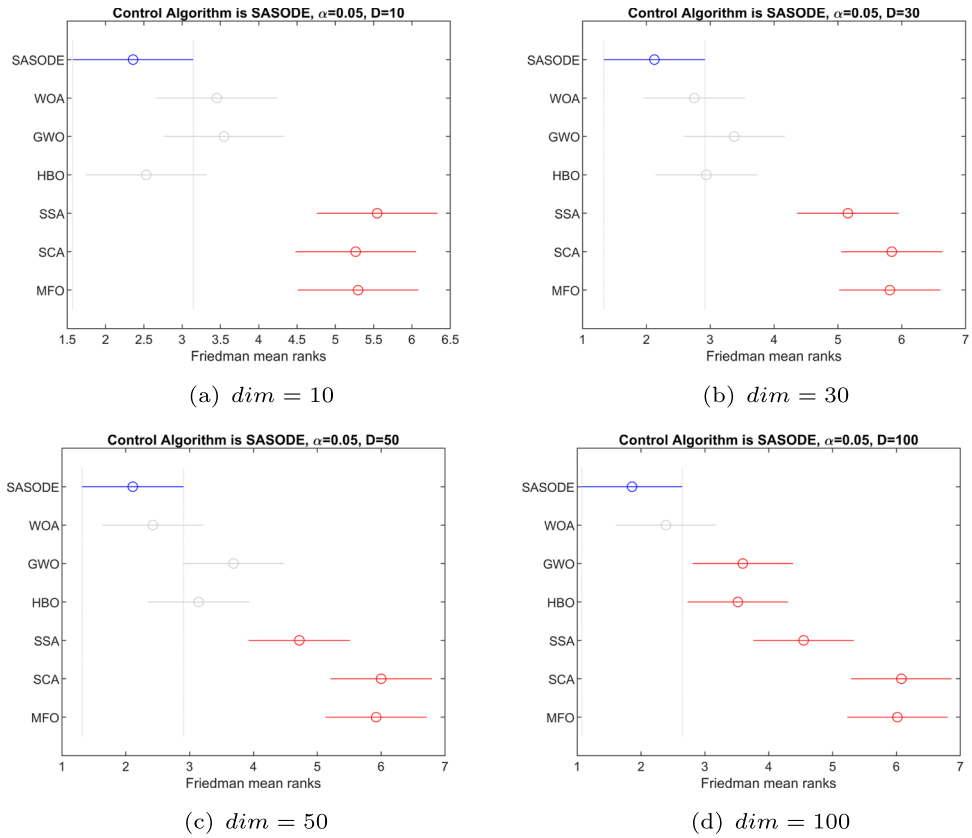
(f) denotes the convergence curves of unimodal functions, and (g)-(l) denotes the convergence curves of multimodal functions. Based on the information presented in Fig. 4, it can be inferred that the following conclusions can be drawn:

1. Based on the iteration curves of the unimodal functions, it can be observed that the convergence accuracy of f1, f4, f8, and f14 is the best among the six algorithms, but on the f12, the accuracy of the SASODE algorithm is slightly worse than that of WOA. On f10, despite having equivalent convergence speed to GWO and WOA, the SASODE outperforms both in terms of convergence precision.
2. Based on the iteration curves of the multimodal functions, the algorithm is optimal on f16, f18, f21, f22 and f30, both in terms of convergence speed and convergence accuracy. For f23, SASODE did not converge during the first 1000 iterations, but the convergence accuracy was significantly better than the remaining five comparison algorithms.
3. To conclude, SASODE exhibits commendable performance on unimodal and multimodal functions for a dimensionality of 30. The outcomes of multimodal functions are superior to those of unimodal, as demonstrated by the numerical outcomes pertaining to residual dimensions.

4.1.6 Scalability analysis

To examine the impact of dimensionality on the performance of the algorithm, this section conducts experiments on dimensions of 10, 30, 50, and 100 respectively, and subsequently compares the algorithms applied to all test functions. The Wilcoxon test and Friedman test results are included in Table 6 and Fig. 5 for analysis. Based on these findings, it can be concluded that:

1. From the findings presented in Table 6, it is evident that SASODE ranks first in essentially all dimensions, but the different dimensions can be divided in detail.



(a) $dim = 10$ (b) $dim = 30$

(c) $dim = 50$ (d) $dim = 100$

Fig. 5 Friedman mean ranks on unimodal functions and multimodal functions

Table 7 Results of CEC2017 from SASODE and seven other metaheuristic algorithms on 10D

Function	Stats	SASODE	WOA	GWO	HBO	SSA	SCA	MFO	HHO
f1	Mean	**0.00E+00**	1.29E+06	2.76E+07	4.66E+02	3.52E+03	7.86E+08	4.25E+07	4.20E+05
	Std	**0.00E+00**	1.76E+06	9.03E+07	6.92E+02	3.80E+03	3.27E+08	2.07E+08	4.20E+05
f2	Mean	**0.00E+00**	0.00E+00	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	0.00E+00	0.00E+00
f3	Mean	**0.00E+00**	1.33E+03	9.49E+02	1.41E+01	1.97E-09	1.51E+03	5.25E+03	2.05E+00
	Std	**0.00E+00**	1.49E+03	1.42E+03	6.36E+01	8.11E-10	1.10E+03	8.45E+03	2.05E+00
f4	Mean	**3.04E+00**	3.33E+01	8.39E+00	4.94E+00	9.20E+00	4.69E+01	1.76E+01	1.91E+01
	Std	**7.05E-01**	4.58E+01	4.40E+00	6.75E-01	1.58E+01	2.87E+01	2.27E+01	1.91E+01
f5	Mean	1.50E+01	5.43E+01	1.38E+01	**1.12E+01**	1.91E+01	4.70E+01	2.87E+01	4.76E+01
	Std	8.23E+00	1.95E+01	7.83E+00	**3.58E+00**	8.62E+00	6.43E+00	1.02E+01	4.76E+01
f6	Mean	**3.03E-14**	3.36E+01	8.56E-01	0.00E+00	8.56E+00	1.69E+01	1.78E+00	3.22E+01
	Std	**5.11E-14**	1.21E+01	1.09E+00	0.00E+00	6.52E+00	3.84E+00	3.48E+00	3.22E+01
f7	Mean	3.03E+01	8.14E+01	3.09E+01	**2.58E+01**	3.49E+01	7.23E+01	3.62E+01	8.56E+01
	Std	6.32E+00	1.98E+01	1.41E+01	**4.28E+00**	1.22E+01	1.03E+01	1.29E+01	8.56E+01
f8	Mean	1.72E+01	4.02E+01	1.37E+01	**1.20E+01**	2.15E+01	4.10E+01	2.69E+01	3.09E+01
	Std	1.02E+01	1.51E+01	6.06E+00	**4.64E+00**	6.41E+00	6.14E+00	1.02E+01	3.09E+01
f9	Mean	**0.00E+00**	4.11E+02	1.02E+01	0.00E+00	3.23E+01	1.06E+02	7.51E+01	3.90E+02
	Std	**0.00E+00**	2.30E+02	2.27E+01	0.00E+00	8.17E+01	5.62E+01	1.42E+02	3.90E+02
f10	Mean	**3.95E+02**	9.78E+02	6.35E+02	6.55E+02	9.27E+02	1.29E+03	9.56E+02	1.03E+03
	Std	2.88E+02	3.35E+02	2.69E+02	1.98E+02	2.51E+02	**1.86E+02**	3.13E+02	1.03E+03
f11	Mean	**7.56E-01**	1.08E+02	4.00E+01	2.84E+00	7.64E+01	1.00E+02	8.18E+01	7.69E+01
	Std	**1.00E+00**	7.31E+01	4.20E+01	1.23E+00	8.12E+01	4.42E+01	1.05E+02	7.69E+01
f12	Mean	**3.76E+01**	3.59E+06	5.37E+05	1.17E+05	2.14E+06	1.22E+07	1.39E+06	2.91E+06
	Std	**5.80E+01**	4.16E+06	7.69E+05	1.44E+05	2.20E+06	6.91E+06	2.83E+06	2.91E+06
f13	Mean	**4.52E+00**	1.81E+04	8.56E+03	1.36E+03	1.33E+04	2.65E+04	1.11E+04	1.35E+04
	Std	**2.09E+00**	1.27E+04	5.69E+03	1.70E+03	1.08E+04	2.14E+04	1.15E+04	1.35E+04
f14	Mean	**2.98E-01**	9.66E+02	1.32E+03	4.39E+01	1.21E+02	2.14E+02	1.65E+03	1.71E+02
	Std	**5.32E-01**	1.27E+03	1.77E+03	4.06E+01	5.68E+01	1.03E+02	2.15E+03	1.71E+02
f15	Mean	**2.74E-01**	3.49E+03	3.34E+03	8.96E+01	1.11E+03	6.87E+02	8.34E+03	1.84E+03
	Std	**3.32E-01**	3.28E+03	4.34E+03	1.54E+02	9.86E+02	5.97E+02	9.35E+03	1.84E+03
f16	Mean	**7.95E-01**	2.45E+02	1.14E+02	1.51E+00	1.12E+02	1.37E+02	1.12E+02	2.24E+02
	Std	**2.06E+00**	1.52E+02	8.93E+01	6.37E-01	1.04E+02	6.53E+01	9.40E+01	2.24E+02
f17	Mean	**2.11E+00**	9.62E+01	6.26E+01	3.83E+00	6.97E+01	7.30E+01	6.96E+01	8.28E+01
	Std	4.47E+00	5.87E+01	2.87E+01	**3.67E+00**	3.47E+01	1.03E+01	5.00E+01	8.28E+01
f18	Mean	**1.53E+00**	1.46E+04	2.43E+04	2.59E+03	1.67E+04	1.36E+05	2.12E+04	1.39E+04
	Std	**5.05E+00**	1.34E+04	1.64E+04	2.29E+03	9.10E+03	9.15E+04	1.41E+04	1.39E+04
f19	Mean	**6.24E-02**	1.94E+04	1.36E+04	4.35E+01	1.08E+03	1.83E+03	1.67E+04	7.53E+03
	Std	**2.76E-01**	2.25E+04	4.70E+04	6.93E+01	2.07E+03	3.21E+03	1.28E+04	7.53E+03
f20	Mean	8.12E-01	1.70E+02	7.16E+01	**1.04E-02**	6.41E+01	8.76E+01	7.95E+01	1.70E+02
	Std	3.03E+00	6.58E+01	5.34E+01	**5.70E-02**	3.48E+01	2.98E+01	6.10E+01	1.70E+02
f21	Mean	**1.43E+02**	2.00E+02	1.99E+02	1.52E+02	1.74E+02	1.69E+02	2.18E+02	1.97E+02
	Std	5.78E+01	6.37E+01	**3.93E+01**	5.24E+01	6.05E+01	6.85E+01	3.97E+01	1.97E+02
f22	Mean	**9.10E+01**	1.18E+02	1.03E+02	9.73E+01	1.03E+02	1.61E+02	9.78E+01	1.14E+02
	Std	2.83E+01	1.89E+01	1.69E+01	9.76E+00	**2.08E+00**	2.18E+01	2.89E+01	1.14E+02
f23	Mean	**3.07E+02**	3.50E+02	3.17E+02	3.15E+02	3.21E+02	3.52E+02	3.29E+02	3.69E+02
	Std	**2.27E+00**	2.05E+01	9.29E+00	4.41E+00	8.54E+00	7.36E+00	9.06E+00	3.69E+02

Table 7 continued

Function	Stats	SASODE	WOA	GWO	HBO	SSA	SCA	MFO	HHO
f24	Mean	3.28E+02	3.81E+02	3.48E+02	**2.59E+02**	3.43E+02	3.78E+02	3.60E+02	4.30E+02
	Std	4.33E+01	1.84E+01	1.29E+01	8.96E+01	4.68E+01	3.89E+01	**1.12E+01**	4.30E+02
f25	Mean	4.15E+02	4.27E+02	4.36E+02	**4.08E+02**	4.21E+02	4.59E+02	4.33E+02	4.20E+02
	Std	2.25E+01	8.88E+01	1.76E+01	1.74E+01	2.41E+01	**1.51E+01**	2.43E+01	4.20E+02
f26	Mean	3.02E+02	8.74E+02	5.98E+02	**2.74E+02**	3.48E+02	4.71E+02	4.20E+02	9.45E+02
	Std	**8.61E+00**	4.89E+02	4.52E+02	8.42E+01	2.05E+02	3.17E+01	1.66E+02	9.45E+02
f27	Mean	**3.90E+02**	4.26E+02	3.99E+02	3.91E+02	3.94E+02	4.03E+02	3.94E+02	4.45E+02
	Std	**8.57E-01**	2.38E+01	1.25E+01	1.54E+00	1.35E+01	1.74E+00	2.17E+00	4.45E+02
f28	Mean	3.87E+02	5.79E+02	5.71E+02	**3.80E+02**	4.40E+02	4.95E+02	4.92E+02	5.44E+02
	Std	1.16E+02	1.56E+02	7.26E+01	**4.22E+01**	1.48E+02	7.33E+01	9.89E+01	5.44E+02
f29	Mean	**2.31E+02**	4.53E+02	2.86E+02	2.83E+02	3.05E+02	3.27E+02	3.23E+02	4.29E+02
	Std	**3.94E+00**	1.05E+02	4.85E+01	1.64E+01	5.79E+01	3.64E+01	5.09E+01	4.29E+02
f30	Mean	1.51E+05	1.16E+06	5.08E+05	**2.60E+04**	1.92E+05	7.84E+05	5.93E+05	1.04E+06
	Std	3.50E+05	1.45E+06	6.15E+05	**2.29E+04**	3.36E+05	4.54E+05	5.05E+05	1.04E+06

2. Analyzing the whole experimental results vertically, in the case of 10D, WOA and HBO are the second echelon except the SASODE algorithm, SCA and MFO are ranked after the above two algorithms, and SSA is the worst performing algorithm. In the 30D, 50D and 100D cases, WOA is one of the best competitive algorithms for SASODE, except for GWO and HBO, followed by SSA and MFO. SCA is the worst algorithm in all three cases.
3. The results of the horizontal comparison show that the performance of the algorithm gradually improves as the dimensionality increases and the difficulty in finding the optimal function increases. From this, we can infer that SASODE ranks first among the six algorithms in terms of the overall level of the optimization function in high dimensions.
4. The results of the Friedman rank on various dimensions demonstrate that SASODE has a slight edge over HBO, and outperforms SSA, SCA, and MFO in 10D. Moreover, the same situation can be observed in 30D and 50D. The biggest difference is that other algorithms are slightly inferior to SASODE. However, SASODE exhibits more statistical significance than other algorithms in 100D, except for the WOA. It indicates that the performance gains of SASODE are heightened as the dimensionality increases.

4.2 Experiments on CEC 2017

To understand the performance of SASODE on different benchmark test suites. The recently proposed CEC 2017 test suite includes unimodal, multimodal, hybrid, and composite functions that are tested in this section [51]. The present SASODE algorithm is evaluated against various leading algorithms in the field including WOA, GWO, HBO, SSA, SCA, MFO, and HHO. These algorithms are run independently with a maximum iteration count of 1000, and the process is repeated 30 times. In order to ensure equitable comparison, fitness evaluations for each algorithm are capped at 50000, and population size is fixed at 50.

The outcomes pertaining to our study are exhibited in Table 7. As delineated in the table, SASODE manifests superior performance for unimodal functions (f1, f3). Furthermore, with respect to most multimodal functions (f4-f10), SASODE performs better compared to other algorithms, except for f5, f7 and f8. It should be noted that the good exploration behavior of SASODE on multimodal functions is due to the Lehmer mean and variable subpopulation size strategies. For the mixed functions (f11-f20), SASODE ranks first among all tested functions. The aforementioned outcomes suggest that SASODE attains equilibrium between EE. In terms of the amalgamated functions (f21-f30), SASODE demonstrated superior outcomes on more than fifty percent of the functions as compared to its counterparts. To sum up, SASODE demonstrated superior performance relative to its rivals in CEC 2017.

4.3 The application in engineering optimization problems

The effectiveness of the proposed algorithm is verified by three real-world optimization problems such as Parameter estimation for frequency-modulated sound waves (FM),

Table 8 Results of Real world optimization problems from SASODE and GWO, HBO, HHO and LFD

Function	Stats	SASODE	EO	GWO	HBO	HHO	LFD
FM	Best	**0**	0	0.053201	2.76E-25	0.049553	8.865514
	Worst	23.97243	20.16705	23.04639	12.56826	21.98638	23.64362
	Mean	6.721396	10.05338	14.87228	**3.538301**	15.20837	16.89727
	Std	8.876354	6.521431	5.570155	5.060894	6.126967	**4.620611**
	Median	**0.006419**	11.79421	13.40932	0.172716	16.33427	17.57606
LJ	Best	-27.5223	**-28.4225**	-27.3672	-15.6585	-25.4166	-12.5645
	Worst	-15.2313	-22.0815	-17.9967	-9.29099	-9.92174	-9.39773
	Mean	-23.8258	**-26.0689**	-23.0756	-11.4448	-17.8416	-10.1641
	Std	3.453044	1.89422	2.49569	1.331784	4.006767	**0.806277**
	Median	-25.0552	**-26.4905**	-23.1168	-11.3836	-18.3289	-9.90675
SPRP	Best	**0.5**	0.5	0.5	0.5	0.5	0.5
	Worst	0.5	0.5	0.5	0.5	0.5	0.505272
	Mean	**0.5**	0.5	0.5	0.5	0.5	0.500264
	Std	**0**	0	0	0	0	0.001179
	Median	**0.5**	0.5	0.5	0.5	0.5	0.5

Lennard-Jones Potential Problem (LJ) and Spread Spectrum Radar Polly Phase Code Design (SPRP) [52]. The detailed definitions and mathematical models of these problems can be found in Appendix A. Moreover, several state-of-the-art metaheuristics are utilized as the comparison algorithms, such as EO, MFO, RUN [53], GWO, HBO, HHO, LFD [54], SCA, SSA and WOA. For the specific parameter settings of each algorithm, see the corresponding original work. The comparison algorithms have undergone fitness evaluations totaling 150000, and each algorithm's population size is set at 50. This results in 3000 iterations for these algorithms. It is worth noting that the effectiveness of the mutation operator in DE is highly sensitive to the population size. Therefore, SASODE utilizes a population size of 150. To maintain fairness, the iteration about SASODE is limited to 1000 despite undergoing the same number of fitness evaluations as the comparison algorithm. To ensure optimal readability of the table, the results will be presented across two tables, namely Tables 8 and 9.

Tables 8 and 9 show the experimental results of FM. The results depict that SASODE surpasses over fifty percent of the evaluated algorithms. The statistical values of median and mean illustrate that SASODE accomplishes the optimal value more frequently compared to the other algorithms. The reason is that the adaptive parameter strategy can better adapt to

Table 9 Results of Real world optimization problems from SASODE and MFO, RUN, SCA, SSA and WOA

Function	Stats	SASODE	MFO	RUN	SCA	SSA	WOA
FM	Best	**0**	0	6.2E-06	10.39823	2.93E-10	11.49727
	Worst	24.94037	25.09711	25.16314	22.86694	25.62236	23.64362
	Mean	**6.721396**	18.58255	15.90388	15.93672	16.06055	18.67931
	Std	8.876354	6.735963	6.023931	**4.105501**	5.941114	4.238271
	Median	**0.006419**	19.6166	19.35855	15.61022	17.51297	19.83866
LJ	Best	-27.5223	-20.554	**-28.4225**	-11.0587	-26.7605	-23.7917
	Worst	-5.06569	-25.5037	-3.22927	-10.1503	-8.08966	-9.39773
	Mean	-23.8258	-11.278	**-26.9948**	-8.32382	-16.96	-18.3792
	Std	3.453044	3.868638	**0.843083**	2.482408	4.698715	4.850882
	Median	-25.0552	-10.7748	**-27.3184**	-9.24752	-16.1619	-20.0703
SPRP	Best	**0.5**	0.5	0.5	0.5	0.5	0.5
	Worst	0.5	0.542243	0.588805	1	0.5	0.663841
	Mean	**0.5**	0.503338	0.50444	0.576355	0.5	0.536747
	Std	**0**	0.01067	0.019857	0.127375	0	0.049517
	Median	**0.5**	0.5	0.5	0.511374	0.5	0.500428

Table 10 Datasets of the feature selection

No.	Datasets	Number feature	Number sample
1	Ionosphere	34	351
2	Sonar	60	200
3	Credit6000	65	6000
4	Dnatest	180	1186
5	Permission	88	4407
6	Spambase	57	4601
7	SPECTheart	22	267
8	Waveform	40	3345
9	WDBC	30	569

the evolutionary process of this complex problem and make the algorithm reach the optimum more smoothly.

As shown in Tables 8 and 9, SASODE and some other algorithms, such as EO, SSA, HBO, HHO and GWO, rank first in the design of the SPRP. The results indicate that the problem is relatively easy to solve. There are many algorithms that can be optimal, but some are still not stable enough (such as LFD, MFO, RUN, SCA, and WOA). In general, SASODE is able to maintain good stability on this problem.

5 Feature selection optimization

To further corroborate the efficacy of SASODE in addressing practical issues, we have applied the SASODE towards feature selection problem, which has been proven to belong to an expensive optimization problem class. As a result, we will begin with introducing the datasets employed by the algorithm, followed by a parametric analysis of the comparison algorithm, and conclude by analyzing experimental results.

5.1 Datasets

The datasets utilized in this stage of the experiment are displayed in Table 10 and encompass critical information like dataset name, number of attributes, and samples. It is worth mentioning that this paper selects eight real datasets sourced from the UCI Machine Learning Laboratory [55] in addition to the Permission dataset for Android malicious application classification extracted from the literature [56]. From Table 10 we can see that the number of attributes in the datasets are all below five hundred, but the number of samples varies from several hundred to several thousand, and the attributes are not proportional to the number of samples, and these characteristics make the datasets a challenging task in classification.

5.2 Experimental settings

Due to the unbalanced number of samples in the above datasets,the experimental process employs K-fold cross-validation methodology. Specifically, the value of k is established as 10, and dividing the dataset into ten copies of the same size, nine of which are used for training data in the feature selection process and one for testing data. The experimental classifier is KNN classifier.

5.3 Comparison of algorithm parameter settings and evaluation criteria

The comparison algorithms used in this section include the differential evolution algorithm with opposition-based learning ODE, and variants of ODE, CODE [57], COODE [58], and QRODE [59], as well as the emerging algorithms HHO, JAYA [60], and WOA algorithms proposed in recent years for solving feature selection. To ensure fairer experimental results, we used the control variables method, and all algorithms were iterated 100 times and run 20 times independently. Table 11 shows the parameter settings of the algorithm.

Feature selection is a class of discrete optimization problems, the optimization goal is to improve the accuracy of the classifier to reduce the number of features selected [61]. Give a dataset with N features, each individual in the population is a binary vector$X = (x_1, x_2, ...x_N)$, each bit $x_d \in \{0, 1\}$, $d = 1, 2, ..., N$, where 1 denotes the dth feature is selected,and 0 denotes it is not selected. In this paper, two different evaluation functions are used. One is the classifier accuracy rate and the other is the classifier error rate. Equations (9) and (10) are the calculation formula.

$$ACC = \frac{TP + TN}{TP + TN + FP + FN} \tag{9}$$

Table 11 Parameter settings

Algorithm	SASODE	CODE	COODE	HHO	ODE	QRODE	WOA
Parameter values	$\mu_J = 0.3, c = 0.01$	$Jr = 0.3$	$Jr = 0.3$	$\beta = 1.5$	$Jr = 0.3$	$Jr = 0.05$	$b = 1$

* Jr is the jumping rate of the ODE. μ_J is the initial mean of the Gaussian distribution. c is a positive number, the experiment verifies that the effect is optimal when $c = 0.01$
** β is a parameter in the Levy formula in HHO
*** b is a parameter in the update formula in WOA

Table 12 Comparison of mean and standard deviation of different algorithms on different datasets

Dataset	SASODE	CODE	COODE	HHO
	Mean± Std	Mean±Std	Mean± Std	Mean± Std
Ionosphere	**0.960**± 0.022(1)	0.958±0.025(2)	0.951±0.019(4)	0.935±0.028(8)
Sonar	0.974 ±0.024(3)	0.967 ±0.025(6)	0.974±0.024(2)	0.920±0.044(8)
Credit6000	**0.864** ±0.007(1)	0.862 ±0.005(5)	0.859±0.007(6)	0.848±0.006(8)
Dnatest	0.895±**0.010**(3)	**0.907** ±0.016(1)	0.894±0.022(5)	0.848±0.032(7)
Permission	**0.969**± **0.005**(2)	0.966 ±0.006(6)	0.969±0.006(1)	0.960±0.009(8)
Spambase	**0.940** ±**0.006**(3)	0.939 ±0.006(5)	0.934±0.008(6)	0.916±0.013(8)
SPECTheart	0.816± **0.008**(4)	0.813 ±0.015(7)	0.819±0.025(2.5)	0.809±0.024(8)
Waveform	0.933 ±**0.007**(2)	**0.935** ±0.007(1)	0.925±0.008(6)	0.913±0.009(8)
WDBC	**0.968**± **0.013**(1)	0.961 ±0.016(3)	0.954±0.016(8)	0.957±0.019(6)
Mean Rank	2	4	5	8
Dataset	JAYA	ODE	QRODE	WOA
	Mean± Std	Mean±Std	Mean± Std	Mean± Std
Ionosphere	0.951±0.024(5)	0.950±0.017(6)	0.952±0.022(3)	0.940±**0.016**(7)
Sonar	0.973±0.028(4)	**0.978**±**0.019**(1)	0.972±0.020(5)	0.930±0.024(7)
Credit6000	0.863±0.006(3)	0.864±0.006(2)	0.862±0.007(4)	0.851±**0.004**(7)
Dnatest	0.893±0.017(6)	0.895±0.016(4)	0.898±0.018(2)	0.842±0.025(8)
Permission	0.968±0.005(4)	0.968±0.005(3)	0.967±0.005(5)	0.963±0.007(7)
Spambase	0.940±0.008(4)	0.940±0.008(1)	0.940±0.006(2)	0.920±0.012(7)
SPECTheart	0.813±0.017(5.5)	**0.824**±0.020(1)	0.819±0.015(2.5)	0.813±0.017(5.5)
Waveform	0.930±0.006(5)	0.932±0.006(4)	0.932± 0.008(3)	0.916±0.008(7)
WDBC	0.960±0.017(4)	0.962±0.013(2)	0.956±0.015(7)	0.958±0.016(5)
Mean Rank	5	3	4	7

$$error\left(KNN_{classifier}\right) = 1 - ACC = \frac{Wrong_{num}}{Correct_{num} + Wrong_{num}} \tag{10}$$

5.4 Comparison of experimental results

The data displayed in Table 12 represents the mean standard deviation computations using (9) as the evaluation metric. The average ranking is presented in the last row of the table, while the best experimental results are highlighted in bold in the table. In the analysis of the experimental results, this paper divides the competing algorithms into two categories. The first part is the analysis and comparison between SASODE and different variants of differential evolution algorithms, and the second part is the comparison between SASODE and other algorithms.

Table 13 Wilcoxon rank sum test results

Dataset	CODE	COODE	HHO	JAYA	ODE	QRODE	WOA
Ionosphere	=	=	+	=	+	=	+
Sonar	=	=	+	=	=	=	+
Credit6000	=	+	+	=	=	=	+
Dnatest	-	=	+	=	=	=	+
Permission	=	=	+	=	=	=	+
Spambase	=	+	+	=	=	=	+
SPECTheart	=	=	+	=	=	=	=
Waveform	=	+	+	=	=	=	+
WDBC	=	+	+	=	=	+	+
+/=/-	0/8/1	4/5/0	9/0/0	0/9/0	1/8/0	1/8/0	8/1/0

5.4.1 Comparison with ODE and its variant

By analyzing the data in Table 12, we can understand that SASODE has the highest accuracy on Ionosphere, Credit6000, Permission, and Spambase, which is because the algorithm can effectively avoid the best individuals in the population from performing OBL. The traditional OBL ordinary differential equation algorithm is OBL with less than jumping probability for all individuals in the population, which is tantamount to destabilizing the population, making the local best individuals in the population eliminated with a certain probability of being the individuals with poor fitness values. We can see that SASODE has a lower standard deviation than the other improved DE algorithms on the remaining eight datasets, except for ODE. According to the iteration step of the algorithm, it is easy to find that SASODE selects individuals to join the subpopulation by comparing each individual with a Gaussian random number and that the algorithm is significantly more random compared to the traditional ODE, so the algorithm is more stable during the iteration. The range of the difference was found to be 0.001, and since the algorithms are all randomized, the small differences in stability between the algorithms are negligible while ensuring accuracy. In addition to the comparison of mean variances between algorithms, the ranking analysis shows that among all algorithms, SASODE ranks first, followed by ODE, CODE, QRODE tied for third, and COODE ranked fourth.

5.4.2 Comparison with other algorithms

In addition to the comparison with ODE variants, SASODE has more obvious advantages in accuracy compared with HHO, JAYA and WOA algorithms, especially WOA algorithm. The results of the Wilcoxon rank sum test are shown in Table 13, from the table it can be inferred that SASODE

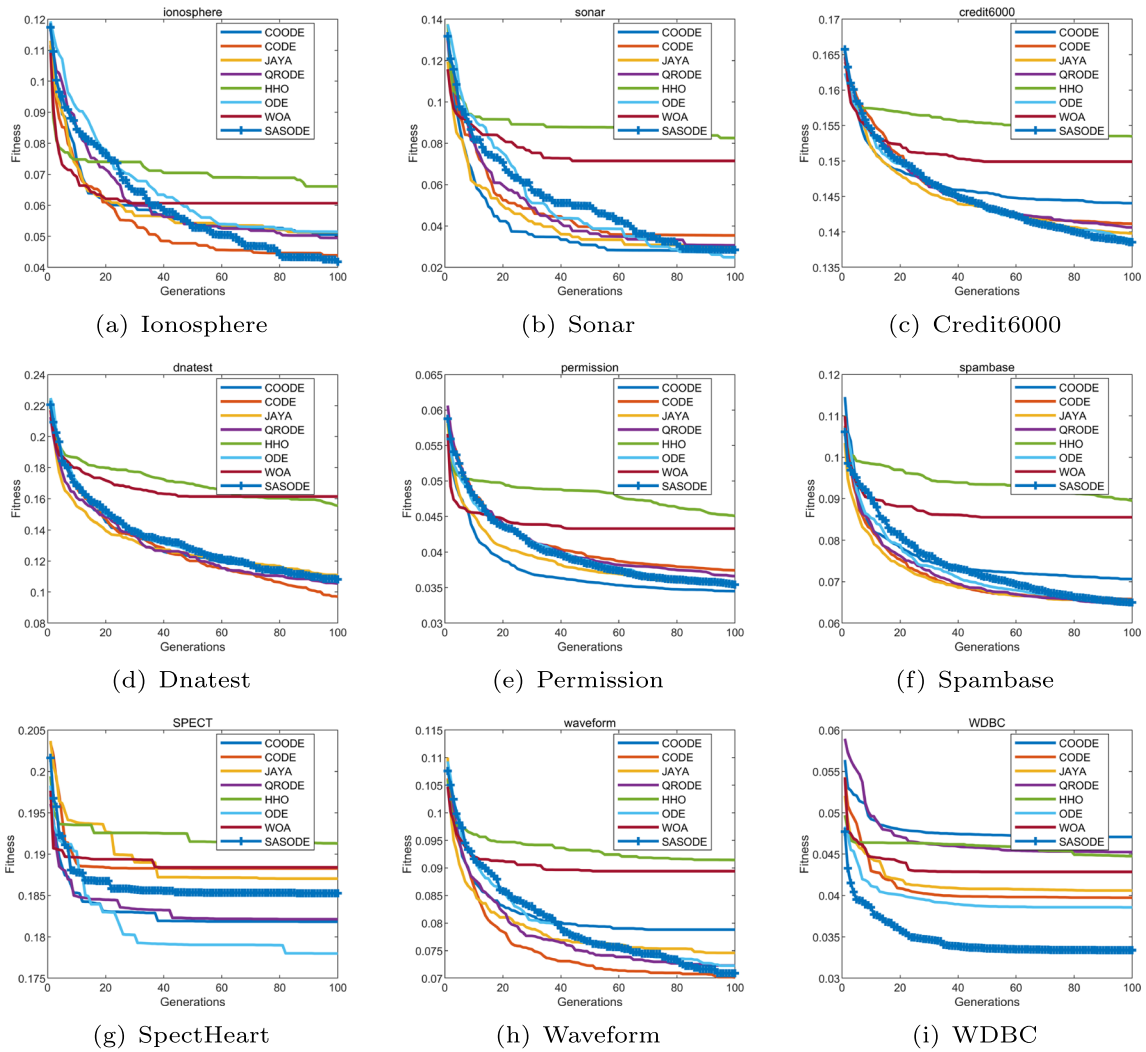


Fig. 6 Convergence behavior of the SASODE and seven other algorithms for the nine UCI datasets

displays superior performance in comparison to WOA across all nine data sets. Even though the WOA algorithm has a superior standard deviation on Ionosphere and Credit6000 compared to SASODE, its accuracy has been underwhelming when compared to the algorithm introduced in this paper. Secondly, according to the results of the rank sum test, it can be inferred that JAYA and SASODE algorithms are similar. However, when considering both the mean and standard deviation, JAYA algorithm was found to be notably inferior to SASODE algorithm. This indicates that SASODE outperforms HHO, JAYA, and WOA algorithms.

5.4.3 Convergence behavior

In addition to comparing the results of different algorithms on different datasets according to (10), we visualize the algorithm iteration 100 with the evaluation function and the visualization results as in Fig. 6. It is easy to see by the iterative curves that the blue curve performs well on most of the datasets. SASODE converges the least on the Ionosphere, Sonar, Credit6000, Spambase, Waveform, and WDBC datasets, which means that the algorithm has the highest accuracy and the lowest error rate. However, a closer look also shows that SASODE can converge to better values, but not as fast as the other algorithms on the Sonar, Permission and Waveform datasets. This is due to the larger number of features in the Sonar dataset, the larger number of samples in the Permission and Waveform datasets, and SASODE focuses on the jumping rate of each individual in the population, which prevents it from simultaneously considering both the effectiveness and efficiency of the optimization in the early stage. Therefore, the performance of SASODE may not be very good in dealing with high-dimensional large-sample problems. But in the later stage, the update of the jumping rate is based on historical information and there is no need to find the optimal search direction, so as the number of iterations increases (after 60 generations), the algorithm gradually finds the optimal solution and converges.

5.4.4 Number of features and running time analysis

Table 14 displays the mean quantity of features that were chosen by various algorithms across a range of datasets, as well as the mean time taken by each algorithm to make these selections. It is evident from Table 14 that SASODE does not exhibit significant benefits with regard to the quantity of selected features and the execution time. This is because the algorithm assigns a jumping rate to each individual in the population during the iteration process, which theoretically increases the time cost and leads to a slowdown in the pre-optimization process, but this operation helps the algorithm to jump out of the local optimum to increase the probability

Table 14 Comparison results of different algorithms in terms of the number of selected features and time

Dataset	SASODE		CODE		COODE		HHO	
	Size	Time	Size	Time	Size	Time	Size	Time
Ionosphere	7.55	40.20	7.35	19.12	8.4	18.79	5.95	27.90
Sonar	19	37.61	17.05	18.37	16.6	17.70	17.3	17.36
Credit6000	28.35	409.93	27.9	170.01	26.6	161.31	19.7	121.67
Dnatest	82.7	88.00	81.8	38.99	86	39.15	83.5	39.72
Permission	37.3	350.97	32.85	140.48	36.55	152.08	52.4	183.78
Spambase	30.9	264.98	30.35	115.31	30.65	113.64	35.95	127.29
SPECTheart	6.95	38.18	7.35	18.50	5.55	18.55	5.8	16.57
Waveform	**19.65**	114.73	21.3	49.31	20.15	51.28	23.1	54.30
WDBC	4.3	41.22	3.55	19.40	4.55	20.18	5.5	18.56
Dataset	JAYA		ODE		QRODE		WOA	
	Size	Time	Size	Time	Size	Time	Size	Time
Ionosphere	8.8	12.34	6.8	18.82	7.2	16.74	**4.35**	**10.85**
Sonar	17.35	11.09	18.85	17.96	17.3	15.47	**16.15**	**10.56**
Credit6000	28.5	107.80	27.25	169.12	25.9	140.37	**18.3**	**67.91**
Dnatest	83.1	24.48	84.1	39.72	83	35.23	**81.7**	**24.02**
Permission	**33**	**88.98**	38.1	152.64	36.5	133.04	59.85	115.54
Spambase	**29.55**	**69.06**	31.2	113.10	30.65	99.57	38.35	81.44
SPECTheart	**4.65**	**11.73**	7.35	18.64	6.2	16.48	7.65	10.99
Waveform	20.8	**31.91**	20.35	51.19	21.4	45.00	25.9	35.80
WDBC	**3.55**	**12.44**	4	20.31	4.35	17.40	5.05	11.25

of finding the globally optimal solution, which explains the fact that SASODE selects the smallest number of features on the waveform, but takes a much longer time to find the optimal solution than JAYA.

Combined with the in-depth analysis of the numerical results in Table 12, it is apparent that SASODE exhibits the most superior classification accuracy when the quantity of features is not considered. The features selected by WOA are few, but have the lowest accuracy rate. In other words, the features selected by WOA are not representative. Similarly, the same is true for JAYA. Upon further examination of the experimental results, it has been deduced that the variances between the quantities of features chosen by SASODE, WOA, and JAYA do not exceed three, the time taken for feature selection does not surpass 50%. Consequently, in instances where high accuracy is ensured, the difference between these two indicators may be disregarded. We can even boldly infer that the improved algorithm is applicable to smaller data sets and can guarantee high accuracy without consuming too much time. In conclusion, SASODE outperforms other comparison algorithms in terms of comprehensive performance.

6 Conclusions

In this paper, a novel evolutionary algorithm, termed SASODE, is proposed specifically for solving optimization problems. To distinguish population-based OBL in ODE, first, SASODE proposes opposite operation based on subpopulation, an idea that fully mobilizes the advantages of individuals and maximizes the utilization of individuals in OBL population. Second, an adaptive parameter control strategy is introduced in SASODE to determine the number of individuals eligible to perform OBL and to adjust the size of the subpopulation. Finally, the concept of surviving individuals in subpopulations based on historical information is proposed to guide the algorithm search in a more favourable direction for convergence. Experimental results show that SASODE outperforms state-of-the-art algorithms in recent years on different types of optimization problems, confirming the effectiveness and robustness of the new mechanism in SASODE.

In future research work, ODE still remains much room for further improvements, and several research directions can be recommended. The performance impact of subpopulation strategies in different opposition-based learning. The combination of different adaptive strategies and subpopulation strategies in DE can be further considered. SASODE is currently only used in single-objective optimization problems, and the performance on multi-objective optimization problems may be an exciting research work.

Appendix A: Engineering optimization problem model

A.1 Parameter estimation for frequency-modulated sound waves (FM)

The first problem is Frequency modulation sound wave synthesis (FM) . This problem has six variables, namely, $\vec{X} = \{a_1, \omega_1, a_2, \omega_2, a_3, \omega_3\}$ are called estimated parameters. Additional details regarding this function have been included below:

$$y(t) = a_1 \cdot \sin(\omega_1 \cdot t \cdot \theta + a_2 \cdot \sin(\omega_2 \cdot t \cdot \theta + a_3 \cdot \sin(\omega_3 \cdot t \cdot \theta))) \tag{A1}$$

$$y_0(t) = (1.0) \cdot \sin((5.0) \cdot t \cdot \theta - (1.5) \cdot \sin((4.8) \cdot t \cdot \theta + (2.0) \cdot \sin((4.9) \cdot t \cdot \theta))) \tag{A2}$$

where $\theta = 2\pi/100$, the range of the FM function lies between [-6.4,6.35].

$$f(\vec{X}) = \sum_{t=0}^{100} (y(t) - y_0(t))^2 \tag{A3}$$

A.2 Lennard-Jones potential problem (LJ)

The objective of the following multimodal optimization problem is to minimize the energy of a pure Lennard-Jones (LJ) cluster. A large number of local optima exist in this problem [52]. Given M atoms and their Cartesian coordinates $\vec{a}_i = \{\vec{x}_i, \vec{y}_i, \vec{z}_i\}$, where i ranges from 1 to M, the objective function can be expressed as follows:

$$\mathrm{V}_M(a) = \sum_{i=1}^{M-1} \sum_{j=i+1}^{M} \left(r_{ij}^{-12} - 2 \cdot r_{ij}^{-6} \right) \tag{A4}$$

where $r_{ij} = ||\vec{a}_i - \vec{a}_j||_2$ with gradient

$$\nabla_j V_M(a) = -12 \sum_{i=1, i \neq j}^{M} \left(r_{ij}^{-14} - r_{ij}^{-8} \right) \left(\vec{a}_j - \vec{a}_i \right), \quad j = 1, \ldots, M \tag{A5}$$

The three variables take values in the range $x_1 \in [0, 4]$, $x_2 \in [0, 4]$, $x_3 \in [0, \pi]$. The coordinates x_i for other atom is given to be bound in the range $\left[-4 - \frac{1}{4} \left\lfloor \frac{i-4}{3} \right\rfloor, 4 + \frac{1}{4} \left\lfloor \frac{i-4}{3} \right\rfloor\right]$.

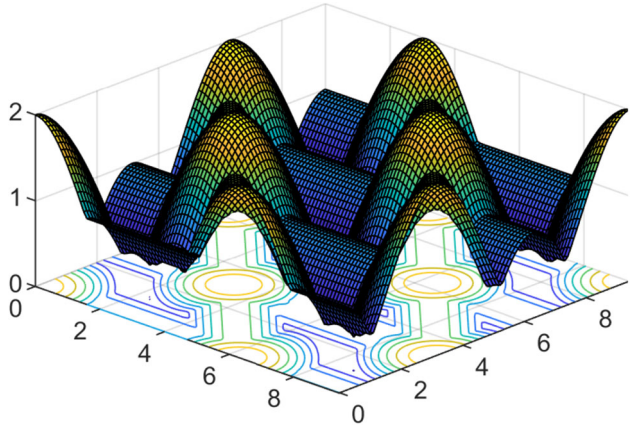


Fig. 7 Landscape of SPRP (D=2)

A.3 Spread spectrum radar polly phase code design (SPRP)

The objective function of the SPRP problem is a nonlinear and non-convex function. The mathematical model of the problem is shown below [62].

$$\begin{aligned}
&\text{global } \min_{y\in Y} f(y) = \max\{\phi_1(y), ..., \phi_{2m}(y)\}\\
&Y = \{(y_1, ..., y_n) \in R^n \mid 0 \le y_j \le 2\pi,\ j = 1, ..., n\}\\
&\text{where } m = 2n - 1 \text{ and}\\
&\phi_{2i-1}(y) = \sum_{j=i}^{n} \cos\left(\sum_{k=|2i-j-1|+1}^{j} y_k\right),\ i = 1, ..., n\\
&\phi_{2i}(y) = 0.5 + \sum_{j=i+1}^{n} \cos\left(\sum_{k=2i-j|+1}^{j} y_k\right),\ i = 1, ..., n-1\\
&\phi_{m+i}(y) = -\phi_i(y), \quad i = 1, ..., m
\end{aligned} \tag{A6}$$

Figure 7 shows the objective function at $D = 2$. Since the problem is NP-hard and the problem is segmented and smoothed, it cannot be solved efficiently using traditional optimization algorithms.

Funding The authors are grateful for the support of National key research and development program of China (2020YFA0908300), and National Natural Science Foundation of China (21878081).

Availability of data and materials Written informed consent for publication of this paper was obtained from all authors.

Declarations

Competing interests The author(s) declared no potential conflicts of interest with respect to the research, author- ship, and/or publication of this article.

References

1. Gao D, Wang G-G, Pedrycz W (2020) Solving fuzzy job-shop scheduling problem using DE algorithm improved by a selection mechanism. IEEE Trans Fuzzy Syst 28(12):3265–3275. https://doi.org/10.1109/tfuzz.2020.3003506
2. Tirkolaee EB, Goli A, Weber G-W (2020) Fuzzy mathematical programming and self-adaptive artificial fish swarm algorithm for just-in-time energy-aware flow shop scheduling problem with outsourcing option. IEEE Trans Fuzzy Syst 28(11):2772–2783
3. Tirkolaee EB, Alinaghian M, Hosseinabadi AAR, Sasi MB, Sangaiah AK (2019) An improved ant colony optimization for the multi-trip capacitated arc routing problem. Comput Electr Eng 77:457–470
4. Alswaitti M, Albughdadi M, Isa NAM (2019) Variance-based differential evolution algorithm with an optional crossover for data clustering. Appl Soft Comput 80:1–17. https://doi.org/10.1016/j.asoc.2019.03.013
5. Zhang Y, Gong D-w, Gao X-z, Tian T, Sun X-y (2020) Binary differential evolution with self-learning for multi-objective feature selection. Inf Sci 507:67–85. https://doi.org/10.1016/j.ins.2019.08.040
6. Sharafi Y, Teshnehlab M (2021) Opposition-based binary competitive optimization algorithm using time-varying v-shape transfer function for feature selection. Neural Comput & Applic 33(24):17497–17533. https://doi.org/10.1007/s00521-021-06340-9
7. Elaziz MA, Dahou A, Abualigah L, Yu L, Alshinwan M, Khasawneh AM, Lu S (2021) Advanced metaheuristic optimization techniques in applications of deep neural networks: a review. Neural Comput & Applic 33(21):14079–14099. https://doi.org/10.1007/s00521-021-05960-5
8. Abualigah L, Yousri D, Elaziz MA, Ewees AA, Al-qaness MAA, Gandomi AH (2021) Aquila optimizer: a novel meta-heuristic optimization algorithm. Comput Ind Eng 157:107250. https://doi.org/10.1016/j.cie.2021.107250
9. Fausto F, Reyna-Orta A, Cuevas E, Andrade ÁG, Perez-Cisneros M (2019) From ants to whales: metaheuristics for all tastes. Artif Intell Rev 53(1):753–810. https://doi.org/10.1007/s10462-018-09676-2
10. Ser JD, Osaba E, Molina D, Yang X-S, Salcedo-Sanz S, Camacho D, Das S, Suganthan PN, Coello CAC, Herrera F (2019) Bio-inspired computation: where we stand and what's next. Swarm Evol Comput 48:220–250. https://doi.org/10.1016/j.swevo.2019.04.008
11. Holland J (1975) Adaptation in natural and artificial systems: an introductory analysis with application to biology. Control Artif Intell
12. Storn R, Price K (1997) Differential evolution - a simple and efficient heuristic for global optimization over continuous spaces. J Glob Optim 11(4):341–359. https://doi.org/10.1023/a:1008202821328
13. 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
14. Mirjalili S (2016) SCA: A sine cosine algorithm for solving optimization problems. Knowl-Based Syst 96:120–133. https://doi.org/10.1016/j.knosys.2015.12.022
15. Faramarzi A, Heidarinejad M, Stephens B, Mirjalili S (2020) Equilibrium optimizer: a novel optimization algorithm. Knowl-Based Syst 191:105190. https://doi.org/10.1016/j.knosys.2019.105190
16. Li Y, Wang S (2019) Differential evolution algorithm with elite archive and mutation strategies collaboration. Artif Intell Rev 53(6):4005–4050. https://doi.org/10.1007/s10462-019-09786-5

17. Vasant P, Zelinka I, Weber G-W (2019) Intelligent Computing & Optimization. Springer, ???
18. Zeng Z, Zhang M, Chen T, Hong Z (2021) A new selection operator for differential evolution algorithm. Knowl-Based Syst 226:107150. https://doi.org/10.1016/j.knosys.2021.107150
19. Rosić MB, Simić MI, Pejović PV (2021) An improved adaptive hybrid firefly differential evolution algorithm for passive target localization. Soft Comput 25(7):5559–5585. https://doi.org/10.1007/s00500-020-05554-8
20. Deng L-B, Li C-L, Sun G-J (2020) An adaptive dimension level adjustment framework for differential evolution. Knowl-Based Syst 206:106388. https://doi.org/10.1016/j.knosys.2020.106388
21. Tizhoosh HR (2005) Opposition-based learning: a new scheme for machine intelligence. In: International conference on computational intelligence for modelling, control and automation and international conference on intelligent agents, web technologies and internet commerce (CIMCA-IAWTIC'06). IEEE, vol 1, pp 695–701
22. Mahdavi S, Rahnamayan S, Deb K (2018) Opposition based learning: a literature review. Swarm Evol Comput 39:1–23. https://doi.org/10.1016/j.swevo.2017.09.010
23. Rahnamayan S, Tizhoosh HR, Salama MMA (2008) Opposition-based differential evolution. IEEE Trans Evol Comput 12(1):64–79. https://doi.org/10.1109/tevc.2007.894200
24. Choi TJ, Togelius J, Cheong Y-G (2021) A fast and efficient stochastic opposition-based learning for differential evolution in numerical optimization. Swarm Evol Comput 60:100768
25. Choi TJ (2023) A rotationally invariant stochastic opposition-based learning using a beta distribution in differential evolution. Expert Syst Appl 120658
26. Mohapatra S, Mohapatra P (2023) Fast random opposition-based learning golden jackal optimization algorithm. Knowl-Based Syst 110679
27. Wang Z, Huang L, Yang S, Li D, He D, Chan S (2023) A quasi-oppositional learning of updating quantum state and q-learning based on the dung beetle algorithm for global optimization. Alexandria Eng J 81:469–488
28. Li AD, Xue B, Zhang M (2021) Improved binary particle swarm optimization for feature selection with new initialization and search space reduction strategies. Appl Soft Comput 106(Mar.):107302. https://doi.org/10.1016/j.asoc.2021.107302
29. Zhang Y, Wang S, Phillips P, Ji G (2014) Binary pso with mutation operator for feature selection using decision tree applied to spam detection. Knowl Based Syst 64(jul.):22–31. https://doi.org/10.1016/j.knosys.2014.03.015
30. Nssibi M, Manita G, Korbaa O (2023) Advances in nature-inspired metaheuristic optimization for feature selection problem: a comprehensive survey. Comput Sci Rev 49:100559
31. Wang Y, Ran S, Wang G-G (2023) Role-oriented binary grey wolf optimizer using foraging-following and lévy flight for feature selection. Appl Math Model
32. Tubishat M, Idris N, Shuib L, Abushariah MA, Mirjalili S (2020) Improved salp swarm algorithm based on opposition based learning and novel local search algorithm for feature selection. Expert Syst Appl 145:113122. https://doi.org/10.1016/j.eswa.2019.113122
33. Hamidzadeh J et al (2021) Feature selection by using chaotic cuckoo optimization algorithm with levy flight, opposition-based learning and disruption operator. Soft Comput 25(4):2911–2933
34. Wolpert DH, Macready WG (1997) No free lunch theorems for optimization. IEEE Trans Evol Comput 1(1):67–82. https://doi.org/10.1109/4235.585893
35. Lacerda MGP, Araujo Pessoa LF, Lima Neto FB, Ludermir TB, Kuchen H (2021) A systematic literature review on general parameter control for evolutionary and swarm-based algorithms. Swarm Evol Comput 60:100777. https://doi.org/10.1016/j.swevo.2020.100777
36. Tang L, Dong Y, Liu J (2015) Differential evolution with an individual-dependent mechanism. IEEE Trans Evol Comput 19(4):560–574. https://doi.org/10.1109/tevc.2014.2360890
37. Cui L, Li G, Lin Q, Chen J, Lu N (2016) Adaptive differential evolution algorithm with novel mutation strategies in multiple sub-populations. Comput Oper Res 67:155–173. https://doi.org/10.1016/j.cor.2015.09.006
38. Lin X, Luo W, Xu P (2021) Differential evolution for multimodal optimization with species by nearest-better clustering. IEEE Trans Cybern 51(2):970–983. https://doi.org/10.1109/tcyb.2019.2907657
39. Črepinšek M, Liu S-H, Mernik M (2013) Exploration and exploitation in evolutionary algorithms. ACM Comput Surv 45(3):1–33. https://doi.org/10.1145/2480741.2480752
40. Zhang J, Sanderson AC (2009) JADE: Adaptive differential evolution with optional external archive. IEEE Trans Evol Comput 13(5):945–958. https://doi.org/10.1109/tevc.2009.2014613
41. Tanabe R, Fukunaga A (2013) Success-history based parameter adaptation for differential evolution. In: 2013 IEEE Congress on Evolutionary Computation, pp 71–78. IEEE
42. Tanabe R, Fukunaga AS (2014) Improving the search performance of SHADE using linear population size reduction. In: 2014 IEEE Congress on Evolutionary Computation (CEC). IEEE, ???. https://doi.org/10.1109/cec.2014.6900380
43. Yao X, Liu Y, Lin G (1999) Evolutionary programming made faster. IEEE Trans Evol Comput 3(2):82–102. https://doi.org/10.1109/4235.771163
44. Askari Q, Saeed M, Younas I (2020) Heap-based optimizer inspired by corporate rank hierarchy for global optimization. Expert Syst Appl 161:113702. https://doi.org/10.1016/j.eswa.2020.113702
45. Mirjalili S, Mirjalili SM, Lewis A (2014) Grey wolf optimizer. Adv Eng Softw 69:46–61. https://doi.org/10.1016/j.advengsoft.2013.12.007
46. Mirjalili S, Lewis A (2016) The whale optimization algorithm. Adv Eng Softw 95:51–67. https://doi.org/10.1016/j.advengsoft.2016.01.008
47. Mirjalili S (2015) Moth-flame optimization algorithm: a novel nature-inspired heuristic paradigm. Knowl-Based Syst 89:228–249. https://doi.org/10.1016/j.knosys.2015.07.006
48. Mirjalili S, Gandomi AH, Mirjalili SZ, Saremi S, Faris H, Mirjalili SM (2017) Salp swarm algorithm: A bio-inspired optimizer for engineering design problems. Adv Eng Softw 114:163–191. https://doi.org/10.1016/j.advengsoft.2017.07.002
49. Derrac J, García S, Molina D, Herrera F (2011) A practical tutorial on the use of nonparametric statistical tests as a methodology for comparing evolutionary and swarm intelligence algorithms. Swarm Evol Comput 1(1):3–18. https://doi.org/10.1016/j.swevo.2011.02.002
50. Kudo F, Yoshikawa T, Furuhashi T (2011) A study on analysis of design variables in pareto solutions for conceptual design optimization problem of hybrid rocket engine. In: 2011 IEEE Congress of evolutionary computation (CEC), pp 2558–2562. IEEE
51. Awad N, Ali M, Liang J, Qu B, Suganthan P (2016) Problem definitions and evaluation criteria for the cec 2017 special session and competition on single objective bound constrained real-parameter numerical optimization. In: Technical report, pp 1–34. Nanyang Technological University Singapore, ???
52. Das S, Suganthan PN (2010) Problem definitions and evaluation criteria for cec 2011 competition on testing evolutionary algorithms on real world optimization problems. Jadavpur University, Nanyang Technological University, Kolkata, pp 341–359
53. Ahmadianfar I, Heidari AA, Gandomi AH, Chu X, Chen H (2021) RUN beyond the metaphor: an efficient optimization algorithm based on runge kutta method. Expert Syst Appl 181:115079. https://doi.org/10.1016/j.eswa.2021.115079

54. Houssein EH, Saad MR, Hashim FA, Shaban H, Hassaballah M (2020) Lévy flight distribution: a new metaheuristic algorithm for solving engineering optimization problems. Eng Appl Artif Intell 94:103731. https://doi.org/10.1016/j.engappai.2020.103731
55. Dua D, Graff C (2017) UCI Machine Learning Repository. http://archive.ics.uci.edu/ml
56. Wang L, Gao Y, Gao S, Yong X (2021) A new feature selection method based on a self-variant genetic algorithm applied to android malware detection. Symmetry 13(7):1290. https://doi.org/10.3390/sym13071290
57. Rahnamayan S, Jesuthasan J, Bourennani F, Salehinejad H, Naterer GF (2014) Computing opposition by involving entire population. In: 2014 IEEE Congress on evolutionary computation (CEC), pp 1800–1807. IEEE
58. Xu Wang L, He B, Wang N (2011) Modified opposition-based differential evolution for function optimization. J Comput Inf Syst 7(5):1582–1591
59. Ergezer M, Simon D, Du D (2009) Oppositional biogeography-based optimization. In: 2009 IEEE International conference on systems, man and cybernetics, pp 1009–1014. IEEE
60. Rao RV (2016) Jaya: a simple and new optimization algorithm for solving constrained and unconstrained optimization problems. Int J Ind Eng Comput 19–34. https://doi.org/10.5267/j.ijiec.2015.8.004
61. Bing X, Zhang M, Browne WN, Xin Y (2016) A survey on evolutionary computation approaches to feature selection. IEEE Trans Evol Comput 20(4):606–626. https://doi.org/10.26686/wgtn.14214497.v1
62. Mladenović N, Petrović J, Kovačević-Vujčić V, Čangalović M (2003) Solving spread spectrum radar polyphase code design problem by tabu search and variable neighbourhood search. Eur J Oper Res 151(2):389–399. https://doi.org/10.1016/s0377-2217(02)00833-0

Le Wang received the M.S. degree from the School of Information Science and Engineering, North Minzu University, Yinchuan, China. She is currently pursuing her Ph.D. degree in the School of Information Science and Engineering, East China University of Science and Technology, Shanghai, China. Her current research interests include evolutionary computation, multi-objective optimization and surrogate-assisted optimization.

Jiahang Li received the. M.Sc. degree from the School of Mathematics and Information Science, North Minzu University, Yinchuan, China. He is currently pursuing the Ph.D. degree with the School of Mechanical Science and Engineering, Huazhong University of Science and Technology, Wuhan, China. His current research interests include evolutionary computation and its applications in production scheduling.

Xuefeng Yan received the B.S. degree in biochemical engineering and the Ph.D. degree in control theory engineering from Zhejiang University, Hangzhou, China, in 1995 and 2002, respectively. He is now a professor of East China University of Science and Technology, Shanghai, China. His research interests include complex chemical and biological process modeling, optimizing and controlling, process monitoring, fault diagnosis, and intelligent information processing.