IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received January 3, 2020, accepted January 22, 2020, date of publication February 3, 2020, date of current version February 10, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.2970992

A Symmetric Sine Cosine Algorithm With Adaptive Probability Selection

BIN WANG[1], TIAN XIANG[1], NING LI[2], WENJUAN HE[1], WEI LI[1], AND XINHONG HEI[1]
[1]Faculty of Computer Science and Engineering, Xi'an University of Technology, Xi'an 710048, China
[2]Faculty of Computer, Baoji University of Arts and Sciences, Baoji 721013, China

Corresponding author: Xinhong Hei (heixinhong@xaut.edu.cn)

This work was supported in part by the National Natural Science Foundation of China under Project Codes 61803301, 61773313, and 61773314, in part by the National Key Research and Development Program of China under Grant 2018YFB1201500, in part by the Key Research and Development Program of Shaanxi Province under Grant 2017ZDXM-GY-098, and in part by the Science and Technology Innovation Team Plan of Shaanxi Province under Grant 2019TD-014.

ABSTRACT To improve the ability of the sine cosine algorithm (SCA) in the exploitation process, an improved symmetric SCA with adaptive probability selection (SSCA-APS), is proposed. The search process of this algorithm is divided into early and late stages. In the early stage, the operators of the traditional SCA algorithm continue to be used. In the late stage, three improvements were applied. Firstly, the symmetric sine and cosine operators are proposed. The adaptive probability selection strategy is adopted to integrate original sine and cosine operators and symmetric sine and cosine operators for dynamically adjusting the step size of the search range. Furthermore, to prevent the population from falling into local optimization, Gaussian perturbation is used to mutate the globally optimal individuals of the current generation. In addition, the information of two randomly selected individuals and the globally optimal individual is integrated by quadratic interpolation to maintain population diversity and produce a new individual. 23 test functions were used to verify the performance of the proposed algorithm. The simulation results indicate that the performance of the SSCA-APS algorithm has competitiveness when it is compared with classical SCA and some state-of-the-art SCA variants.

INDEX TERMS Symmetric sine cosine, quadratic interpolation, Gauss perturbation, adaptive probability selection.

I. INTRODUCTION

For numerical optimization problems, researchers have created various solutions, some of which present pioneering algorithms, such as SCA [1], which is the first algorithm to explore the optimal solution in the search space by using standard sine and cosine function.

To improve the performance of the SCA algorithm, some existing algorithms are integrated into SCA. For example, Nenavath and Jatoth proposed the hybrid algorithm SCA-DE [2], which mixed SCA with Differential Evolution, and experiments on unimodal, multimodal and fixed-dimensional multimodal functions showed a competitive performance. Singh and Singh proposed the hybrid GWOSCA [3] as a combination of Grey Wolf Optimizer used for exploitation and SCA used for exploration, and experimental results proved that this hybrid variant can highly be effective in solving constrained or unconstrained problems.

The associate editor coordinating the review of this manuscript and approving it for publication was Huaqing Li.

Elaziz *et al*. presented an improved SCA (named opposition based SCA, for short OBSCA [4]) that considered opposition based learning as a mechanism for a better exploration of the search space to generate more accurate solutions. Meshkat and Parhizgar, in their algorithm called WUPMSCA [5], used a weighted update position mechanism instead of the position update method of SCA to obtain better solution. Zamli *et al.* proposed a hybrid Q-learning sine cosine based strategy, called the Q-learning sine-cosine (QLSCA [6]) algorithm, which eliminate the switching probability, and dynamically identified the best operation during runtime. And they integrated Levy flight motion and crossover into the QLSCA to enhance the solution diversity.

There are also some improved SCA. ISCA [7] changed the position update equation by introducing an inertia weight to avoid falling into local optima and added a non-linear conversion parameter decreasing strategy based on the Gaussian function to solve the curse of dimensionality and to accelerate convergence. Meshkat and Parhizgar used the sine function to update the position of each individual based on the current

best or a random individual's position in an algorithm called SOA [8], which outperformed SCA in terms of accuracy and convergence speed. Chiwen Qu *at el.*, in their algorithm MSCA [9], adopted the method of exponential decreasing conversion parameter and linear decreasing inertia weight to balance the exploration and exploitation, and used the greedy Levy mutation strategy and the random individuals near the optimal individual to replace the optimal individual. Gholizadeh and Sojoudizadeh introduced a modified SCA (MSCA [10]) for discrete sized optimization of truss structures, by replacing some of the worst individuals with variants of the current global best solution. Gupta and Deepan attempted to jump out from the local optima by proposing a modified version of SCA named as modified Sine Cosine Algorithm (m-SCA [11]), in which used opposite numbers based on perturbation rate to generate the opposite population. Then they introduced an improved version of sine cosine algorithm was named as ISCA [12], which hybridization of exploitation skills of crossover with optimal solutions and global search mechanisms and integration of self-learning. Dunia and Ramzy proposed methods of chaotic sine cosine algorithm (CSCAs [13]) that included replaced random parameters with the chaotic sequences, and implemented five one dimensional various chaotic maps. The improvement of the above algorithm focuses on the population level. The sine cosine algorithm and its variants have been widely used in many aspects, such as the wing design of aircraft problem [14], Levy flight problem [15], and optimal reactive power allocation (ORPD [16]) problems.

Nevertheless, these algorithms do not consider the essential characteristics of sine and cosine functions. In the sine and cosine function curves, the probability density function and probability distribution function of the random mutation are constant regardless of the early stage or the late stage. In other words, the probability that the value of the mutation function is greater than the intermediate value (as in Figure 1 and Figure 2, functions "Y1" and "Y2" take the intermediate value 0.5) is greater than the probability that it is less than the intermediate value, which may affect the convergence speed of the algorithm. Detailed explanation and proof will be given later part III. Therefore, neither the original SCA algorithm nor other improved algorithms have the symmetric sine and cosine operators (see Figure 7 and Figure 8) proposed in this paper.

As we all know, there is no free lunch, and no algorithm can solve all problems. Therefore, researchers have made efforts on balancing algorithms exploitation and exploration capabilities. In recent years, many theoretical studies have been carried out to enhance the performance of the SCA further. Thus, the main goal of this paper is to accelerate the convergence speed of the algorithm further and avoid falling into local optima. The work of this paper is to dig out the characteristics of sine and cosine operators as far as possible and to accelerate the convergence speed of the algorithm. This algorithm uses a set of symmetric sine and cosine functions by flipping the original sine cosine curve in accordance with

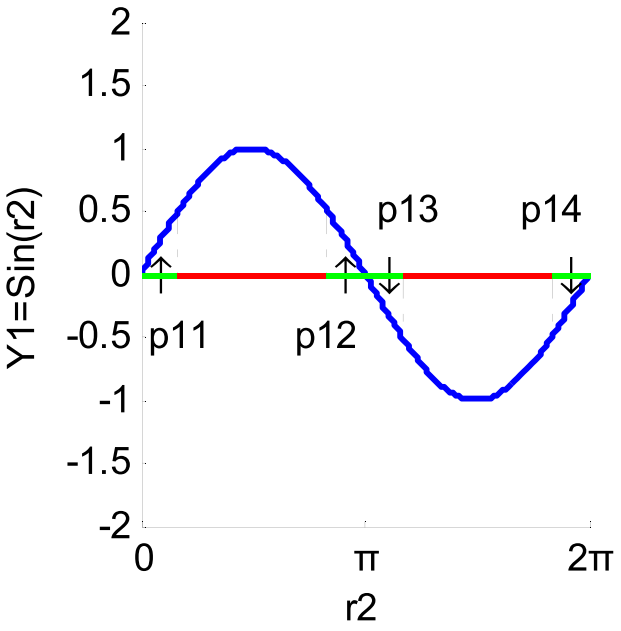


FIGURE 1. Probability density diagram of sine function.

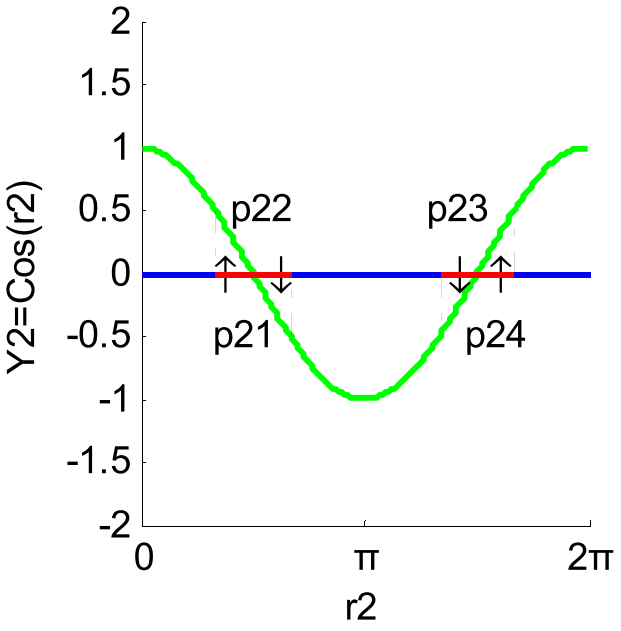


FIGURE 2. Probability density diagram of cosine function.

lines parallel to the X axis (abscissa axis). In the meantime, it uses adaptive probability selection [17], [18], mechanism to select appropriate operators, which makes the selection of strategies more diverse. Besides, in order to further improve the exploitation ability of the algorithm, the idea of quadratic interpolation [19] was also incorporated to update the position of poor individuals. A set of test functions was used to evaluate the proposed algorithm and compare it with the original SCA algorithm and some improved versions.

To sum up, the specific work of this article is as follows:

1) Due to the limitations of original sine and cosine operators, symmetric sine and cosine operators are proposed to enhance the search performance of the algorithm.

2) Adaptive probability selection was used to integrate original operators and the proposed operators.
3) Aiming at the problem of SCA falling into premature convergence, Gaussian perturbation is introduced, which may make the search jump out of the local optima, and therefore, increase the accuracy of the search.
4) The quadratic interpolation strategy is introduced to promote convergence while maintaining the diversity of the population.
5) Experiments were conducted to verify the performance of the algorithm.

The rest of this paper is organized as follows: Part II reviews the SCA algorithm. Part III, according to the characteristics of the SCA algorithm, explains the motivation behind the SSCA-APS algorithm and its details. Part IV describes the setup and the results of the conducted experiments. Part V is the conclusion of this paper. Putting forward the future work is in Part VI.

II. SCA ALGORITHM

Sine cosine algorithm (SCA) is a population-based optimization technology. Through iteration, the position of individuals in the population is updated continuously to obtain the optimal solution. The position update equation is the following (1):

$$x_i^{t+1} = \begin{cases} x_i^t + r_1 \sin(r_2) \left| r_3 Pbest_i^t - x_i^t \right|, & if\ r_4 < 0.5 \\ x_i^t + r_1 \cos(r_2) \left| r_3 Pbest_i^t - x_i^t \right|, & if\ r_4 \geq 0.5 \end{cases} \quad (1)$$

where "x_i^t" is the position of the current individual in the current population at the generation t, "r_1", "r_2", and "r_3" are three random parameters, "r_2" is the random value between $[0, 2\pi]$ (for convenience, it is analyzed only between intervals $[0, \pi/2]$), "r_3" is the random value uniformly distributed between [0, 2], "$Pbest_i^t$" is the best optimal individual of population in the t generation, "|A|" represents the absolute value of "A", and "r_4" is a random value between [0, 1]. It is remarkable that "r_1" decreases with the number of iterations following the next formula (2):

$$r_1 = a - t\frac{a}{T} \quad (2)$$

where t is the number of the current iteration, T is the maximum number of iterations, and a is a positive integer, the value in this paper is 2, it can be known from (2) that the value range of "r1" is in interval [0, 2] and its initial value is 2. Algorithm 1 in Table 1 shows the main steps of the SCA algorithm.

III. SSCA-APS ALGORITHM

This section describes in detail the motivation for the SSCA-APS algorithm. Moreover, it narrates the specific implementation of its components for the overall algorithm.

TABLE 1. SCA algorithm.

Algorithm 1: SCA Algorithm
Initial population (solution set) (X)
1. **While** ($t \leq T$)
2. Evaluate each individual in the population according to the objective function value.
3. Find the individual with the best objective function value (*Pbest*).
4. Update the position of the individual in the population with (1).
5. Update corresponding parameters by (2).
6. **End While**
Output: the optimal solution obtained so far is the global optimal solution output.

A. MOTIVATION

1) PROBABILITY CALCULATION OF RANDOM MUTATION FUNCTION

Based on literature search and research in related fields, we found that SCA and its variants have a common limitation. Therefore, the following is only discussed with SCA as an example. In the original SCA algorithm, the random mutation probability of an individual was calculated by the functions "Y_1" and "Y_2", as in (3):

$$\begin{cases} Y_1 = \begin{cases} \sin(r2), & 0 \leq r2 \leq 2\pi \\ 0, & otherwise \end{cases} \\ Y_2 = \begin{cases} \cos(r2), & 0 \leq r2 \leq 2\pi \\ 0, & otherwise \end{cases} \end{cases} \quad (3)$$

where, $r2$ is a continuous random variable uniformly distributed, $r2 \sim U[0, 2\pi]$, and its probability density function "f" is:

$$\mathrm{f}(r2) = \begin{cases} \dfrac{1}{2\pi}, & 0 \leq r2 \leq 2\pi \\ 0, & otherwise \end{cases} \quad (4)$$

The probability distribution function of "Y_1":

$$\begin{aligned} \boldsymbol{F}_{Y_1} &= P\{Y_1 \leq y_1\} \\ &= P\{\sin(r2) \leq y_1\} \\ &= P\{0 \leq r2 \leq \arcsin(y_2)\} \\ &= \int_0^{\arcsin(y_1)} \frac{1}{2\pi} dx \end{aligned} \quad (5)$$

The probability density of "Y_1" is as follows:

$$P_{Y_1}(\mathrm{y}_1) = F'_{Y_1}(y_1) = \left[\int_0^{\arcsin(y_1)} \frac{1}{2\pi} dx \right]' = \frac{1}{2\pi\sqrt{1-y_1^2}} \quad (6)$$

The probability distribution function of "Y_2":

$$\begin{aligned} \boldsymbol{F}_{Y_2} &= P\left\{Y_2 \leq \mathrm{y}_2\right\} \\ &= P\{\cos(r2) \leq y_2\} \\ &= P\left\{\arccos(y_2) \leq r2 \leq \frac{\pi}{2}\right\} \\ &= \int_{\arccos(y_2)}^{\frac{\pi}{2}} \frac{1}{2\pi} dx \end{aligned} \quad (7)$$

The probability density of "Y_2" is as follows:

$$P_{Y_2}(y_2)=F'_{Y_2}(y_2)=\left[\int_{\arccos(y_2)}^{\frac{\pi}{2}}\frac{1}{2\pi}dx\right]'=\frac{1}{2\pi\sqrt{1-y_2^2}} \quad (8)$$

Without loss of generality:

When the value of the function "Y_1" and "Y_2" is within the range of [−1/2, 1/2], which is considered to promote the individuals to produce small mutations, that is, the small step.

When the value of the function "Y_1"and "Y_2" is in the range of [−1, −1/2] or [1/2, 1], it is considered that it causes individuals to produce large mutations, that is, large step, as in Figure 1 and Figure 2.

For function "Y_1":

When $r2 \in [0, \pi/2]$, the probability of "Y_1" falling within the interval [0, 1/2] is:

$$P_{11}(\frac{1}{2})=\int_0^{\frac{1}{2}}P_{Y_1}(\frac{1}{2})dx=\frac{1}{2\pi}(\arcsin(\frac{1}{2})-\arcsin(0))=\frac{1}{12} \quad (9)$$

When $r2 \in [\pi/2, \pi]$, the probability of "Y_1" falling within the interval [0, 1/2] is:

$$P_{12}(\frac{1}{2})=\frac{1}{12} \quad (10)$$

When $r2 \in [\pi, 3\pi/2]$, the probability of "Y_1" falling within the interval [−1/2, 0] is:

$$P_{13}(-\frac{1}{2})=\frac{1}{12} \quad (11)$$

When $r2 \in [3\pi/2, 2\pi]$, the probability of "Y_1" falling within the interval [−1/2, 0] is:

$$P_{14}(-\frac{1}{2})=\frac{1}{12} \quad (12)$$

The total probability of the function "Y_1" falling within the range of the small step is:

$$P_{smaller}(Y_1)=P_{11}+P_{12}+P_{13}+P_{14}=\frac{1}{3} \quad (13)$$

The total probability of the function "Y_1" falling within the range of large step is:

$$P_{l\arg er}(Y_1)=1-P_{smaller}(Y_1)=\frac{2}{3} \quad (14)$$

By the same token:

The total probability of the function "Y_2" falling within the range of the small step is:

$$P_{smaller}(Y_2)=\frac{1}{3} \quad (15)$$

The total probability of the function "Y_2" falling within the range of large step is:

$$P_{l\arg er}(Y_2)=\frac{2}{3} \quad (16)$$

Generally, in functions "Y_1" and "Y_2", the mutating probability of the small step is less than the mutating probability of the large step.

2) REASONABILITY OF MUTATION STEP SIZE OF SCA

In the SCA algorithm, the actual mutation steps are "r1 sin(r2)" and "r1 cos(r2)". Initial value of "r1" is set to *2* and it gradually reduce with the increase of iterations.

In the early stage of evolution, when $r1 = 2$, the corresponding Figure 3 and Figure 4 as follows:

Observation Can Find That:

1) At the beginning, because $B1+B2 > A1+A2+A3+A4$ in figure 3 and $B1+B2+B3 > A1+A2+A3+A4$ in figure 4, probability of the $|\ r1\ sin(r2)\ |$ and $|$

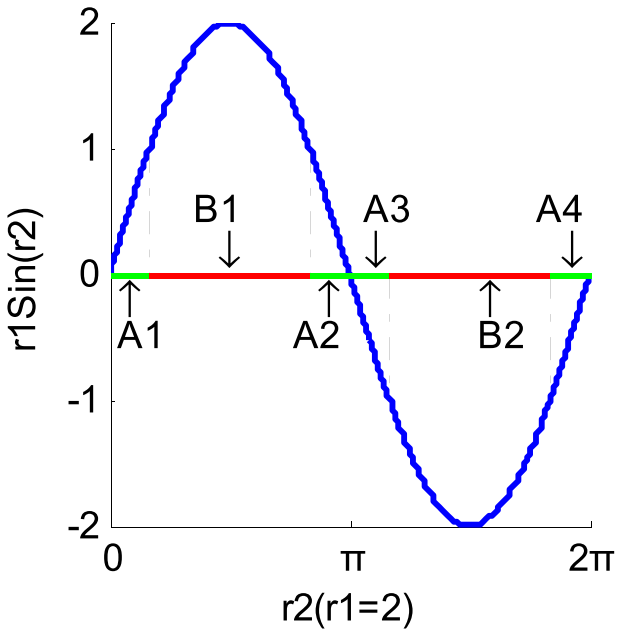


FIGURE 3. The probability density diagram of the sine function in the early stage.

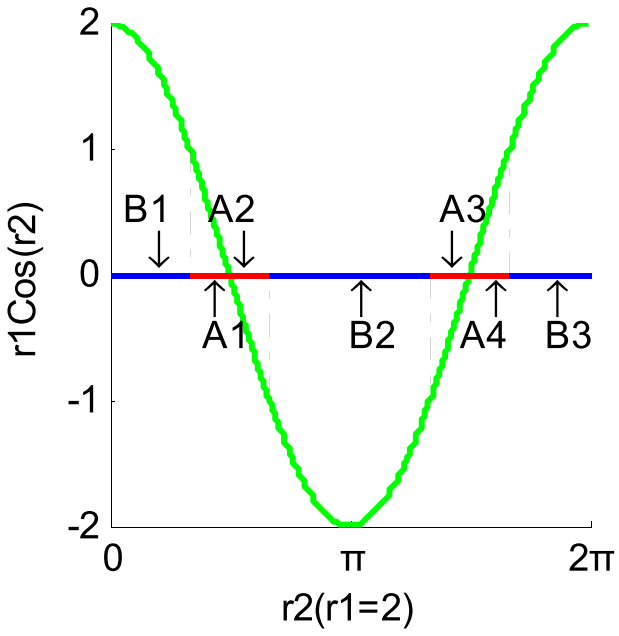


FIGURE 4. The probability density diagram of the cosine function in the early stage.

r1 cos(r2) | greater than 1 is larger, which can increase the step length of mutation. It improves the exploration competence of the algorithm and avoids the phenomenon of prematurity.

2) With the iterative generation increasing, r1 gets smaller and smaller.
3) Then probability of | *r1 sin(r2)* | and | *r1 cos(r2)* | less than 1 is getting larger and larger, which causes the possibility of mutation step length to decrease. It improves the exploitation and gradually enhances convergence ability.

In the late stage of evolution, when $r1 = 1$, as the corresponding Figure 5 and Figure 6.

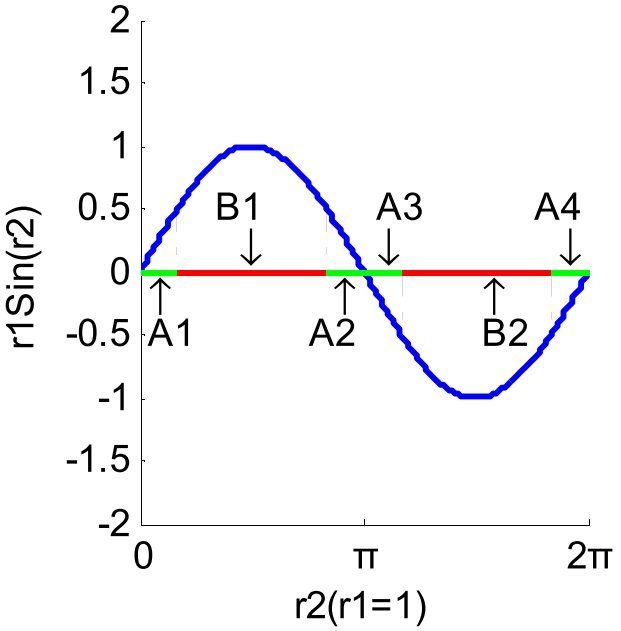


FIGURE 5. The probability density diagram of the sine function in the late stage.

Observation Can Find That:

1) | *r1 sin(r2)* | and | *r1 cos(r2)* | are always less than 1, and the search is into the exploitation process. It can enhance the ability of convergence.
2) As the iterative generation increases, *r1* gets smaller, and local search capabilities is constantly enhanced

General, the original SCA emphasizes exploration ability in the early stage of evolution and focuses on exploitation ability in the late stage of evolution, which is reasonable.

In the late stage of evolution, however, the algorithm has the following problems:

1) Since the shape of sine and cosine function decides that probability of lager step is greater than the small step, it is not conducive to smoothly improve the precision of the local search and convergence of the algorithm.
2) The division between exploration and exploitation process is fixed, and there is no adaptive mechanism.
3) Once the solutions falling into local optimum, it is difficult to get out.

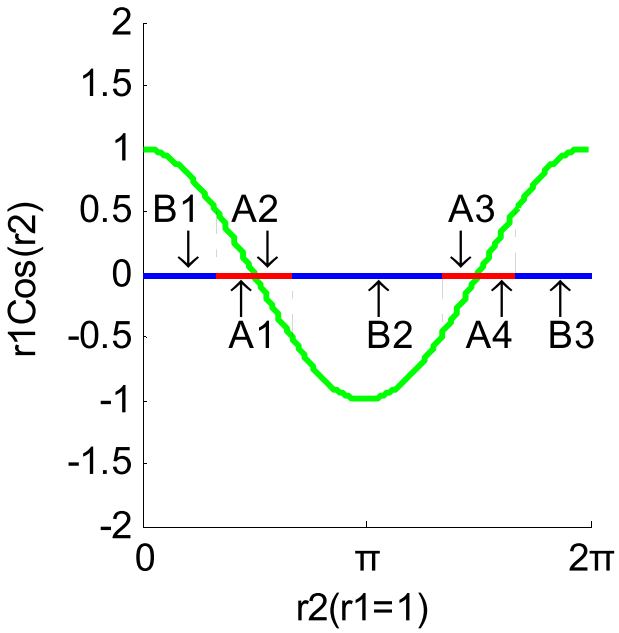


FIGURE 6. The probability density diagram of the cosine function in the late stage.

B. IDEAS TO IMPROVE SCA

1) SYMMETRIC SINE AND COSINE FUNCTIONS

In order to improve the convergence speed and accuracy of the algorithm, sine and cosine functions are flipped horizontally by $y = 0.5$ and $y = -0.5$, and the symmetric sine and cosine functions "Y_3" and "Y_4" will be obtained.

$$\begin{cases} Y_3 = \begin{cases} -\sin(r2) + 1, & 0 \le r\,2 \le \pi \\ -\sin(r2) - 1, & \pi < r2 \le 2\pi \\ 0, & otherwise \end{cases} \\ Y_4 = \begin{cases} -\cos(r2) + 1, 0 \le r2 < \dfrac{\pi}{2}, \dfrac{\pi}{2} & < r2 \le 2\pi \\ -\cos(r2) - 1, & \dfrac{\pi}{2} \le r\,2 \le \dfrac{3\pi}{2} \\ 0, & otherwise \end{cases} \end{cases} \tag{17}$$

The probability distribution function of "Y_3":

$$\begin{aligned} F_{Y_3} &= P\{Y_3 \le y_3\} \\ &= P\{-\sin(r_2) + 1 \le y_3\} \\ &= P\{\arcsin(1 - y_3) \le r_2 \le \frac{\pi}{2}\} \\ &= \int_{\arcsin(1-y_3)}^{\frac{\pi}{2}} \frac{1}{2\pi} dx \end{aligned} \tag{18}$$

The probability density of "Y_3" is as follows:

$$\begin{aligned} P_{y_3}(y_3) &= F'_{y_3}(y_3) \left[\int_{\arcsin(1-3)}^{\frac{\pi}{2}} \frac{1}{2\pi} dx \right]' \\ &= \frac{1}{2\pi\sqrt{1 - (1 - y_3)^2}} \end{aligned} \tag{19}$$

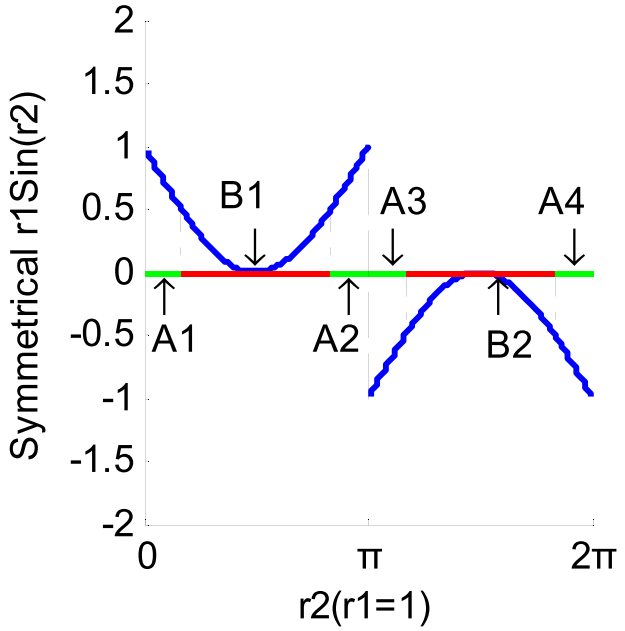


FIGURE 7. The probability density diagram of the symmetrical sine function in the late stage.

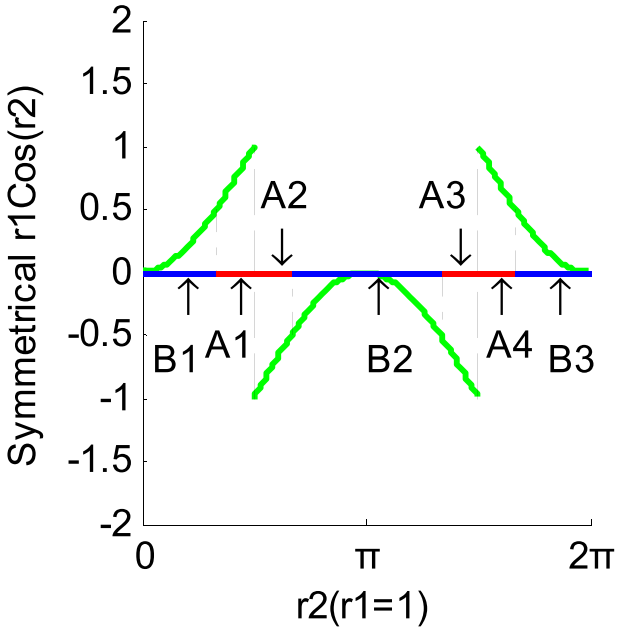


FIGURE 8. The probability density diagram of the symmetrical cosine function in the late stage.

The probability distribution function of "Y_4":

$$\begin{aligned} F_{Y_4} &= P\{Y_4 \le y_4\} \\ &= P\{-\cos(r2) + 1 \le y_4\} \\ &= P\{0 \le r2 \le \arccos(1 - y_4)\} \\ &= \int_0^{\arccos(1-y_4)} \frac{1}{2\pi} dx \end{aligned} \tag{20}$$

The probability density of "Y_4" is as follows:

$$\begin{aligned} P_{y_4}(y_4) &= F'_{y_4}(y_4) \left[\int_0^{\arccos(1-y_4)} \frac{1}{2\pi} \right]' \\ &= \frac{1}{2\pi\sqrt{1-(1-y_4)^2}} \end{aligned} \tag{21}$$

The total probability of the function "Y_3" falling within the range of the smaller step is:

$$P_{\text{smaller}}(Y_3) = \frac{2}{3} \tag{22}$$

The total probability of the function "Y_3" falling within the range of larger step is:

$$P_{l\,\text{arg}\,er}(Y_3) = \frac{1}{3} \tag{23}$$

The total probability of the function "Y_4" falling within the range of the small step is:

$$P_{smaller}(Y_4) = \frac{2}{3} \tag{24}$$

The total probability of the function "Y_4" falling within the range of large step is:

$$P_{l\,\text{arg}\,er}(Y_4) = \frac{1}{3} \tag{25}$$

In summary, in functions "Y_3" and "Y_4", the probability of mutating the large step is less than the probability of mutating the small step.

In the late stage of evolution, the symmetric sine and cosine functions can improve the precision of local search.

2) ADAPTIVE STRATEGY SELECTION

In the late stage of evolution, the method of calculating the success rate of updating is adopted to select the original sine and cosine functions and the symmetric sine and cosine function adaptively. The operation of exploration and exploitation can be adjusted reasonably, which is beneficial to improve the speed and accuracy of convergence.

3) OTHERS

The quadratic interpolation method is introduced to improve the accuracy of the solution and Gaussian mutation will be useful to jump out of the local optimal.

C. SYMMETRIC SINE AND COSINE OPERATOR

Based on the motivation above, this part proposes a new search operator based on symmetric sine and cosine function.

The properties of the symmetric sine cosine operator are opposite to the original. Two sets of piecewise functions as shown in (26) and (27). The following two sets of formulas have the same probability of being chosen and are considered as new symmetric sine and cosine operator.

$$x_i^{t+1} = \begin{cases} x_i^t + r1(-\sin(r2) + 1)\left|r3P_i^t - x_i^t\right|, \\ if \quad 0 < r2 \le \pi \\ x_i^t + r1(-\sin(r2) - 1)\left|r3P_i^t - x_i^t\right|, \\ if \quad \pi < r2 \le 2\pi \end{cases} \tag{26}$$

$$x_i^{t+1} = \begin{cases} x_i^t + r1(-\cos(r2)+1)\left|r3P_i^t - x_i^t\right|, \\ if \quad r2 < \dfrac{\pi}{2} \ or \quad r2 > \dfrac{3\pi}{2} \\ x_i^t + r1(-\cos(r2)-1)\left|r3P_i^t - x_i^t\right|, \\ if \quad \dfrac{\pi}{2} \le r2 \le \dfrac{3\pi}{2} \end{cases} \tag{27}$$

Among them, "r1, r2, and r3" have the same meaning as "r_1, r_2, and r_3" in SCA.

To combine the new symmetric sine cosine operator with the original sine cosine operator, there are two solutions: the first one is that the probability of choosing the new operator and the original operator is fixed. In order to be fair, the probability of selecting each strategy is set to 0.5. The other is to select strategies adaptively of according to the success rate of offspring. The latter method is used in the paper, and part D below gives the specific steps of the adaptive probabilistic selection mechanism.

D. ADAPTIVE PROBABILISTIC SELECTION MECHANISM

To better select appropriate mutation operator for population, an adaptive probabilistic selection mechanism is introduced to select appropriate strategies. This adaptive selection strategy has been used in many papers [20], [21], and has been proved effective.

$$p^{t+1} = \phi \cdot p^t + (1-\phi) sr^t \tag{28}$$

The experimental results, "φ" with a value of 0.95, showed in [20] are directly used in this paper. For the sake of fairness, the initial selection probability is set to $p^0 = 0.5$, which means the probability of using the original SCA operator is the same as that of the new operator with the added symmetric piecewise function. "sr^t" represents the probability of successfully updating the offspring of the new operator, which is given by equation (29).

$$sr^t = \frac{S\_new^{t-1}}{(S\_new^{t-1} + 2 \times S\_old^{t-1})} \tag{29}$$

"S_new^{t-1}" and "S_old^{t-1}" represent the number of children successfully updated by the new operator and the original operator respectively. To balance the selection probability, the number of new strategies is twice as many as the number of original strategies, so "S_oldt-1" is multiplied by 2. In this way, In this way, the selected strategy is not limited to the new strategy, but also has the probability to select the original strategy. The adaptive probabilistic selection framework is given by Algorithm 2.

E. BOUNDARY HANDLING MECHANISM

After some empirical tests, it was found that the convergence speed of the operator is too slow in the process of implementation.

This phenomenon may occur due to the problem that the generated new solution may exceed the pre-defined boundary constraints. Therefore, some measures must be applied to the generated new solution to bring it back into the search space. Moreover, the boundary processing mechanism [22] proposed by other researchers, is added here to deal with the transboundary problem of a position simply and efficiently. The mathematical expression is as follows:

$$x_{(i,j)}^{t+1} = \begin{cases} \min(lu(2,j), 2lu(1,j) - x_{(i,j)}^{t+1}), \\ if \quad x_{(i,j)}^{t+1} < lu(1,j) \\ \max(lu(1,j), 2lu(2,j) - x_{(i,j)}^{t+1}), \\ if \quad x_{(i,j)}^{t+1} < lu(2,j) \end{cases} \tag{30}$$

where "lu(1, j)" and "lu(2, j)" represents the lower and upper bounds of functions, respectively. By using this transboundary processing mechanism, the search range of the solution can be effectively limited to the pre-defined search space, and the convergence efficiency can be improved.

TABLE 2. Adaptive strategy selection framework.

Algorithm 2: Adaptive Strategy Selection Framework

```
Initialization parameter p0, φ, r5. (r5 is a random value between [0,1])
while t ≤ T
 if r1 < 1
  for i = 1...N (N is population size)
   if p ≤ r5
    Update operations with a set of policies (1) in SCA.
    S_old^t = S_old^{t-1} + 1.
   Else
    Update operation with the two new strategies (26) and (27) proposed.
     S_new^t = S_new^{t-1} + 1.
   End if
  End for
  Update probability p with the formula (28) and (29)
 End if
End while
```

F. QUADRATIC INTERPOLATION OPERATOR

Empirical evidence also showed that although the proposed operator based on sine and cosine is improved, the accuracy of the algorithm is still insufficient.

In order to improve the overall quality of individuals in the population and exploitation ability of operators, the quadratic interpolation method [23] is introduced. Researchers have proved that some algorithms [24]–[26] which used quadratic interpolation can further improve the population quality and increase the convergence speed.

Quadratic interpolation is used to produce a new individual based on three different individuals, the best individual "$pbest^t$" and two randomly chosen members of the current population, respectively, as "$x_{(id1,j)}^t$" and "$x_{(id2,j)}^t$", with t being the current generation number. The mathematical expression of the overall generation rule is as follows:

$$w1 = \left(pbest^t - x_{(id2,j)}^t\right)^2 f\left(x_{id1}^t\right) \tag{31}$$

$$w2 = \left(x_{(id1,j)}^t - pbest^t\right)^2 f\left(x_{id2}^t\right) \tag{32}$$

$$w3 = \left(x_{(id2,j)}^t - x_{(id1,j)}^t\right)^2 f\left(pbest^t\right) \tag{33}$$

$$u1 = \left(pbest^t - x_{(id2,j)}^t\right) f\left(x_{id1}^t\right) \tag{34}$$

$$u2 = \left(x^t_{(id1,j)} - pbest^t\right) f\left(x^t_{id2}\right) \tag{35}$$

$$u3 = \left(x^t_{(id2,j)} - x^t_{(id1,j)}\right) f\left(pbest^t\right) \tag{36}$$

$$x^t_{(qi,j)} = \frac{w1 + w2 + w3}{u1 + u2 + u3} \tag{37}$$

where "f(x_{id1})", "f(x_{id2})", and "f(pbest)" are the objective function values of "x_{id1}", "x_{id2}", and "pbest", respectively. "$x^t_{(qi,j)}$" is the position of the individual in the t generation and j dimension obtained by using the quadratic interpolation operator.

It is easy to see that a new individual can be generated from (37). According to the objective function value, the newly generated individual is compared with the worst individual in the population, and the best one between the two survives is retained for the next generation. In this way, the diversity of the population can be preserved, and the quality of the individuals in the population can be improved, which is an excellent way to balance diversity and convergence.

The pseudo-code of quadratic interpolation is as Table 3:

TABLE 3. Quadratic interpolation pseudo-code.

Algorithm 3: Quadratic Interpolation Pseudo-code
Input: A population with N individuals
1. Find the maximum objective function value f_{max} and the corresponding individual x_{worst}.
2. Two individuals x_{id1} and x_{id2} are randomly selected from the current population.
3. Obtain x_{qi} by (37)
4. The fitness value fit_qi was obtained by evaluating x_{qi}.
5. Compare the size of the function value fit_qi with f_{max}, if fit_q is smaller, replace x_{worst} with x_{qi}.
Output: population after the worst individual is replaced

G. GUSSIAN PERTURBATION

The common characteristic of the population-based algorithm is easy to get into local optimization. As a population optimization algorithm, SCA naturally has the same problem.

It is known through analysis that the optimal global individual obtained at present may fall into local optimization. Therefore, the Gaussian perturbation [27] operator is used to mutate the optimal global individual to increase the chance of jumping out of local optima, and to increase the accuracy of the algorithm. When the threshold value set is reached (the threshold value set is 5 here), the global optimal position "pbest" is not updated, indicating that it may fall into local optimization. At this time, formula (38) is executed, as shown below.

$$pbest = pbest + Guassian() \tag{38}$$

where pbest is the location of the globally optimal individual, that is, a d-dimensional vector, and Gaussian () is the random vector generated by Gaussian distribution [28], [29].

H. FRAMEWORK OF SSCA-APS

The overall framework of the symmetric sine cosine algorithm for adaptive probability selection (SSCA-APS), is described in Algorithm 4. The proposed algorithm divides the evolution process into early stage and late stage and selects SCA or symmetric SCA strategy in the late stage by means of adaptive probabilistic selection. And Gaussian perturbation, quadratic interpolation and boundary processing are used to improve the performance of the algorithm.

TABLE 4. SSCA-APS pseudo-code.

Algorithm 4: SSCA-APS Pseudo-code

Input: Population size **N**, dimension **D** and boundary **lu** of the problem, the maximum number of fitness evaluation **FESMAX.**

Initalization: Randomly initialize population **x**, set current generation t=0, max generation **T**, initializes other parameters.

1. The population was evaluated according to the objective function value.
2. Find the minimum objective function value f_{min} and the corresponding individual pbest.
3. Find the maximum objective function value f_{max} and the corresponding individual x_{worst}.
4. Set the current evaluation times as FES = N.

Main loop:

5. **while** FES $\leq$ FESMAX
6. According to (2) calculate the value of r1.
7. $S\_new^{t-1} = 0$, and $S\_old^{t-1} = 0$.
8. t = t + 1.

In the early stage:

9. **If** r1 > 1
10. flag = 0;
11. Update individual location according to **Algorithm 1**.
12. **If** the current individual was successfully updated
13. $flag^t = flag^{t-1} + 1$.
14. **End if**
15. **If** $flag^t = 0$.
16. Gaussian perturbation is carried out on pbest with the (38).
17. **End if**
18. **If** mod(t, 5) = 0
19. flag = 0.
20. **End if**
21. **End if**

In the late stage:

22. **If** r1 $\leq$ 1
23. **for** i = 1…N, calculate the value of r2, r3, r4, and r5.
24. **If** p $\leq$ r5
25. Update individual location according to **Algorithm 1**.
26. **If** the current individual was successfully updated
27. $S\_old^t = S\_old^{t-1} + 1$.
28. **End if**
29. Conduct boundary check with (30).
30. **If** $S\_old^t = 0$
31. Gaussian perturbation is carried out on pbest with the (38).
32. **End if**
33. **Else**
34. (26) and (27) were used to update the population.
35. Eq.30 was used to conduct boundary checks and treatment.
36. Evaluate the individual $newx_i$ and compare it with x_i according to the fitness value.
37. **If** it gets better than before, replace the old one.
38. $S\_new^t = S\_new^{t-1} + 1$.
39. **End if**
40. **End if**
41. **End for**
42. **If** mod(t, 5) = 0
43. $S\_old^t = 0$, $S\_new^t = 0$.
44. **End if**
45. **End if**
46. Update the population by quadratic interpolation in **Algorithm 3**.
47. Update global optimal individual pbest and fitness value.
48. FES = FES + N.
49. **End while**

Output: Take the current best solution as the optimal solution output.

Algorithm 4 shows the pseudo-code for SSCA-APS, starting with the size of the population N, the dimension D of the decision variable, the boundary lu of the problem, and the maximum number of fitness evaluations $FESMAX$. In this paper, the objective function value is used as the criterion of fitness evaluation. First, it initializes the population and other parameters and evaluate the population and find the best and worst fitness individuals in the population. Then, the main loop is then divided into two stages. In the early stage, the sine and cosine operators in the original SCA are used, and an individual is detected every five generations for successful update. If not, Gaussian perturbation is performed on the globally optimal individual. In the late stage, each generation refreshes two counters: they are used to detect whether the SCA operator (i.e. S_old^t) and the symmetric operator (i.e. S_new^t) can contribute to the population renewal. If not, it still perturbs the globally optimal individual $pbest$. Finally, the obtained solution is output as the optimal solution.

IV. NUMERICAL EXPERIMENTS AND ANALYSIS

This part uses the form of tables and graphs to verify the algorithm proposed in this paper and give a proper explanation.

A. EXPERIMENTAL SETTINGS

The 23 benchmark functions used in the experiment are briefly described as follows: functions F1-F13 are high-dimensional problems, among them, F1-F5 are unimodal, F6 is the step function that has one minimum value and is discontinuous. F7 is a noisy quartic function, and F8-F13 are multimodal functions. Function F14-F23 are low-dimensional and multimodal functions, and they only have local minima. The final results of multimodal functions are important since they affect the ability of the algorithm to jump out of the local optimum and to be better approached toward the global optimum.

The proposed algorithm is compared with classical SCA and its several variants, which are SOA, ISCA, SCADE, NMSCA. The performances of these algorithms on 23 test functions are given in Table 5 and Table 6. Their population sizes are uniformly set to 100 ($N = 100$), and the dimension D of problem are 30(F1-F13), 2(F14, F16-F18), 4(F15, F21-F23), 3(F19), and 6(F20), respectively. The maximum number of fitness evaluation FESMAX are set $D \times 10000$, and the generation $T = \lfloor FESMAX/N \rfloor$($\lfloor A \rfloor$ is the largest integer no greater than A).

B. WILCOXON SIGN RANK TESTS

In order to prove the effectiveness of the proposed algorithm, the experimental results of each algorithm were compared statistically with or without significant differences. The proposed SSCA-APS is validated against other SCA variants. Wilcoxon sign rank tests was used here, and the results were shown in Table 5.

To identify SSCA-APS with SCA and other variants algorithms on 23 benchmark problems [30], similar statistical tools are employed. Due to most values of the calculated p-values is smaller than 0.05, the obtained ranks are statistically reliable while the proposed strategies are used. In other words, there are some improvements achieved by employing proposed techniques, the improvements are significant from the statistical point of view. In order to more intuitively see whether there is any difference between the compared algorithms or not, the last column indicates "1" when there is a significant difference and "0" otherwise. And "NaN" represents positive infinity.

TABLE 5. Wilcoxon sign rank tests.

Functions	SOA (p)	h	ISCA (p)	h	SCADE (p)	h	NMSCA (p)	h
F1	2.66E-04	1	2.66E-04	1	2.66E-04	1	NaN	0
F2	2.66E-04	1	2.66E-04	1	2.66E-04	1	NaN	0
F3	2.66E-04	1	2.66E-04	1	2.66E-04	1	NaN	0
F4	2.66E-04	1	2.66E-04	1	2.66E-04	1	NaN	0
F5	2.66E-04	1	2.66E-04	1	2.50E-13	1	2.66E-04	1
F6	2.70E-13	1	2.70E-13	1	2.70E-13	1	2.70E-13	1
F7	2.66E-04	1	2.66E-04	1	2.66E-04	1	2.66E-04	1
F8	2.66E-04	1	2.66E-04	1	2.50E-13	1	2.66E-04	1
F9	2.67E-04	1	2.70E-13	1	2.70E-13	1	2.70E-13	1
F10	2.66E-04	1	5.81E-08	1	2.50E-13	1	2.50E-13	1
F11	2.66E-04	1	NaN	0	NaN	0	NaN	0
F12	2.66E-04	1	2.66E-04	1	2.50E-13	1	2.66E-04	1
F13	2.67E-04	1	2.67E-04	1	2.70E-13	1	2.67E-04	1
F14	2.66E-04	1	2.66E-04	1	2.66E-04	1	2.66E-04	1
F15	2.67E-04	1	2.67E-04	1	2.68E-02	1	2.67E-04	1
F16	2.66E-04	1	2.66E-04	1	2.66E-04	1	2.66E-04	1
F17	2.66E-04	1	2.66E-04	1	2.66E-04	1	2.66E-04	1
F18	2.67E-04	1	2.67E-04	1	2.67E-04	1	2.67E-04	1
F19	2.66E-04	1	2.66E-04	1	2.65E-04	1	2.66E-04	1
F20	9.89E-04	1	1.84E-02	1	1.66E-04	1	2.44E-03	1
F21	2.66E-04	1	2.66E-04	1	1.40E-04	1	2.66E-04	1
F22	2.66E-04	1	2.66E-04	1	7.87E-05	1	2.66E-04	1
F23	2.66E-04	1	2.66E-04	1	1.64E-04	1	2.66E-04	1

As can be seen from the Table 5, SSCA-APS differs significantly from SOA algorithm in all benchmark functions. And in addition to the benchmark function F1, the proposed algorithm and ISCA algorithm show significant differences in the remaining benchmark functions. Compared with NMSCA, the proposed algorithm showed significant differences in other test problems except F1-F4 and F11. It indicates that the proposed algorithm has almost no difference from NMSCA in high-dimensional and single-mode problems, that is, the results of F1-F4 and F11 benchmark problems are not comparable but can be compared for most other test problems. From what has been discussed above, the result of rank-sum test is that the proposed SSCA-APS is significantly different from other SCA variants, so the proposed algorithm has passed the validity test, and the comparative results are reliable. Therefore, error accuracy and convergence analysis can be carried out.

C. COMPARISON OF SSCA-APS RESULTS WITH OTHER ALGORITHMS

In the Table 6 and Table 7, "NaN" represents positive infinity, and it can be seen from Table 6 that, except for F1, F2, and F11, the remaining 20 benchmark functions all show that the proposed SSCA-APS algorithm has significant changes compared with the results of SCA algorithm. 50 independent

TABLE 6. Comparison table of experimental results.

Functions	SCA		Standard Deviation	SOA		Standard Deviation	SSCA-APS	Standard Deviation	P value	Difference
F1	**0.00E+00**	~	**0.00E+00**	3.32E-04	-	8.05E-05	**0.00E+00**	**0.00E+00**	2.58E-01	0
F2	3.70E-315	~	**0.00E+00**	1.10E-01	-	2.35E-02	**0.00E+00**	**0.00E+00**	2.66E-04	0
F3	2.22E+04	-	1.52E+04	**2.32E-02**	+	**8.34E-03**	2.04E+04	1.45E+04	3.51E-04	1
F4	8.23E-19	-	5.82E-18	1.93E-02	-	4.36E-03	**9.65E-236**	**0.00E+00**	2.66E-04	1
F5	2.76E+01	-	**3.72E-01**	6.52E+01	-	9.22E+01	**1.42E+01**	1.28E+01	2.70E-13	1
F6	**0.00E+00**	~	**0.00E+00**	**0.00E+00**	~	**0.00E+00**	**0.00E+00**	**0.00E+00**	2.66E-04	1
F7	1.56E-02	-	1.34E-02	3.74E-02	-	1.42E-02	**1.19E-02**	**9.26E-03**	2.66E-04	1
F8	**5.47E-03**	+	**5.75E-03**	5.13E+03	+	6.72E+02	-2.20E+06	8.66E+06	2.70E-13	1
F9	**0.00E+00**	~	**0.00E+00**	5.89E+01	-	1.18E+01	**0.00E+00**	**0.00E+00**	2.69E-13	1
F10	**8.88E-16**	~	**0.00E+00**	1.38E-02	-	1.81E-03	**8.88E-16**	**0.00E+00**	NaN	1
F11	**0.00E+00**	~	**0.00E+00**	9.74E-03	-	9.92E-03	**0.00E+00**	**0.00E+00**	2.66E-04	0
F12	5.57E-03	-	2.35E-03	4.16E-03	-	2.05E-02	**1.41E-03**	**1.02E-03**	2.67E-04	1
F13	1.01E-01	-	3.90E-02	**2.33E-03**	+	**4.91E-03**	4.25E-02	2.98E-02	2.67E-04	1
F14	2.41E-04	-	1.68E-03	6.60E+00	-	6.42E+00	**3.92E-06**	**5.32E-07**	2.67E-04	1
F15	2.12E-04	-	**1.37E-04**	1.67E-03	-	4.71E-03	**1.74E-04**	1.40E-04	2.66E-04	1
F16	**6.46E-07**	+	**5.63E-07**	6.50E-06	-	7.24E-06	2.87E-06	5.09E-06	2.66E-04	1
F17	3.77E-05	+	8.26E-05	**3.33E-06**	-	**2.83E-06**	2.71E-05	5.72E-05	2.67E-04	1
F18	**6.59E-05**	+	**1.03E-04**	3.97E-04	-	3.96E-04	8.51E-05	1.92E-04	2.67E-04	1
F19	**8.16E-05**	+	**7.26E-05**	1.08E-04	+	1.03E-04	1.96E-04	2.48E-04	3.93E-03	1
F20	6.63E-02	-	6.34E-02	**2.94E-02**	+	**5.25E-02**	3.81E-02	5.82E-02	2.66E-04	1
F21	3.88E+00	-	2.19E+00	4.08E-01	-	1.39E+00	**1.13E-01**	**7.20E-01**	2.66E-04	1
F22	3.09E+00	-	2.64E+00	2.13E-01	-	1.05E+00	**1.20E-01**	**7.51E-01**	2.66E-04	1
F23	3.37E+00	-	2.64E+00	3.25E-01	-	1.30E+00	**1.20E-01**	**7.64E-01**	2.58E-01	1

TABLE 7. Comparison table of experimental results.

Func-tions	ISCA		Standard Deviation	SCADE		Standard Deviation	NMSCA		Standard Deviation	SSCA-APS	Standard Deviation
F1	5.99E-02	-	4.23E-01	5.72E-136	-	3.95E-135	**0.00E+00**	~	**0.00E+00**	**0.00E+00**	**0.00E+00**
F2	8.28E-99	-	5.86E-98	8.64E-71	-	2.74E-70	**0.00E+00**	~	**0.00E+00**	**0.00E+00**	**0.00E+00**
F3	7.05E+03	+	3.83E+03	1.07E-112	+	5.68E-112	**0.00E+00**	+	**0.00E+00**	2.04E+04	1.45E+04
F4	2.78E-01	-	1.54E+00	3.31E-65	-	1.62E-64	**0.00E+00**	+	**0.00E+00**	9.65E-236	**0.00E+00**
F5	2.86E+01	-	**3.22E-01**	2.90E+01	-	**0.00E+00**	2.79E+01	-	4.62E-01	1.42E+01	1.28E+01
F6	**0.00E+00**	~	**0.00E+00**	**0.00E+00**	~	**0.00E+00**	**0.00E+00**	~	**0.00E+00**	**0.00E+00**	**0.00E+00**
F7	**9.07E-03**	+	1.03E-02	1.05E-02	+	1.06E-02	1.16E-02	+	1.01E-02	1.19E-02	**9.26E-03**
F8	1.11E+03	+	1.37E+03	1.26E+04	+	**1.84E-12**	**1.30E-01**	+	1.01E-01	-2.20E+06	8.66E+06
F9	**0.00E+00**	~	**0.00E+00**	**0.00E+00**	~	**0.00E+00**	**0.00E+00**	~	**0.00E+00**	**0.00E+00**	**0.00E+00**
F10	5.69E-07	-	3.65E-06	**8.88E-16**	~	**0.00E+00**	**8.88E-16**	~	**0.00E+00**	**8.88E-16**	**0.00E+00**
F11	**0.00E+00**	~	**0.00E+00**	**0.00E+00**	~	**0.00E+00**	**0.00E+00**	~	**0.00E+00**	**0.00E+00**	**0.00E+00**
F12	2.55E-01	-	2.98E-01	1.67E+00	-	**4.49E-16**	2.53E-03	-	1.13E-03	1.41E-03	**1.02E-03**
F13	2.06E+00	-	2.73E+00	3.00E+00	-	**0.00E+00**	4.99E-02	-	2.22E-02	4.25E-02	2.98E-02
F14	1.76E-01	-	7.16E-01	1.06E+01	-	3.11E+00	2.62E-05	-	6.98E-05	**3.92E-06**	**5.32E-07**
F15	4.82E-04	-	1.25E-03	1.45E-01	-	2.09E-02	1.93E-04	-	1.14E-04	**1.74E-04**	**1.40E-04**
F16	**2.99E-07**	+	**7.28E-07**	7.67E-01	-	4.45E-01	6.64E-04	-	5.76E-04	2.87E-06	5.09E-06
F17	1.51E-03	-	9.35E-03	5.27E+01	-	9.92E+00	3.87E-04	-	5.95E-04	**2.71E-05**	**5.72E-05**
F18	4.80E-04	-	3.39E-03	4.42E+02	-	2.51E+02	8.09E-03	-	1.08E-02	**8.51E-05**	**1.92E-04**
F19	1.60E-02	-	1.55E-02	3.79E+00	-	**2.27E-04**	1.96E-03	-	2.37E-03	**1.96E-04**	2.48E-04
F20	1.49E-01	-	1.32E-01	3.32E+00	-	**2.35E-15**	5.55E-02	-	6.31E-02	**3.81E-02**	5.82E-02
F21	4.32E+00	-	1.75E+00	9.88E+00	-	1.74E-06	1.69E-01	-	**1.37E-01**	**1.13E-01**	7.20E-01
F22	4.47E+00	-	1.84E+00	1.01E+01	-	6.80E-03	1.80E-01	-	**1.72E-01**	**1.20E-01**	7.51E-01
F23	4.59E+00	-	1.98E+00	1.02E+01	-	7.23E-06	2.12E-01	-	**2.46E-01**	**1.20E-01**	7.64E-01

runs were conducted respectively, and the mean error and standard deviation are shown in Table 6 and Table 7. The "+" means that the result is better than the proposed SSCA-APS algorithm, "−" means that the result is worse, and "∼" means that the result is relatively close. The highlighted sections in bold black represent the best performing algorithms for each of the benchmark problems. The results show that SSCA-APS algorithm has better performance and higher accuracy in most problems. Compared with SCA and several other improved SCA, it is more competitive in low-dimensional (F14-F23) problems and performs better in high-dimensional (F1-F7) problems. In the high-dimensional multi-peak problem (F12, F13), it has obvious advantages over SCA, but for the test problem (F8) where the optimal value is not at the origin, SSCA-APS is outperformed.

As has been shown in the Table 7, SSCA-APS has the largest number of black bolded parts for 23 benchmark problem, and the experimental results show that, overall, the proposed SSCA-APS algorithm has some competitive advantages compared with other algorithms.

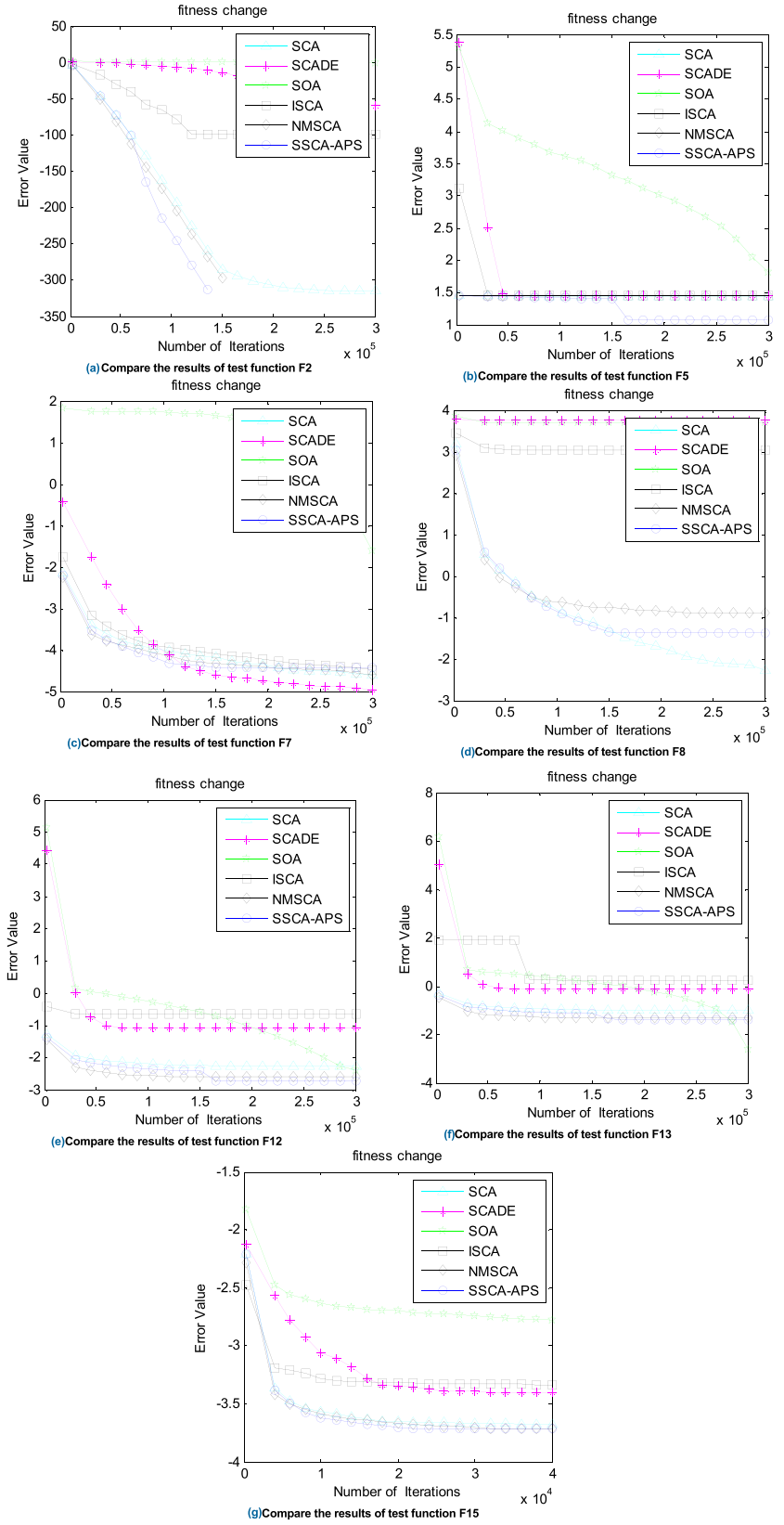


FIGURE 9. (a). Compare the results of test function F2. (b). Compare the results of test function F5. (c). Compare the results of test function F7. (d). Compare the results of test function F8. (e). Compare the results of test function F12. (f). Compare the results of test function F13. (g). Compare the results of test function F15.

These algorithms are ranked according to error mean and standard deviation to see the advantages and disadvantages of each algorithm clearly and intuitively, and the results are shown in the following Table 8.

TABLE 8. Ranking.

Algorithm	Error mean rank	Standard deviation rank	Rank
SCA	2	2	2
SOA	3	5	4
ISCA	3	5	4
SCADE	4	4	4
NMSCA	2	3	2.5
SSCA-APS	1	1	1

The smaller the error, the higher the accuracy, the smaller the standard deviation and the stronger the robustness of the algorithm, that is, the more stable the performance of the operator. Therefore, it can be seen from Table 8 that the performance of the proposed algorithm is the best in both accuracy and stability.

To see the operation effect of the algorithm more intuitively, the function error values of these algorithms running independently for 50 times were compared, several representative results as shown in Figure 9 (a)-(g). And it clearly shows the performance of the proposed algorithm and SCA and its variants in terms of convergence speed, fitness change, and accuracy.

In Figure 9, the faster the error curve drops mean the faster the algorithm converges, and the lower the curve mean the higher the accuracy. Due to the limitation of aesthetics and layout, only the convergence graph on the benchmark problem F2, F5, F7, F8, F12, F13, and F15 was selected to represent the display. Blue triangle, red star, green star, black square, black diamond and blue circle respectively represent SCA, SCADE, SOA, ISCA, NMSCA and SSCA-APS algorithm. As can be seen from Figure 9, the proposed algorithm significantly improves the optimization accuracy in low-dimensional problems. In general, the proposed algorithm is improved in both accuracy and convergence.

D. ALGORITHM COMPLEXITY ANALYSIS AND COMPARISON

In order to objectively compare the quality of the algorithms, here we analyze and compare the complexity of the algorithms involved in the experiments. The complexity of the algorithm mainly includes two parts: time complexity and space complexity. Since none of the algorithms in the experiment introduces external files, that is, they operate in populations with the same number of individuals, and the spatial complexity of each algorithm is approximately equal. The following mainly analyzes the time complexity of the algorithm.

For the SCA algorithm, its time complexity is mainly related to the number of fitness evaluations of individuals in the population. The population size is N, the problem dimension is D, and the number of cycles is T. Then, the algorithm has a time complexity of $O(N \cdot D \cdot T)$. In SCADE, three additional individuals need to be randomly selected and evaluated separately in each iteration, so its time complexity is $O\ (4N{\cdot}D \cdot T)$. The SOA, ISCA, and NMSCA algorithms only change the equations of the SCA search strategy, so the time complexity is the same as SCA, which is $O(N \cdot D \cdot T)$. For the proposed algorithm SSCA-APS, because it evolves in two stages, and in order to adaptively adjust the probability, the fitness evaluation and probability calculation are performed at each iteration, so the time complexity is $O(N^2 \cdot D \cdot T)$. It can be seen that SSCA-APS algorithm is more complicated than the original SCA and its variants. However, from the experimental results, the increase of algorithm complexity contributes to the improvement of algorithm performance.

V. CONCLUSION

In the original SCA algorithm, the standard sine cosine functions were used as the probability functions to generate mutation step sizes. In general, the standard sine cosine functions have greater probability of generating larger strides than small ones, which is not conducive to the convergence in the exploitation stage. In response to this problem, the symmetrical sine cosine operators were proposed and applied in the late stage of evolution. To balance the convergence and diversity of the algorithm, the adaptive probabilistic selection mechanism was added, which was used to dynamically select the standard sine cosine operators or the proposed symmetrical sine cosine operators.

With the increase number of iterations, the mutation strength of the SCA algorithm gradually decreases, so it is easy for that solution to fall into local optimal. The introduction of Gaussian mutation can overcome this problem. To further improve the overall quality of the population, quadratic interpolation is introduced to replace the worst individual in the population. In addition, a simple and efficient boundary handling mechanism is added so that the search can be conducted within the predefined search space. The results show that this method can accelerate the approach speed towards the ideal point in the search space.

In simulations, by analyzing experiments on SCA and some state-of-the-art variants in 23 benchmark functions, it was concluded that the proposed SSCA-APS algorithm on high-dimensional problems compared with the SCA and other several improved SCA, has strong competitiveness, and on the low dimensional problem, it behaved better than the SCA. Overall, the experimental results show that the proposed SSCA-APS algorithm has some competitive advantages compared with other algorithms.

VI. FUTURE WORK

Although the mathematical proof and numerical experiments have been shown that the improved sine cosine algorithm of this paper has certain competitiveness compared with SCA and its variants, further research on practical problems is still needed, such as the optimal sitting and sizing of photovoltaic based distributed generator [32], [33], distribution

static synchronous compensator [34], optimization of airfoil design [35], and other applications [36].

In addition, the algorithm proposed in this paper divides the search into two phases based on an iterative process, which may have certain risks. So how to properly measure the evolutionary stage has become the direction of future research, such as judging the evolutionary stage based on the diversity of the population or using the distance between individuals in the population as the evaluation criteria.

Moreover, the accuracy of the algorithm is improved, but at the same time the computational complexity is also increased. Future research can work on how to improve the efficiency of the algorithm and reduce the complexity of the algorithm.

REFERENCES

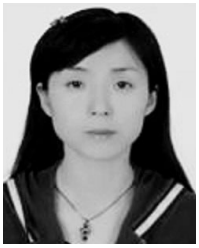
[1] S. Mirjalili, "SCA: A sine cosine algorithm for solving optimization problems," *Knowl.-Based Syst.*, vol. 96, pp. 120–133, Mar. 2016.
[2] H. Nenavath and R. K. Jatoth, "Hybridizing sine cosine algorithm with differential evolution for global optimization and object tracking," *Appl. Soft Comput.*, vol. 62, pp. 1019–1043, Jan. 2018.
[3] N. Singh and S. Singh, "A novel hybrid GWO-SCA approach for optimization problems," *Eng. Sci. Technol., Int. J.*, vol. 20, no. 6, pp. 1586–1601, Dec. 2017.
[4] M. Abd Elaziz, D. Oliva, and S. Xiong, "An improved Opposition-Based Sine Cosine Algorithm for global optimization," *Expert Syst. Appl.*, vol. 90, pp. 484–500, Dec. 2017.
[5] M. Meshkat and M. Parhizgar, "A novel weighted update position mechanism to improve the performance of sine cosine algorithm," in *Proc. 5th Iranian Joint Congr. Fuzzy Intell. Syst. (CFIS)*, Mar. 2017, pp. 166–171.
[6] K. Z. Zamli, F. Din, B. S. Ahmed, and M. Bures, "A hybrid Q-learning sine-cosine-based strategy for addressing the combinatorial test suite minimization problem," *PLoS ONE*, vol. 13, no. 5, May 2018, Art. no. e0195675.
[7] W. Long, T. Wu, X. Liang, and S. Xu, "Solving high-dimensional global optimization problems using an improved sine cosine algorithm," *Expert Syst. Appl.*, vol. 123, pp. 108–126, Jun. 2019.
[8] M. Meshkat and M. Parhizgar, "Sine optimization algorithm (SOA): A novel optimization algorithm by change update position strategy of search agent in sine cosine algorithm," in *Proc. 3rd Iranian Conf. Intell. Syst. Signal Process. (ICSPIS)*, Dec. 2017, pp. 11–16.
[9] C. Qu, Z. Zeng, J. Dai, Z. Yi, and W. He, "A modified sine-cosine algorithm based on neighborhood search and greedy levy mutation," *Comput. Intell. Neurosci.*, vol. 2018, pp. 1–19, Jul. 2018.
[10] S. Gholizadeh and R. Sojoudizadeh, "Modified sine-cosine algorithm for sizing optimization of truss structures with discrete design variables," *Int. J. Optim. Civil Eng. Int. J. Optim. Civil Eng.*,vol. 9, no. 2, pp. 195–212, 2019.
[11] S. Gupta and K. Deep, "A hybrid self-adaptive sine cosine algorithm with opposition based learning," *Expert Syst. Appl.*, vol. 119, pp. 210–230, Apr. 2019.
[12] S. Gupta and K. Deep, "Improved sine cosine algorithm with crossover scheme for global optimization," *Knowl.-Based Syst.*, vol. 165, pp. 374–406, Feb. 2019.
[13] S. Dunia Tahir and S. Ramzy Ali, "Chaotic sin-cosine optimization algorithms," *Int. J. Soft Comput.*, vol. 133, pp. 108–122, Aug. 2018.
[14] Y. Zhou, R. Wang, and Q. Luo, "Elite opposition-based flower pollination algorithm," *Neuro Comput.*, vol. 188, pp. 294–310, May 2016.
[15] S. Li, H. Fang, and X. Liu, "Parameter optimization of support vector regression based on sine cosine algorithm," *Expert Syst. Appl.*, vol. 91, pp. 63–77, Jan. 2018.
[16] S. Abdel-Fatah, M. Ebeed, and S. Kamel, "Optimal reactive power dispatch using modified sine cosine algorithm," in *Proc. Int. Conf. Innov. Trends Comput. Eng. (ITCE)*, Feb. 2019, pp. 510–514.
[17] X. Song, M. Zhao, Q. Yan, and S. Xing, "A high-efficiency adaptive artificial bee colony algorithm using two strategies for continuous optimization," *Swarm Evol. Comput.*, vol. 50, Nov. 2019, Art. no. 100549.
[18] B. Xu, X. Chen, and L. Tao, "Differential evolution with adaptive trial vector generation strategy and cluster-replacement-based feasibility rule for constrained optimization," *Inf. Sci.*, vol. 435, pp. 240–262, Apr. 2018.
[19] X. Chen, C. Mei, B. Xu, K. Yu, and X. Huang, "Quadratic interpolation based teaching-learning-based optimization for chemical dynamic system optimization," *Knowl.-Based Syst.*, vol. 145, pp. 250–263, Apr. 2018.
[20] D. Tian, X. Zhao, and Z. Shi, "Chaotic particle swarm optimization with sigmoid-based acceleration coefficients for numerical function optimization," *Swarm Evol. Comput.*, vol. 51, Dec. 2019, Art. no. 100573.
[21] X. Yao, Y. Liu, and G. Lin, "Evolutionary programming made faster," *IEEE Trans. Evol. Comput.*, vol. 3, no. 2, pp. 82–102, Jul. 1999.
[22] R. Storn and K. V. Price, "Differential Evolution–A Simple and Effcient Heuristic for Global Optimization over Continuous Spaces," *J. Global Optim.*, vol. 11, pp. 341–359, Dec. 1997.
[23] J. Derrac, S. García, D. Molina, and F. Herrera, "A practical tutorial on the use of nonparametric statistical tests as a methodology for comparing evolutionary and swarm intelligence algorithms," *Swarm Evol. Comput.*, vol. 1, no. 1, pp. 3–18, Mar. 2011.
[24] S. García, A. Fernández,, J. Luengo, and F. Herrera, "Advanced nonparametric tests for multiple comparisons in the design of experiments in computational intelligence and data mining: Experimental analysis of power," *Inf. Sci.*, vol. 180, no. 10, pp. 2044–2064, May 2010.
[25] A. Zareian, S. Azadi, and R. Kazemi, "Estimation of road friction coefficient using extended Kalman filter," *Recursive Least Square, Neural Netw. Proc. Inst. Mech. Eng.*, vol. 230, no. 1, pp. 52–68, 2016.
[26] Y. Sun, T. Yang, and Z. Liu, "A whale optimization algorithm based on quadratic interpolation for high-dimensional global optimization problems," *Appl. Soft Comput.*, vol. 85, Dec. 2019, Art. no. 105744.
[27] A. Gowrisankar and M. G. P. Prasad, "Riemann–Liouville calculus on quadratic fractal interpolation function with variable scaling factors," *J. Anal.*, vol. 27, no. 2, pp. 347–363, Jun. 2019.
[28] Y. Feng, G.-G. Wang, J. Dong, and L. Wang, "Opposition-based learning monarch butterfly optimization with Gaussian perturbation for large-scale 0-1 knapsack problem," *Comput. Elect. Eng.*, vol. 67, pp. 454–468, Apr. 2018.
[29] T. Claeys, A. B. J. Kuijlaars, K. Liechty, and D. Wang, "Propagation of singular behavior for Gaussian perturbations of random matrices," *Commun. Math. Phys.*, vol. 362, no. 1, pp. 1–54, Aug. 2018.
[30] S.-Y. Lee and M. Yang, "Discontinuity in the asymptotic behavior of planar orthogonal polynomials under a perturbation of the Gaussian weight," *Commun. Math. Phys.*, vol. 355, no. 1, pp. 303–338, Oct. 2017.
[31] C. A. Penfold, S. Anastasiya, J. E. Reid, H. Yun, W. Lorenz, G. Zoubin, G. Murray, and S. M. Azim, "Branch-recombinant Gaussian processes for analysis of perturbations in biological time series," *Bioinf.*, vol. 34, no. 17, pp. i1005–i1013, 2018.
[32] R. Sirjani and A. Rezaee Jordehi, "Optimal placement and sizing of distribution static compensator (D-STATCOM) in electric distribution networks: A review," *Renew. Sustain. Energy Rev.*, vol. 77, pp. 688–694, Sep. 2017.
[33] M. A. E. S. Mohamed, A. A. E. Mohammed, A. M. A. Elhamed, and M. E. Hessean, "Optimal allocation of photovoltaic based and DSTATCOM in a distribution network under multi load levels," *Eur. J. Eng. Res. Sci.* vol. 4, no. 8, pp. 114–119, 2019.
[34] A. Ramadan, M. Ebeed, and S. Kamel, "Performance Assessment of a Realistic Egyptian Distribution Network Including PV Penetration with DSTATCOM," in *Proc. Int. Conf. Innov. Trends Comput. Eng. (ITCE)*, Feb. 2019, pp. 426–431.
[35] H. Wang, Y. Jin, and J. Doherty, "Committee-based active learning for surrogate-assisted particle swarm optimization of expensive problems," *IEEE Trans. Cybern.*, vol. 47, no. 9, pp. 2664–2677, Sep. 2017.
[36] Y. Jin, H. Wang, T. Chugh, D. Guo, and K. Miettinen, "Data-driven evolutionary optimization: An overview and case studies," *IEEE Trans. Evol. Comput.*, vol. 23, no. 3, pp. 442–458, Jun. 2019.

BIN WANG received the B.S. degree in measurement and control technology and instrumentation from the Xi'an University of Technology, Xi'an, China, in 1994, and the M.S. degree in computer science and technology from Xidian University, Xi'an, in 2004. He is currently an Associate Professor with the Faculty of Computer Science and Engineering, Xi'an University of Technology, Xi'an. He current research interests include evolutionary algorithms and artificial intelligence.

TIAN XIANG received the B.S. degree in computer science and technology from the Xi'an University of Finance and Economics, Xi'an, China, in 2015, where she is currently pursuing the M.S. degree in computer technology with the Faculty of Computer Science and Engineering. Her current research interests are evolutionary computing and machine learning.

NING LI received the M.S. degree in computer science and technology from the Northeast Normal University, Jilin, China, in 2014. He is currently pursuing the Ph.D. degree in pattern recognition and intelligent systems with the Faculty of Computer Science and Engineering, Xi'an University of Technology, Xi'an, China. He is also an Engineer with the Faculty of Computer, Baoji University of Arts and Sciences, Baoji, China. His current research interests include evolutionary algorithms, neural networks, and multiobjective optimization.

WENJUAN HE received the B.S. degree in computer management and application from the Xi'an University of Finance and Economics, Xi'an, China, in 1997, and the M.S. degree in computer science and technology from the Xi'an University of Technology, Xi'an, in 2004. She is currently a Lecturer with the Faculty of Computer Science and Engineering, Xi'an University of Technology. Her current research interests include evolutionary algorithms and artificial intelligence.



WEI LI received the B.S. degree in computer education from Shaanxi Normal University, Xi'an, China, in 1995, and the M.S. and Ph.D. degrees in computer science and technology from the Xi'an University of Technology, Xi'an, in 2001 and 2016, respectively. She is currently an Associate Professor with the Faculty of Computer Science and Engineering, Xi'an University of Technology. Her current research interests include evolutionary algorithms, neural networks, and data mining.

XINHONG HEI received the B.S. and M.S. degrees in computer science and technology from the Xi'an University of Technology, Xi'an, China, in 1998 and 2003, respectively, and the Ph.D. degree from Nihon University, Tokyo, Japan, in 2008. He is currently a Professor with the Faculty of Computer Science and Engineering, Xi'an University of Technology. His current research interests include AI, big data, and safety-critical systems.