Journal of Network Computing and Applications (2025)
Clausius Scientific Press, Canada

DOI: 10.23977/jnca.2025.100107
ISSN 2371-9214 Vol. 10 Num. 1

A Nature-inspired Fully Enhanced Hybrid Algorithm Based on Intra-group Competition Mechanism

Rongguo Qu*, Zhenxing Yu, Dewei Yang, Jin Su, Qinwei Fan

College of Science, Xi'an Polytechnic University, Xi'an, Shanxi, 710600, China
**Corresponding author*

Keywords: Flower Pollination Algorithm, Coati Optimization Algorithm, K-means Clustering Method, Competition Mechanism, Polynomial Mutation

Abstract: The flower pollination algorithm exhibits notable strengths, including robust search capabilities, minimal parameter requirements, and a straightforward architecture, but due to the randomness of its local search, it leads to slow convergence. Comparatively, the raccoon optimization algorithm does not require parameter adjustment, and the local search range is gradually reduced over time, ensuring the algorithm's effectiveness and convergence. However, for solving high-dimensional complex problems, the global search time is too long to reach the optimal global solution. Therefore, This study introduces a novel coati flower pollination algorithm incorporating an intra-group competition mechanism, effectively integrating the global exploration capabilities of FPA with the local exploitation characteristics of COA. The algorithm divides the population by k-means clustering to improve diversity and utilizes the competition mechanism to promote information exchange among individuals. For winning and losing individuals, the improved flower pollination algorithm and coati optimization algorithm are used for iterative updating, respectively, and adaptive polynomial mutation is introduced to avoid local optima. The superiority of the algorithm is verified on the CEC2017.

1. Introduction

It is well-known that optimization problems are prevalent in various fields, covering engineering design [1], shop floor scheduling [2], image processing [3], face recognition [4], feature selection [5], stock prediction [6], air quality detection [7], and many other practical applications [8], [9]. The common feature of these problems is seeking optimal solutions or decisions to maximize or minimize a specific objective function [10]. However, since traditional methods are usually based on deterministic rules or mathematical models, their performance is limited when confronted with complex, nonlinear, or high-dimensional optimization problems. For example, in engineering design, the combination of optimization variables and the complexity of constraints often prevent traditional analytical methods from finding the global optimal solution [11]. Similarly, in feature selection, when confronted with high-dimensional datasets, there is the challenge of removing redundant features and selecting the optimal combination of features to ensure that the data can be utilized more efficiently [12].

To address the inherent constraints associated with conventional approaches, researchers have

started to explore more flexible and adaptable solutions. Heuristic algorithms [13], [14], as a common alternative, search for optimal solutions by simulating the evolution of nature, principles of physics, or artificial intelligence techniques. These algorithms include genetic algorithms [15], ant colony algorithms [16], particle swarm optimization [17], and so on. In contrast to conventional approaches, heuristic algorithms demonstrate enhanced robustness and superior global exploration capabilities, effectively mitigating the risk of local optima entrapment while maintaining applicability across diverse optimization scenarios. Nevertheless, as established by the "No Free Lunch" theorem [18], universal optimization methods do not exist, implying that each algorithmic approach possesses distinct strengths and inherent limitations. Therefore, to better solve real-life problems, researchers propose many other new algorithms or improve existing algorithms, or even mix them to achieve better results [19].

Particularly, hybrid optimization methodologies [20]-[22] have garnered significant research interest due to their ability to synergistically integrate the merits of diverse algorithms for addressing complex optimization challenges. The fundamental principle underlying these hybrid approaches lies in the strategic combination or sequential integration of multiple optimization techniques, thereby leveraging their complementary strengths while mitigating individual limitations. With hybrid algorithms, higher search efficiency, better convergence, and more accurate results can be achieved. For example, in 2021, Hussain. et al. found that the global optimization search of Harris Hawks Optimization (HHO) [23] is too stochastic, which may result in the algorithm failing to search the effective region, and to ameliorate this problem, an efficient hybrid sine-cosine Harris Hawks Optimization (SCAHHO) [24] was proposed. Since the Sine-cosine Algorithm (SCA) [25] possesses the properties of both sine and cosine functions and can perform an effective global search in the solution space, the algorithm uses the search strategy of SCA to replace the global search strategy of the original HHO, which improves the global search capability of the HHO and at the same time ensures that the algorithm can efficiently converge to the global optimal solution. 2022, Ewees. et al. proposed a Slime Mould Algorithm (SMA) improved Gradient-based optimizer (GBO) [26]. Owing to its distinctive characteristics including accelerated convergence rates, enhanced robustness, and superior solution precision, the SMA [27] has been strategically integrated to address the inherent limitations of the gradient-based optimizer (GBO), particularly its inadequate local exploitation capability frequently leading to premature convergence. This hybrid approach incorporates SMA as a local search operator within the GBO framework [28], thereby significantly improving the algorithm's capacity for comprehensive exploration across the solution space.

However, combining multiple algorithms to construct a good hybrid algorithm is difficult because the results of hybrid algorithms often depend on how multiple algorithms are constructed. So far, most of the hybrid algorithm construction methods use a randomized hybrid approach. And this randomized hybrid approach is to choose each strategy randomly in each iteration, and the convergence performance is difficult to guarantee. Therefore, when constructing hybrid algorithms, while considering the randomness of the algorithms, it is more necessary to consider selecting their corresponding iteration strategies for different populations to better ensure their convergence performance.

The flower pollination algorithm (FPA) [29], developed by Yang in 2012, represents a biologically-inspired computational optimization technique that mimics the natural pollination processes observed in flowering plants. This algorithm demonstrates distinctive characteristics including robust search capabilities, minimal parameter requirements, and structural simplicity. However, its search mechanism is constrained by the conversion probability parameter, leading to challenges in maintaining an optimal balance between exploitation and exploration processes. Furthermore, the algorithm's local search strategy exhibits excessive randomness, resulting in suboptimal convergence rates toward global optima. The Coati Optimization Algorithm (COA) [30] is a new heuristic

algorithm proposed by Dehghani. et al. In 2022, researchers developed the coati optimization algorithm (COA) through systematic investigation of coati predation behaviors and predator avoidance mechanisms. This algorithm exhibits distinctive advantages, including parameter-free operation and an adaptive local search range that progressively narrows during the optimization process, thereby ensuring both search efficiency and algorithmic convergence. The incorporation of COA's local search strategy effectively addresses FPA's convergence limitations caused by excessive randomness in its local search mechanism. Therefore, these two algorithms are mixed in this paper.

Different from the above hybrid class algorithms that only perform a simple combination of multiple algorithms, the algorithm proposed in this paper divides the population and uses a competition mechanism for each divided sub-population, choosing different iteration strategies for different individuals, which improves the problem that convergence is difficult to guarantee in hybrid algorithms. The principal contributions of this research can be summarized as follows:

(a) The k-means clustering technique is employed to partition the randomly initialized population, effectively segmenting the solution space into distinct regions. This strategic division enhances the algorithm's exploration capability across the entire search domain. Furthermore, this approach significantly minimizes redundant exploration of similar solutions, thereby accelerating the algorithm's convergence rate.

(b) Use of competition mechanism in each divided subpopulation. Within each subpopulation, individuals are systematically sorted in ascending order based on their fitness evaluations. The superior half of these ranked individuals are classified as dominant solutions, while the remaining members are designated as non-dominant solutions. Moreover, the winning individuals are iteratively updated using the coati local search method, and the failed individuals are iteratively updated using the improved flower pollination algorithm.

(c) Enhancement of the flower pollination algorithm's search strategy is presented. This study, inspired by the PSO algorithm, integrates the local and global search methods of the flower pollination algorithm into a unified formula. By employing two operators to regulate these search methods, it removes the reliance on conversion probability and facilitates more efficient exploration within the solution space.

(d) Adaptive polynomial mutation of globally optimal individuals. To prevent the algorithm from getting trapped in a local optimum, after each update iteration, the global optimum is mutated using a polynomial mutation strategy, and the mutated individual is compared with the unmutated individual to select the one with the better fitness value as the final individual.

The remainder of this manuscript is structured as follows: Section 2 elaborates on the fundamental principles underlying both the flower pollination algorithm (FPA) and coati optimization algorithm (COA). Section 3 provides a comprehensive description of the proposed KSCOFPA methodology. Section 4 presents extensive experimental results and corresponding discussions. Finally, Section 5 concludes the study with a summary of key findings and outlines potential directions for future investigations.

2. Preliminaries

2.1 Flower Pollination Algorithm

Self-pollination and cross-pollination are the two primary methods of pollination. Self-pollination occurs when a plant's pollen fertilizes its flowers in the absence of reliable pollinators. In contrast, cross-pollination involves the transfer of pollen between different plants. The behavior of pollinators during cross-pollination follows a specific distribution. This study assumes that each plant has only one flower and one pollen gamete, with each gamete linked to a unique solution, adhering to a defined set of pollination rules.

(1) Global pollination involves the transfer of pollen gametes by transmitters through $Lévy$ flight, a process known as heterogeneous pollination.

(2) Abiotic self-pollination, a form of local pollination, occurs when a plant fertilizes its own flowers using its pollen without external pollinator intervention.

(3) Flower constancy posits that the reproductive success between two flowers is influenced by their level of similarity.

(4) The probability of conversion $p \in [0,1]$ dictates the extent to which cross-pollination shifts to self-pollination, as factors such as distance bias the process toward self-pollination.

In the initial phase of FPA, a population $P(t) = \{X_t^i\}$, $X_t^i = \left[x_{i,1}^t, x_{i,2}^t, \cdots, x_{i,j}^t, \cdots, x_{i,D}^t\right]$, $j = 1,2,\cdots,D$; $i = 1,2,\cdots,N$ consisting of N individuals is randomly initialized. Here, D denotes the dimensionality of the optimization problem, t represents the current iteration count, and the population size N reflects both the diversity of potential solutions and the number of individuals undergoing iteration at each step. This setup ensures a broad exploration of the solution space while maintaining computational efficiency. The random generation of the population facilitates the algorithm's ability to explore diverse regions, which is crucial for identifying optimal or near-optimal solutions in complex optimization tasks.

The self-pollination process, also known as local pollination, can be described as follows:

$$X_i^{t+1} = X_i^t + \varepsilon\left(X_j^t - X_k^t\right) \tag{1}$$

Here, the variables X_j^t and X_k^t represent pollen originating from two distinct flowers belonging to the same plant species. These variables are modeled as randomly selected integers within the population. Furthermore, the scaling factor ε is a random variable uniformly sampled from the range [0,1], introducing stochasticity into the local pollination process.

In this study, heterogeneous pollination is characterized as a global pollination process where pollinators, such as insects or birds, engage in long-distance flights to enable cross-pollination between distant flowers. This mechanism can be mathematically expressed as:

$$X_i^{t+1} = X_i^t + L(X_i^t - g^*) \tag{2}$$

$$L \sim \frac{\lambda\Gamma(\lambda)sin(\pi\lambda/2)}{\pi}\frac{1}{s^{1+\lambda}}$$

Here, X_i^t denotes the position of the i-th pollen particle at iteration t, and g^* represents the current population's best solution. Additionally, the intensity of pollination is determined by the parameter$L$, which specifies the step size according to the $Lévy$ distribution. Here, $\Gamma(\lambda)$ is the standard gamma function, and this distribution is valid for large steps $s > 0$. In all our simulations below, we have used λ= 1.5. The switch between local and global pollination is controlled by the probability p. The FPA algorithm gives its highest performance when the transition probability p equals 0.8, as demonstrated by reference [29]. The step by step procedure of FPA is presented in Algorithm1.

Algorithm1: Flower Pollination Algorithm

1: Setting parameters (current iteration number t, maximum iteration number T, population size N, transition probability p)
2: Initialize pollen position
3: Calculate all pollen fitness values and save the global optimum
4: **While** $t \leq T$ **do**
5: **For** $i = 1 : N$ **do**
6: **If** rand < p **do**
7: Use (2) to update pollen location
8: **Else**

9: Use (1) to update pollen location
10: **End If**
11: Boundary processing of transgressing individual locations
12: Calculating updated pollen fitness values to optimally preserve pollen positions
13: **End For**
14: Save the global optimal pollen position for this iteration
15: **End While**
16: Return the optimal pollen position and the corresponding fitness value

2.2 Coati Optimization Algorithm

The COA population undergoes two distinct phases of updating, which are inspired by two natural behaviors exhibited by coatis. These behaviors are:

(1) The strategy coatis employ when attacking iguanas,

(2) The escape strategy they employ when evading predators.

Hence, the process of updating the position of candidate solutions in the COA relies on modeling these two behaviors of coatis.

2.2.1 Hunting and attacking strategy on iguana

This particular strategy involves a coordinated effort by a group of coatis to climb a tree and approach an iguana with the intent to frighten it. Meanwhile, several other coatis remain on the ground, positioned under the tree, patiently waiting for the iguana to eventually fall down. Once the iguana falls to the ground, the coatis swiftly attack and hunt it. This strategy effectively prompts the coatis to explore various positions within the search space, highlighting the COA's capability for global search in problem-solving scenarios.

To mathematically simulate the position of the coatis as they ascend from the tree, we can employ the following equation:

$$x_{i,j}^{P1} = x_{i,j} + r \cdot \left(Igu_j - I \cdot x_{i,j}\right),\ i = 1,2,\ldots,\frac{N}{2};\quad j = 1,2,\ldots,m. \tag{3}$$

Once the iguana falls to the ground, it is randomly positioned within the search space. Subsequently, the coatis on the ground adjust their positions in the search space based on this random placement. This adjustment process can be mathematically simulated using the equations:

$$Igu_j^G = lb_j + r \cdot (ub_j - lb_j),$$
$$X_{i,j}^{P1} = \begin{cases} X_{i,j} + r \cdot (Igu_j^G - I \cdot X_{i,j}), & F_{Igu}^G < F_i \\ X_{i,j} + r \cdot (X_{i,j} - Igu_j^G), & else \end{cases}$$
$$i = \frac{N}{2}+1, \frac{N}{2}+2, \cdots, N;\quad j = 1,2,\cdots,m \tag{4}$$

To determine whether the newly calculated position for each coati is acceptable for the update process, a condition is applied. If the new position leads to an improvement in the value of the objective function, the coati adopts the new position. Otherwise, the coati remains in its previous position. This update condition is applied for each coati, denoted by the index $i = 1,2,\ldots,N$, and can be represented by the equation:

$$X_i = \begin{cases} X_i^{P1}, & F_i^{P1} < F_i \\ X_i, & else \end{cases} \tag{5}$$

Here, X_i^{P1} represents the new position calculated for the ith coati. $x_{i,j}^{P1}$ corresponds to the jth dimension of the ith coati's position. F_i^{P1} denotes the objective function value associated with the ith coati. r is a random real number within the range [0,1]. Igu signifies the position of the best member, which is equivalent to the iguana's position in the search space. Igu_j represents the jth dimension of the iguana's position. I is an integer randomly selected from the set {1, 2}. Igu^G stands for the position of the iguana on the ground, which is randomly generated. Igu_j^G denotes the jth dimension of the iguana's position on the ground. F_{Igu}^G represents the objective function value associated with the iguana's position on the ground. The symbol $\cdot$ refers to the floor function, which returns the greatest integer less than or equal to a given number.

2.2.2 The process of escaping from predators

In the described scenario, when a predator threatens a coati, the coati quickly reacts by escaping from its current position. The coati's movement in this strategy aims to relocate itself to a nearby safe position. This behavior demonstrates the COA's ability for local search and exploitation. To simulate this escaping behavior, a random position is generated near the current position of each coati. This generation process can be modeled using equations.

$$lb_j^{local} = \frac{lb_j}{t};\quad ub_j^{local} = \frac{ub_j}{t}$$
$$i = 1,2,\cdots,N;\quad j = 1,2,\cdots,m;\quad t = 1,2,\cdots,T \tag{6}$$

$$X_{i,j}^{P2} = X_{i,j} + (1-2r)\cdot(lb_j^{local} + r\cdot(ub_j^{local} - lb_j^{local})),$$
$$i = 1,2,\cdots,N;\quad j = 1,2,\cdots,m \tag{7}$$

To determine whether the newly calculated position is acceptable or not, a condition is applied based on the improvement of the objective function value. This condition can be simulated using the equation:

$$X_i = \begin{cases} X_i^{P2}, & F_i^{P2} < F_i \\ X_i, & else \end{cases} \tag{8}$$

In the given context: X_i^{P2} represents the new position calculated for the ith coati based on the second phase of COA. $x_{i,j}^{P2}$corresponds to the jth dimension of the ith coati's position in the second phase. F_i^{P2}denotes the objective function value associated with the i th coati in the second phase. r is a random number within the range [0,1]. t represents the iteration counter, indicating the current iteration of the algorithm. lb_j^{local} and ub_j^{local} refer to the local lower bound and local upper bound, respectively, of the jth decision variable. lb_j and ub_j represent the lower bound and upper bound, respectively, of the jth decision variable. The step-by-step procedure for the COA is presented in Algorithm 2.

Algorithm 2: Coati Optimization Algorithm

1: Setting parameters (current iteration number t, maximum iteration number T, population size N)
2: Initialize coati population position
3: Calculate all individual fitness values and save the global optimum
4: **While** t ≤ T **do**
5: **For** i = 1 : N/2 **do**
6: Calculate individual positions using (3)
7: Calculate individual positions using (5)
8: **End For**

```
    For  i = (N/2+1) : N  do
        Calculate individual positions using (4)
        Calculate individual positions using (5)
    End For
    Calculate local upper and lower bounds using (6)
    For  i = 1 : N  do
        Calculate individual positions using (7)
        Update individual locations using (8)
    End For
    Select the current optimal individual position
End While
Return the optimal individual position and the corresponding fitness value
```

3. The proposed method

The original FPA algorithm has the advantages of simple structure, no need for gradient information, fewer parameters, easy to implement, etc. However, FPA, like other intelligent optimization algorithms, suffers from defects such as easy to fall into local optimums and slow convergence speed. In order to improve its deficiencies, this paper proposes a competitive coati flower pollination algorithm based on k-means clustering(KSCOFPA).

The algorithm adopts the k-means clustering method to divide the population into k different categories. Then, a population competition mechanism is introduced in each category to rank the fitness values of all particles in that category, and the top 50\% of individuals with better fitness values are classified as the winning individuals, and the rest as the losing individuals. Since the local update of the coati optimization algorithm is to generate a random location near the location of each coati, and the random location will become smaller and smaller as the number of iterations grows, this can ensure the effectiveness of its local search and the convergence of the algorithm.

Therefore, this paper adopts the coati local search strategy as the updating method for the winning individuals. For the failed individuals, this paper adopts the combination of self-flowering and hetero-flowering pollination of flower pollination for updating, and the positive cosine operator is used to control the change of the search center of gravity. In addition, this paper modifies the pollen object of self-pollination in the flower pollination algorithm, and transforms the pollen object of self-pollination into the winning individual corresponding to its ranking, in order to improve the search efficiency of the algorithm.

Finally, this paper uses the adaptive polynomial mutation strategy for the global optimal solution in the iterative update process, which avoids the population from falling into a local optimum and at the same time provides the possibility of exploring other more optimal solutions.

3.1 Segmentation of populations based on k-means clustering algorithm

It is well known that most of the intelligent optimization algorithms use the method of exchanging information by all individuals uniformly in an overall solution space. Although this approach can help populations to share information comprehensively, it also suffers from the problems that may cause the algorithms to fall into local optimal solutions prematurely and increase the computational and storage costs. To solve this problem, KSCOFPA uses the k-means clustering algorithm[31] to divide the population before each iteration update, which is equivalent to dividing the population into different sub-populations based on the space in which they are located. This not only makes the individuals in the sub-populations relatively concentrated and the information exchange relatively easy, but also effectively promotes the individuals to fully explore and discover potential high-quality solutions in their respective spaces, and also improves the search efficiency of the algorithm.

3.2 Competition Mechanism

Next, in order to promote information exchange between individuals, KSCOFPA uses a competition mechanism in each divided sub-population[32]. That is, all individuals in that subpopulation are ranked according to the size of their fitness values, and the top 50% of the population individuals with better fitness values are rounded down as winners, and the rest as losers; for the winning individuals, they are updated according to the iterative updating method of the winners; for the rest of the losers, they are updated according to the updating method of the losers.

3.2.1 Winner update strategy

For the individual who wins the competition in the sub-population, it is updated using the local update strategy of the raccoon optimization algorithm, which can gradually reduce the search area as the iteration time grows, and can ensure the randomness and convergence of the winning individual's search. The specific update formula is as follows:

$$X_{win_{i,j}}^{t+1} = X_{win_{i,j}}^{t} + (1-2r)\cdot(lb_j^{local} + r\cdot(ub_j^{local} - lb_j^{local})), \quad i=1,2,\cdots,N; \quad j=1,2,\cdots,m \tag{9}$$

where $X_{win_{i,j}}^{t+1}$ denotes the j-th dimension of the i-th winner at the (t+1)-st iteration, $X_{win_{i,j}}^{t}$ denotes the j-th dimension of the i-th winner at the t-th iteration, and the rest of the parameters are the same as equation (6) and equation (7).

3.2.2 Update strategy for losers

For individuals that failed to compete in a subpopulation, a combination of self-pollination and allopollination was used for updating, and a positive cosine operator [33] was used instead of the transition probability to control the transition of the search center of gravity. In addition, the two random homozygous pollen individuals during self-pollination are replaced with this loser pollen individual and its corresponding winning pollen individual, and if this loser individual does not have a corresponding winning individual, a winning individual is randomly selected for learning. The specific mathematical expression is as follows:

$$\begin{aligned} c_1 &= A\times\sin((1-\frac{t}{T})\times\frac{\pi}{2})+B, \\ c_2 &= A\times\cos((1-\frac{t}{T})\times\frac{\pi}{2})+B, \\ X_{los_i}^{t+1} &= X_{los_i}^{t} + c_1\times L\times(X_{los_i}^{t} - G_{best}) + c_2\times\varepsilon\times(X_{win_i}^{t} - X_{los_i}^{t}) \end{aligned} \tag{10}$$

where A=2, B=0.5, then t and T denote the current and maximum number of iterations, respectively. $X_{win_i}^{t}$ denotes the i-th winning individual at the t-th iteration in a given subpopulation; $X_{los_i}^{t}$ denotes the i-th failing individual at the t-th iteration in a given subpopulation, G_{best} is the current global optimal individual, and the rest of the parameters are the same as in the flower pollination algorithm.

3.3 Adaptive polynomial mutation strategy

In order to avoid the algorithm from falling into a local optimum, KSCOFPA sets an adaptive polynomial variation on the global optimum each time. Specifically, an adaptive mutation probability P is set after all individuals are updated at the end of this iteration, and a polynomial mutation is performed on the current global optimum when the mutation probability is larger than a random number $rand\in(0,1)$ [34].

$$P = \frac{S_t}{S_0} \tag{11}$$

where S_t denotes the standard deviation of the fitness value of the t-th iteration of the population, and S_0 denotes the standard deviation of the fitness value at the initialization of the population.

$$G_{best_j}^{new} = G_{best_j} + \delta \times (u_j - l_j) \tag{12}$$

$$\delta = \begin{cases} [2\mu + (1-2\mu) \times (1-\delta)^{\eta_m+1}]^{\frac{1}{\eta_m+1}} - 1, & \mu \le 0.5 \\ 1 - [2(1-\mu) + 2(\mu - 0.5) \times (1-\delta_2)^{\eta_m+1}]^{\frac{1}{\eta_m+1}}, & else \end{cases} \tag{13}$$

$$\delta_1 = \frac{X_{i,j}^t - l_j}{u_j - l_j}, \quad \delta 2 = \frac{u_j - X_{i,j}^t}{u_j - l_j} \tag{14}$$

Where, G_{best_j} where denotes the jth dimension of the globally optimal solution in this iteration. $G_{best_j}^{new}$ denotes the jth dimension of the global optimal solution mutated in this iteration, random number $\mu \in [0,1]$, η_m is the distribution index, l_j denotes the lower bound of the j-th dimension of the population, u_j denotes the upper bound of the j-th dimension of the population.

3.4 Algorithmic Pseudocode

Algorithm 4: KSCOFPA

1: Setting parameters (current iteration number t, maximum iteration number T, population size N)
2: Initialize population position
3: Calculate all individual fitness values and save the global optimum
4: While t ≤ T do
5: Use k-means to divide the population into k sub-populations
6: For i = 1 : k do
7: Sorting all individuals in the ith subpopulation
8: Consider the top 50% of individuals as winners and the rest as losers
9: Use (10) to update the loser's position
10: Calculate the fitness value and select the individual with the better fitness value as the new generation
11: Use (9) to update the location of the successor
12: Calculate the fitness value and select the individual with the better fitness value as the new generation ~~winner~~
13: End For
14: Selecting the contemporary globally optimal individual
15: Use (11) to calculate the mutation probability P
16: If P > rand do
17: Mutation of globally optimal individuals using (12)
18: End If
19: End While
20: Return the optimal individual position and the corresponding fitness value

4. Experiments and Simulations

In order to demonstrate the effectiveness of KSCOFPA, in this section we perform performance tests of KSCOFPA on the CEC2017 test suite [35]. This section is divided into three main subsections;

the first subsection provides a description of the parameter settings for each algorithm, and the second subsection describes the algorithms comparing the experiments on the CEC2017 test suite.

EXPERIMENTAL ENVIRONMENT: all experiments were conducted using Matlab R2024a on a personal computer equipped with an Intel i5-1240 CPU (running at 1.7 GHz) and 16.00 GB RAM.

4.1 Experimental Parameter Settings

The parameter settings for each algorithm used in this experiment are shown in Table 1:

Table 1 Algorithm experiment parameter setting table

Algorithms	Parameter settings
AO	$\beta=1.5, s=0.01, U=0.00565, \omega=0.005, \alpha=0.1, \delta=0.1.$
AOA	$MOP_{Max}=1, MOP_{Min}=0.2, \alpha=5, Mu=0.499.$
FPA	$p=0.8, \lambda=1.5, \varepsilon \in U[0,1]$.
SSA	c_1, c_2 randomized distribution.
SCA	$\alpha=2.$
HHO	$\beta=1.5.$
WOA	α decreases linearly from 2 to 0.
KSCOFPA	$k=3, \lambda=1.5, \varepsilon \in U[0,1].$

4.2 Comparison Experiments

1) Experiment Description

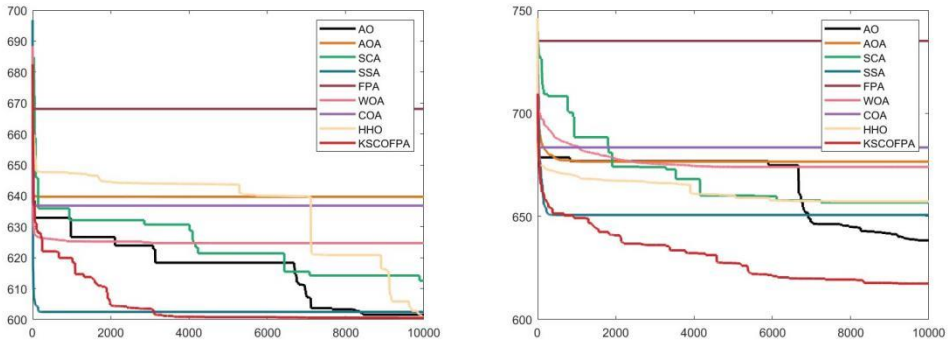
In this experiment, eight other well-known algorithms (including AO[36], AOA [37], HHO [38], SCA [25], SSA [39], COA [30], FPA [29] and WOA [40]) are selected as comparison algorithms for KSCOFPA, and their performances relative to KSCOFPA on the CEC2017 test set are tested separately. The experiments were performed on the 10 dimensions of the CEC2017 test set. The number of initialized populations was 50, the number of iterations was 10000, and the experiments on each function were run independently for 30 times. The results of each experiment were tested using Friedman rank sum test [41] and Wilcoxon rank sum test [42]. In addition, “Mean Rank” denotes the average of the ranks of the Friedman rank sum test for each algorithm; “Total Rank” denotes the overall performance ranking of each algorithm. The symbols “+”, “-”, and “≈” indicate that KSCOFPA outperforms, underperforms, or is similar to the comparison algorithms. “P-Value” denotes the significant result of Wilcoxon rank sum test based on multiple problems. “R+/R-” denotes the corresponding rank sum. When P is less than 0.05, it indicates that KSCOFPA outperforms other comparison algorithms.

2) Experimental results

Table 2 shows the function mean and standard deviation of KSCOFPA and other comparison algorithms after 30 runs in the 10-dimensional CEC2017 test suite. The experimental results show that KSCOFPA outperforms the comparison algorithms in all 10-dimensional CEC2017 test experiments. It ranked first in all 10-dimensional CEC2017 test experiment results and its optimization performance was significantly different from all other eight comparison algorithms in Wilcoxon rank sum test. Fig. 1 lists the algorithmic convergence curves of these eight algorithms on the two test functions (the 6th function and the 8th function of CEC2017), through which it can be clearly observed that the algorithmic optimization search performance of KSCOFPA is significantly better than the other seven comparison algorithms.

Table 2 Table of experimental results of algorithm comparison on 10-dim CEC2017 test set

Fun	Meas	AO	AOA	COA	FPA	SCA	SSA	WOA	HHO	KSCOFPA
F1	Ave	3.98E+04	9.74E+03	7.78E+09	1.53E+10	3.58E+08	4.78E+03	1.07E+04	1.09E+05	4.20E+02
	Std	2.57E+04	3.26E+03	2.54E+09	5.89E+09	1.58E+08	1.84E+03	1.83E+04	3.90E+04	6.51E+02
F3	Ave	3.00E+02	3.00E+02	8.24E+03	1.13E+05	7.44E+02	3.24E+03	3.10E+02	3.00E+02	3.00E+02
	Std	1.01E-01	6.91E-03	2.66E+03	1.47E+05	4.18E+02	8.30E+03	2.21E+01	1.53E-01	3.03E-07
F4	Ave	4.04E+02	4.08E+02	8.42E+02	2.18E+03	4.23E+02	4.27E+02	4.06E+02	4.04E+02	4.01E+02
	Std	2.15E+00	5.16E+00	1.64E+02	7.62E+02	3.98E+00	4.84E+01	1.46E+00	1.17E+00	1.20E+00
F5	Ave	5.21E+02	5.49E+02	5.85E+02	6.46E+02	5.35E+02	5.33E+02	5.48E+02	5.31E+02	5.19E+02
	Std	7.27E+00	1.86E+01	1.41E+01	1.80E+01	5.33E+00	1.42E+01	1.34E+01	1.11E+01	4.92E+00
F6	Ave	6.03E+02	6.37E+02	6.42E+02	6.93E+02	6.13E+02	6.06E+02	6.22E+02	6.11E+02	6.00E+02
	Std	2.98E+00	4.31E+00	8.52E+00	1.48E+01	1.59E+00	7.23E+00	1.27E+01	5.77E+00	3.52E-02
F7	Ave	7.40E+02	7.96E+02	7.79E+02	1.16E+03	7.66E+02	7.26E+02	7.72E+02	7.52E+02	7.21E+02
	Std	1.33E+01	1.15E+01	1.75E+01	8.29E+01	5.35E+00	7.44E+00	1.87E+01	1.46E+01	9.02E+00
F8	Ave	8.22E+02	8.24E+02	8.53E+02	9.30E+02	8.29E+02	8.25E+02	8.37E+02	8.24E+02	8.02E+02
	Std	6.98E+00	3.99E+00	5.99E+00	1.11E+01	4.58E+00	7.78E+00	1.00E+01	6.81E+00	3.19E+00
F9	Ave	9.04E+02	1.35E+03	1.37E+03	4.68E+03	9.44E+02	9.90E+02	1.18E+03	1.16E+03	9.12E+02
	Std	8.85E+00	1.97E+02	1.91E+02	5.40E+02	2.08E+01	1.43E+02	1.29E+02	1.51E+02	5.05E+01
F10	Ave	1.72E+03	2.04E+03	2.41E+03	3.37E+03	1.85E+03	1.58E+03	1.75E+03	1.56E+03	1.49E+03
	Std	2.37E+02	3.25E+02	1.69E+02	3.94E+02	1.90E+02	1.41E+02	2.45E+02	1.73E+02	1.34E+02
F11	Ave	1.13E+03	1.13E+03	1.83E+03	9.59E+03	1.15E+03	1.18E+03	1.16E+03	1.13E+03	1.11E+03
	Std	1.66E+01	7.15E+00	5.03E+02	7.49E+03	6.37E+00	7.91E+01	4.27E+01	1.03E+01	5.33E+00
F12	Ave	9.82E+05	9.64E+04	2.24E+08	1.74E+09	3.50E+06	3.71E+06	1.42E+06	2.42E+05	6.23E+03
	Std	5.68E+05	4.67E+04	2.62E+08	1.01E+09	2.73E+06	7.23E+06	2.26E+06	2.42E+05	2.63E+03
F13	Ave	6.51E+03	9.68E+03	6.21E+04	1.55E+08	9.62E+03	5.87E+03	1.38E+04	9.98E+03	1.23E+03
	Std	2.72E+03	7.70E+03	1.03E+05	1.05E+08	3.70E+03	9.16E+03	8.27E+03	6.35E+03	1.50E+01
F14	Ave	1.52E+03	1.27E+04	1.50E+03	2.48E+06	1.50E+03	1.45E+03	1.50E+03	1.48E+03	1.43E+03
	Std	2.99E+01	9.45E+03	2.39E+01	6.50E+06	2.85E+01	2.71E+01	3.70E+01	1.83E+01	1.24E+01
F15	Ave	1.92E+03	5.77E+03	6.62E+03	1.54E+06	1.64E+03	2.11E+03	1.77E+03	1.59E+03	1.50E+03
	Std	3.09E+02	4.61E+03	3.30E+03	3.87E+06	3.77E+01	1.69E+03	9.34E+01	7.84E+01	3.37E+01
F16	Ave	1.67E+03	2.03E+03	2.08E+03	2.50E+03	1.66E+03	1.73E+03	1.79E+03	1.86E+03	1.66E+03
	Std	8.63E+01	1.15E+02	9.40E+01	2.09E+02	3.64E+01	8.74E+01	1.30E+02	1.01E+02	7.52E+01
F17	Ave	1.76E+03	1.89E+03	1.80E+03	2.33E+03	1.75E+03	1.74E+03	1.76E+03	1.75E+03	1.72E+03
	Std	1.96E+01	1.18E+02	2.78E+01	1.19E+02	9.72E+00	1.47E+01	1.72E+01	2.03E+01	1.26E+01
F18	Ave	2.07E+04	1.41E+04	2.64E+06	5.43E+08	4.12E+04	8.12E+03	1.59E+04	1.21E+04	1.82E+03
	Std	1.38E+04	8.21E+03	4.24E+06	3.86E+08	1.88E+04	1.08E+04	1.06E+04	9.97E+03	7.73E+00
F19	Ave	2.12E+03	1.11E+04	3.97E+03	3.35E+07	2.08E+03	1.92E+03	8.92E+03	5.69E+03	1.90E+03
	Std	1.53E+02	1.00E+04	1.68E+03	6.53E+07	7.85E+01	2.64E+01	7.71E+03	4.81E+03	1.67E+00
F20	Ave	2.05E+03	2.11E+03	2.22E+03	2.40E+03	2.06E+03	2.11E+03	2.10E+03	2.11E+03	2.02E+03
	Std	3.78E+01	6.69E+01	4.26E+01	8.80E+01	7.89E+00	5.29E+01	5.66E+01	6.34E+01	1.24E+01
F21	Ave	2.21E+03	2.30E+03	2.37E+03	2.44E+03	2.21E+03	2.30E+03	2.27E+03	2.31E+03	2.24E+03
	Std	3.36E+01	5.68E+01	2.05E+01	1.79E+01	1.50E+00	5.25E+01	6.50E+01	5.49E+01	5.47E+01
F22	Ave	2.30E+03	2.45E+03	2.97E+03	3.70E+03	2.34E+03	2.31E+03	2.31E+03	2.31E+03	2.30E+03
	Std	1.28E+00	8.05E+01	2.07E+02	4.73E+02	7.54E+00	1.30E+01	5.20E+00	2.69E+01	1.83E+01
F23	Ave	2.63E+03	2.70E+03	2.72E+03	2.79E+03	2.64E+03	2.63E+03	2.64E+03	2.65E+03	2.64E+03
	Std	1.24E+01	3.50E+01	2.20E+01	4.62E+01	4.49E+00	1.09E+01	1.56E+01	1.79E+01	1.14E+01
F24	Ave	2.69E+03	2.78E+03	2.85E+03	2.93E+03	2.68E+03	2.76E+03	2.74E+03	2.79E+03	2.51E+03
	Std	1.23E+02	1.20E+02	8.93E+01	5.78E+01	1.09E+02	4.63E+01	8.37E+01	1.05E+02	3.35E+01
F25	Ave	2.92E+03	2.93E+03	3.40E+03	4.40E+03	2.93E+03	2.96E+03	2.94E+03	2.94E+03	2.91E+03
	Std	2.39E+01	2.54E+01	1.56E+02	3.86E+02	1.30E+01	3.95E+01	2.25E+01	3.50E+01	2.22E+01
F26	Ave	2.99E+03	3.72E+03	3.92E+03	4.95E+03	3.02E+03	3.46E+03	3.12E+03	3.16E+03	2.81E+03
	Std	6.75E+01	3.84E+02	4.18E+02	3.71E+02	2.70E+01	5.04E+02	4.39E+02	3.93E+02	1.43E+02
F27	Ave	3.10E+03	3.21E+03	3.18E+03	3.31E+03	3.10E+03	3.10E+03	3.12E+03	3.13E+03	3.12E+03
	Std	2.97E+00	2.59E+01	3.52E+01	6.72E+01	1.23E+00	1.67E+01	2.99E+01	2.72E+01	1.41E+01
F28	Ave	3.30E+03	3.33E+03	3.75E+03	4.01E+03	3.21E+03	3.40E+03	3.27E+03	3.32E+03	3.21E+03
	Std	1.13E+02	1.48E+02	1.14E+02	1.46E+02	1.55E+01	1.32E+02	1.81E+02	1.89E+02	1.26E+02
F29	Ave	3.18E+03	3.30E+03	3.37E+03	3.93E+03	3.18E+03	3.23E+03	3.27E+03	3.25E+03	3.19E+03
	Std	1.54E+01	9.96E+01	5.53E+01	1.87E+02	1.25E+01	3.14E+01	6.54E+01	7.74E+01	2.47E+01
F30	Ave	2.50E+04	3.37E+05	3.60E+06	9.00E+07	1.75E+05	7.18E+05	4.02E+04	3.69E+05	3.41E+03
	Std	2.95E+04	3.31E+05	3.95E+06	2.18E+07	1.55E+05	9.09E+05	5.22E+04	5.09E+05	5.44E+02
+/-/≈		24/5/0	29/0/0	29/0/0	29/0/0	25/4/0	27/2/0	28/1/0	29/0/0	−
R+/R-		383/52	435/0	435/0	435/0	410/25	427/8	434/1	435/0	−
P-Value		< 0.001	< 0.001	< 0.001	< 0.001	< 0.001	< 0.001	< 0.001	< 0.001	−
α=0.05		Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	−
Mean Rank		3.07	5.9	7.66	9	4.07	4.38	5.07	4.45	1.41
Total Rank		2	7	8	9	3	4	6	5	1



(a) Convergence curve of the 6th test function (b) Convergence curves for the 6th test function

Fig. 1 Algorithm iteration convergence plot for CEC2017 comparison experiments

5. Conclusion and future directions

This paper presents a competitive coati flower pollination algorithm utilizing k-means clustering for population grouping. The k-means algorithm enhances individual learning efficiency and accelerates convergence. A competition mechanism within subpopulations allows for varied evolutionary strategies, facilitating effective information exchange. Winning individuals are updated using a coati local search strategy, which gradually narrows the random search range, ensuring convergence to the global optimum. For losing individuals, an improved flower pollination algorithm is employed, integrating global and local searches without using the transformation probability. The sine-cosine operator constrains the search center's movement, enabling a comprehensive exploration of the solution space. An adaptive polynomial mutation strategy is applied to globally optimal individuals to prevent local optima and enhance population diversity. Performance tests on the CEC2017 benchmark, analyzed with Wilcoxon and Friedman rank-sum tests, demonstrate the algorithm's effectiveness and robustness.

In the future, this algorithm is considered for use in high-dimensional feature selection, engineering optimization, and neural network parameter optimization or parameter optimization for a wider range of other models, in addition to considering other effective ways of mixing other heuristic algorithms.

References

[1] Liu J, Chen Y, Liu X, et al. An efficient manta ray foraging optimization algorithm with individual information interaction and fractional derivative mutation for solving complex function extremum and engineering design problems. Applied Soft Computing, 2024, 150: 111042.

[2] Liu H, Zhao F, Wang L, et al. Evolutionary Multitasking Memetic Algorithm for Distributed Hybrid Flow-Shop Scheduling Problem With Deterioration Effect. IEEE Transactions on Automation Science and Engineering, 2024.

[3] Hu G, Zheng Y, Houssein E H, et al. DRPSO: A multi-strategy fusion particle swarm optimization algorithm with a replacement mechanisms for colon cancer pathology image segmentation. Computers in Biology and Medicine, 2024, 178: 108780.

[4] Mishra S, Thamaraiselvi D, Dhariwal S, et al. LSCO: Light spectrum chimp optimization based spinalnet for live face detection and recognition. Expert Systems with Applications, 2024, 250: 123585.

[5] Hu G, Du B, Wang X, et al. An enhanced black widow optimization algorithm for feature selection. Knowledge-Based Systems, 2022, 235: 107638.

[6] Kumar Chandar S. Grey Wolf optimization-Elman neural network model for stock price prediction. Soft Computing, 2021, 25: 649-658.

[7] Liu J B, Zheng Y Q, Lee C C. Statistical analysis of the regional air quality index of Yangtze River Delta based on complex network theory. Applied Energy, 2024, 357: 122529.

[8] Fan Q W, Kang Q, Zurada J M, et al. Convergence Analysis of Online Gradient Method for High-Order Neural

Networks and Their Sparse Optimization. IEEE Transactions on Neural Networks and Learning Systems, 2023.
[9] Fan Q W, Liu L, Zhang Z W, Yang X F, Xing Z W, He X S. Boundedness and convergence analysis of Pi sigma neural network based on online gradient method and its sparse optimization. East Asian Journal on Applied Mathematics, 2023: 1-22.
[10] Azevedo B F, Rocha A M A C, Pereira A I. Hybrid approaches to optimization and machine learning methods: a systematic literature review. Machine Learning, 2024: 1-43.
[11] Abualigah L, Elaziz M A, Khasawneh A M, et al. Meta-heuristic optimization algorithms for solving real-world mechanical engineering design problems: a comprehensive survey, applications, comparative analysis, and results. Neural Computing and Applications, 2022: 1-30.
[12] Askr H, Abdel-Salam M, Hassanien A E. Copula entropy-based golden jackal optimization algorithm for high-dimensional feature selection problems. Expert Systems with Applications, 2024, 238: 121582.
[13] Hashim F A, Hussien A G. Snake Optimizer: A novel meta-heuristic optimization algorithm. Knowledge-Based Systems, 2022, 242: 108320.
[14] Abdel-Basset M, Mohamed R, Jameel M, et al. Spider wasp optimizer: a novel meta-heuristic optimization algorithm. Artificial Intelligence Review, 2023, 56(10): 11675-11738.
[15] Cavallaro C, Cutello V, Pavone M, et al. Machine Learning and Genetic Algorithms: A case study on image reconstruction. Knowledge-Based Systems, 2024, 284: 111194.
[16] Liang W, Lou M, Chen Z, et al. An enhanced ant colony optimization algorithm for global path planning of deep-sea mining vehicles. Ocean Engineering, 2024, 301: 117415.
[17] Kocak O, Erkan U, Toktas A, et al. PSO-based image encryption scheme using modular integrated logistic exponential map. Expert Systems with Applications, 2024, 237: 121452.
[18] Wolpert D H, Macready W G. No free lunch theorems for optimization. IEEE transactions on evolutionary computation, 1997, 1(1): 67-82.
[19] Elseify M A, Hashim F A, Hussien A G, et al. Single and multi-objectives based on an improved golden jackal optimization algorithm for simultaneous integration of multiple capacitors and multi-type DGs in distribution systems. Applied Energy, 2024, 353: 122054.
[20] Zhang H, Ke J. An Intelligent scheduling system and hybrid optimization algorithm for ship locks of the Three Gorges Hub on the Yangtze River. Mechanical Systems and Signal Processing, 2024, 208: 110974.
[21] Hasanien H M, Alsaleh I, Tostado-Véliz M, et al. Hybrid particle swarm and sea horse optimization algorithm-based optimal reactive power dispatch of power systems comprising electric vehicles. Energy, 2024, 286: 129583.
[22] Xu X F, Wang K, Ma W H, et al. Multi-objective particle swarm optimization algorithm based on multi-strategy improvement for hybrid energy storage optimization configuration. Renewable Energy, 2024, 223: 120086.
[23] Heidari A A, Mirjalili S, Faris H, et al. Harris hawks optimization: Algorithm and applications. Future generation computer systems, 2019, 97: 849-872.
[24] Hussain K, Neggaz N, Zhu W, et al. An efficient hybrid sine-cosine Harris hawks optimization for low and high-dimensional feature selection. Expert Systems with Applications, 2021, 176: 114778.
[25] Mirjalili S. SCA: a sine cosine algorithm for solving optimization problems. Knowledge-based systems, 2016, 96: 120-133.
[26] Ewees A A, Ismail F H, Sahlol A T. Gradient-based optimizer improved by Slime Mould Algorithm for global optimization and feature selection for diverse computation problems. Expert Systems with Applications, 2023, 213: 118872.
[27] Li S, Chen H, Wang M, et al. Slime mould algorithm: A new method for stochastic optimization. Future generation computer systems, 2020, 111: 300-323.
[28] Ahmadianfar I, Bozorg-Haddad O, Chu X. Gradient-based optimizer: A new metaheuristic optimization algorithm. Information Sciences, 2020, 540: 131-159.
[29] Yang X S. Flower pollination algorithm for global optimization. In: International conference on unconventional computing and natural computation. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012: 240-249.
[30] Dehghani M, Montazeri Z, Trojovská E, et al. Coati Optimization Algorithm: A new bio-inspired metaheuristic algorithm for solving optimization problems. Knowledge-Based Systems, 2023, 259: 110011.
[31] Ikotun A M, Ezugwu A E, Abualigah L, et al. K-means clustering algorithms: A comprehensive review, variants analysis, and advances in the era of big data. Information Sciences, 2023, 622: 178-210.
[32] Cheng R, Jin Y. A competitive swarm optimizer for large scale optimization. IEEE transactions on cybernetics, 2014, 45(2): 191-204.
[33] Chen K, Zhou F, Yin L, et al. A hybrid particle swarm optimizer with sine cosine acceleration coefficients. Information Sciences, 2018, 422: 218-241.
[34] Feng X, Yu Y, Wang X, et al. A hybrid search mode-based differential evolution algorithm for auto design of the interval type-2 fuzzy logic system. Expert Systems with Applications, 2024, 236: 121271.
[35] Wu G, Mallipeddi R, Suganthan P N. Problem definitions and evaluation criteria for the CEC 2017 competition on

constrained real-parameter optimization. National University of Defense Technology, Changsha, Hunan, PR China and Kyungpook National University, Daegu, South Korea and Nanyang Technological University, Singapore, Technical Report, 2017.
[36] Abualigah L, Yousri D, Abd Elaziz M, et al. Aquila optimizer: a novel meta-heuristic optimization algorithm. Computers & Industrial Engineering, 2021, 157: 107250.
[37] Abualigah L, Diabat A, Mirjalili S, et al. The arithmetic optimization algorithm. Computer methods in applied mechanics and engineering, 2021, 376: 113609.
[38] Xue J, Shen B. A novel swarm intelligence optimization approach: sparrow search algorithm. Systems science & control engineering, 2020, 8(1): 22-34.
[39] Gharehchopogh F S, Gholizadeh H. A comprehensive survey: Whale Optimization Algorithm and its applications. Swarm and Evolutionary Computation, 2019, 48: 1-24.
[40] Yildiz A R, Abderazek H, Mirjalili S. A comparative study of recent non-traditional methods for mechanical design optimization. Archives of Computational Methods in Engineering, 2020, 27(4): 1031-1048.
[41] Eisinga R, Heskes T, Pelzer B, et al. Exact p-values for pairwise comparison of Friedman rank sums, with application to comparing classifiers. BMC bioinformatics, 2017, 18: 1-18.
[42] Dao P B. On Wilcoxon rank sum test for condition monitoring and fault detection of wind turbines. Applied Energy, 2022, 318: 119209.