Computers, Materials & Continua

Tech Science Press

Doi:10.32604/cmc.2025.060170

ARTICLE

Phasmatodea Population Evolution Algorithm Based on Spiral Mechanism and Its Application to Data Clustering

Jeng-Shyang Pan[1,2,3], Mengfei Zhang[1], Shu-Chuan Chu[2,*], Xingsi Xue[4] and Václav Snášel[5]

[1]College of Computer Science and Engineering, Shandong University of Science and Technology, Qingdao, 266590, China
[2]School of Artificial Intelligence, Nanjing University of Information Science and Technology, Nanjing, 210044, China
[3]Department of Information Management, Chaoyang University of Technology, Taichung, 41349, Taiwan
[4]Fujian Provincial Key Laboratory of Big Data Mining and Applications, Fujian University of Technology, Fuzhou, 350118, China
[5]Faculty of Electrical Engineering and Computer Science, VŠB-Technical University of Ostrava, Ostrava, 70833, Czech Republic

*Corresponding Author: Shu-Chuan Chu. Email: scchu0803@gmail.com

Received: 25 October 2024; Accepted: 27 December 2024; Published: 26 March 2025

ABSTRACT: Data clustering is an essential technique for analyzing complex datasets and continues to be a central research topic in data analysis. Traditional clustering algorithms, such as K-means, are widely used due to their simplicity and efficiency. This paper proposes a novel Spiral Mechanism-Optimized Phasmatodea Population Evolution Algorithm (SPPE) to improve clustering performance. The SPPE algorithm introduces several enhancements to the standard Phasmatodea Population Evolution (PPE) algorithm. Firstly, a Variable Neighborhood Search (VNS) factor is incorporated to strengthen the local search capability and foster population diversity. Secondly, a position update model, incorporating a spiral mechanism, is designed to improve the algorithm's global exploration and convergence speed. Finally, a dynamic balancing factor, guided by fitness values, adjusts the search process to balance exploration and exploitation effectively. The performance of SPPE is first validated on CEC2013 benchmark functions, where it demonstrates excellent convergence speed and superior optimization results compared to several state-of-the-art metaheuristic algorithms. To further verify its practical applicability, SPPE is combined with the K-means algorithm for data clustering and tested on seven datasets. Experimental results show that SPPE-K-means improves clustering accuracy, reduces dependency on initialization, and outperforms other clustering approaches. This study highlights SPPE's robustness and efficiency in solving both optimization and clustering challenges, making it a promising tool for complex data analysis tasks.

KEYWORDS: Phasmatodea population evolution algorithm; data clustering; meta-heuristic algorithm

1 Introduction

Clustering represents a method of unsupervised learning, and the original purpose of cluster analysis is to put physical objects with similar characteristics in people's lives together, grouping the samples in a dataset in such a way that the similarity of the samples in the same group is maximized and the similarity of the samples in different groups is minimized [1]. In the field of data analysis, clustering analysis, as one of the most representative methods, has been studied in depth by researchers. Traditional clustering methods primarily consist of partition-based, hierarchical [2], grid-based [3], and model-based algorithms [4], among others. In addition, there are density-based clustering algorithms [5]. The division-based c lustering is currently among the most commonly used clustering techniques, and its core idea is: for the dataset containing N samples,

the clusters are defined in advance to be L in number, and then the whole dataset is divided into L clusters by successively optimizing a certain objective as the division criterion until the end of the iteration.

Of these, the K-means algorithm is the most widely used partitioning method in practical applications [6,7]. Therefore, it is extensively applied in pattern recognition, data processing, model classification and other fields. With the progress of technology and the increased demand for large-scale data processing, the application of K-means algorithm also shows a trend of continuous expansion. Meanwhile, as clustering algorithms are studied more extensively, providing new possibilities for more efficient and accurate processing of various data types. Therefore, the K-means algorithm still holds an important position in today's data analysis field and is constantly evolving and developing, providing powerful tools and support for solving practical problems [8].

However, K-means has some limitations when performing clustering optimization, such as its results depend on the initial conditions, leading to locally optimal solutions. To address the challenge of local optima, a lot of research has been done in clustering. Researchers have increasingly turned their attention to harnessing the inherent principles of the natural world for computational purposes through the advancement of meta-heuristic algorithms. In the context of life sciences, they began to explore the swarm intelligence behavior exhibited by living organisms (e.g., chromosomes) and animal populations. This collective intelligence phenomenon has garnered interest from numerous scholars globally [9,10]. Drawing inspiration from the intelligence and behaviors found in nature, these algorithms offer innovative approaches to tackling complex problems [11].

Typical metaheuristic algorithms include Genetic Algorithm (GA) [12,13], Coyote Optimization Algorithm (COA) [14], Cuckoo Search (CS) [15], Differential Evolution (DE) [16,17], Harmony Search (HS) [18], Biogeography-Based Optimization (BBO) [19], Particle Swarm Optimization (PSO) [20–22], Grey Wolf Optimizer (GWO) [23,24], Whale Optimization Algorithm (WOA) [25,26], Butterfly Optimization Algorithm (BOA) [27], Archimedes Optimization Algorithm (AOA) [28], Harris Hawks Optimization (HHO) [29] and Ant Colony Optimization (ACO) [30,31]. These algorithms collectively fall under the umbrella term Swarm Intelligence Optimization Algorithm (SIOA). SIOA has many advantages, including easy to find potential solutions, simple parameter tuning, easy to implement, and more effective information exchange between individuals. However, there is no single SIOA that is universally effective for all optimization challenges, as each has its own strengths and weaknesses. Therefore, many researchers have worked on proposing a large number of novel SIOAs and improving the existing algorithms [32]. These improvements include hybridization of algorithms, optimization of parameter tuning strategies, and personalized algorithms designed for specific problems. The goal of these efforts is to continuously expand the applicability of SIOA to better address different types of optimization challenges and to promote the application and development of optimization algorithms in practice.

The Phasmatodea Population Evolution Algorithm (PPE) proposed by Song et al. [33] was inspired by evolutionary features observed in Phasmatodea populations. Since the algorithm was proposed, researchers have successfully applied it to a number of fields, demonstrating its competitiveness. For example, the PPE has been utilized to optimize downlink power allocation in fifth-generation heterogeneous networks, and it has also been applied to task scheduling optimization in cloud computing environments. These experimental findings indicate that the PPE population evolution algorithm holds significant promise [34,35].

We integrate PPE with K-means for optimization.This fusion can effectively utilize advantages such as the more powerful global search capability of PPE to boost the performance of K-means in global search. However, PPE also suffers from the challenge of being prone to local optimal solutions. Therefore, in this paper, we will further optimize PPE and apply it to the K-means clustering optimization problem. To address these issues, in this study, a Phasmatodea population evolution algorithm with a spiral mechanism (SPPE) is

designed, and SPPE also incorporates a balancing factor, which achieves good results in data clustering. The primary contributions outlined in this paper include:

1. To bolster the local search capability and foster population diversity, we integrated the Variable Neighborhood Search factor into our approach.

2. A position update model is introduced, featuring a novel position update formula and a spiral function designed to enhance the exploration capabilities of PPE.

3. Based on the fitness value, a proposed balancing factor is added to the PPE through the information guidance strategy although it can enhance the development efficiency of the PPE.

4. SPPE is evaluated against various algorithms on CEC2013, and the results demonstrate that SPPE performs well on most functions.

5. In the data clustering application, SPPE is assessed alongside several clustering techniques, and the experimental findings indicate that SPPE surpasses the other methods.

The subsequent sections of the paper are structured as follows: Section 2 presents related work, while Section 3 provides an exposition on the traditional PPE algorithm. Section 4 outlines the SPPE algorithm and CEC2013 experiments in detail. Section 5 conducts clustering experiments and compares them with existing methods to evaluate the performance of the SPPE algorithm. Section 6 gives conclusions.

2 Related Work

K-means partitions the dataset into mutually exclusive clusters based on specific criteria, making the data objects within the same cluster more similar. Assume k initial clustering centers in the sample set $Dx = dx_1, dx_2, \ldots, dx_n$, where n represents the number of samples. The cluster centers are represented as $c = c_1, c_2, \ldots, c_n$ and $d(Dx, c)$ denotes the distance between different data objects and the cluster centers, calculated using the Euclidean distance.

$$d(Dx, c) = \sqrt{\sum_{j=1}^{d} (dx_j - c_i)^2} \tag{1}$$

where d indicates the dimension of the data sample, dx denotes the data sample, dx_j is the value in the j-th dimension of dx.

The termination condition for K-means iteration is typically that the points in each cluster remain unchanged. Additionally, the new cluster centroid is determined by averaging all the points within the cluster, it can be judged according to the change of Sum of Square Errors (SSE), which is commonly used in mathematics as shown in Eq. (2):

$$SEE = \sum_{i=1}^{k} \sum_{dx \in c_i} |dis(dx, c_i)|^2 \tag{2}$$

Given that the position of the data object point dx remains constant, if the cluster center c_i also remains unchanged, the SSE value will not vary, indicating that the iteration can be concluded.

To overcome the limitations of K-means, extensive research has been conducted on clustering techniques. A DE method based on one-step K-mean clustering to cope with unconstrained global optimization problems, called clustering-based DE, was proposed by Cai et al. [36]. By fully leveraging the population information, one-step K-means clustering substantially enhances the performance of DE. Nayak et al. [37] introduced FA-K-means, a firefly-inspired K-means algorithm. The approach leverages the firefly algorithm's global search ability to perform efficient cluster analysis. The Enhanced Shuffled Bat Algorithm (EShBAT),

developed by Chaudhary et al. [38], is an improved variant of the Bat Algorithm. A key feature of EShBAT is that it divides the bat population into independent groups called memeplexes, where each group evolves independently using the Bat Algorithm, and it has achieved success in numerical optimization tasks. This study introduces a hybrid algorithm called HESB that combines K-means, K-medoids and EShBAT. This creates a diverse initial population for EShBAT, resulting in an effective clustering algorithm.

Huang [39] proposed a novel method for color image quantization that utilizes the Artificial Bee Colony (ABC) algorithm to choose N colors from M available options for the initial palette, followed by an accelerated K-means lgorithm. Additionally, a sampling process is introduced to reduce the computation time. García et al. [40] proposed a method called K-means combined with the Cuckoo Search algorithm. This method utilizes K-means for discretizing solutions and employs the CS algorithm for optimizing in the continuous domain. Pal et al. [41] combined the Black Hole for data clustering with K-means to achieve better results. The method utilizes some optimization outcomes from K-means to initialize a portion of the population, while the remaining portion is initialized randomly.

3 Principle of PPE

The PPE algorithm originated from the observation and study of the survival characteristics of Phasmatodea populations in natural environments and their population dynamics. In the PPE algorithm, the Phasmatodea population is defined as having two key attributes: population growth rate and population quantity. Its primary aim is to address the challenge of finding a globally optimal solution within a multidimensional space, where the optimal position of the Phasmatodea population is regarded as a candidate for the global optimum. The application of this algorithm benefits from a more profound understanding of the ecological properties of biological populations.

The representation of a Phasmatodea population in n-dimensional space is given by the expression $x = [x_1, x_2, x_3 \ldots x_n]$, where each dimension represents a natural characteristic of that population of Phasmatodea. Generate Np populations using Eq. (3).

$$x_i = Lb + (Ub - Lb) * rand \tag{3}$$

Two of the most significant attributes of Phasmatodea populations that best reflect their outcomes as influenced by environmental and other factors are the growth rate a and the population quantity p. Firstly, a takes values between 0 and 4, and secondly, p is determined using the following formula:

$$p_i = \frac{1}{N_p} \tag{4}$$

where N_p denotes the number of Phasmatodea populations. The evolutionary trend ev in PPE is mainly guided by the first L historical optimal solutions, which are stored into Hx.

$$L = \lfloor log(N_p) \rfloor + 1 \tag{5}$$

The formula for adjusting the position of the Phasmatodea population is as follows:

$$x^{t+1} = x^t + ev \tag{6}$$

$$p^{t+1} = p^t * (1 - p^t) * a^{t+1} \tag{7}$$

If the population shows improvement after updating its position compared to before, the population's ev can be divided into three parts. First, the population will approach the nearest historical optimal solution,

which reflects the self-determination property of the Phasmatodea population. Second, the population will maintain a relatively stable trend through path-dependent effects. Finally, the population may adjust the trend through mutation.

$$ev^{t+1} = (1 - p^{t+1})(s(Hx, x^t) - x^t * c - p^{t+1}(ev^t + m) \tag{8}$$

$$m = (Ub - Lb) * 0.2 \tag{9}$$

where $s(Hx, x^t)$ represents the closest solution to x^t stored in Hx and m denotes the variation of the population under certain natural conditions. The idea of simulated annealing is used after the population has updated its position when the population condition becomes less favorable than before. In other words, the suboptimal population will be accepted. The equation for updating the ev is as follows:

$$ev^{t+1} = rand * (s(Hx, x^t) - x^t) * c - st * B \tag{10}$$

$$st = (Ub - Lb) * 0.1 \tag{11}$$

where B is an n-dimensional random vector generated from a standard normal distribution. In addition to the effect that the optimal solution of a population may have on the change of its trend, the competition between populations with each other may also have an effect on its position. When the distance between two populations falls below a predetermined threshold G, they will engage in competition.

$$G = \frac{(Ub - Lb) * (M_{gen} + 1 - t)}{M_{gen}} * 0.1 \tag{12}$$

When the distance between x_i and x_j is below the G, the geographic locations of these two populations collide, which in turn triggers competition between the populations. In this case, the population's p and ev values are:

$$p_i = (1 - p_i - \frac{f(x_j)}{f(x_i)} p_j) * a_i * p_i + p_i \tag{13}$$

$$ev^{t+1} = ev^t + \frac{f(x_j) - f(x_i)}{f(x_j)} (x_j - x_i) \tag{14}$$

The core mechanism of PPE is built upon this foundation. First, convergent evolution emphasizes how environmental constraints guide populations toward locally optimal solutions, balancing local and global search. Second, path dependence highlights the influence of historical decisions on evolutionary trends, enabling efficient adaptation to changing environments while preserving successful past trajectories. Third, population mutation helps escape local optima, enhancing adaptability in dynamic or multi-modal landscapes. Fourth, the population size and competition model illustrates how resource limitations and competition drive diversity and improve performance, particularly in multi-modal optimization. Together, these mechanisms enable PPEA to balance exploration and exploitation dynamically, offering a robust framework for solving complex, high-dimensional, and dynamic optimization problems.

4 Improved PPE Algorithm: SPPE

4.1 Variable Neighborhood Search

Hansen et al. [42] proposed the Variable Neighborhood Search (VNS) algorithm, which initially defines a set of neighborhoods that are considered potential good solutions. The core concept of VNS is to systematically modify the neighborhood structures of multiple solutions during the local search process.

Under the condition of identical initial solutions, this process deepens and broadens the search space, thus improving the effectiveness of the local search. VNS focuses on locking the superior solutions in the solution space quickly by the fast transformation between different neighborhood structures.

By integrating the VNS strategy, SPPE systematically alters the neighborhood structures during the local search process. This enables the algorithm to lock high-quality solutions quickly and enhances its ability to perform deep searches in locally optimal regions. Such systematic modification of search patterns is a unique addition that sets SPPE apart from traditional metaheuristic algorithms.

In the PPE algorithm, when a particle falls into a local optimal solution, in order to search for the particle in a wider neighborhood, we need to extend the search for the direction of movement of the particle to find a more superior direction. Incorporating the key features of VNS into the PPE results in a powerful PPE algorithm. Another feature of the PPE algorithm with the introduction of VNS is to re-initialize the search for the locally optimal neighborhood. The pseudocode of the VNS algorithm is shown in Algorithm 1.

Algorithm 1: Pseudo-code for VNS

Input: Termination conditions, parameter settings

Output: Optimal program

1: **while** $k \leq kmax$ **do**
2: Generate a random x' from the k-th neighborhood structure *Nk (x)* of x.
3: The local optimum x'' should be found by a local search method with x' as the initial solution.
4: If this local optimum x'' is superior to the current one, update x to x''.
5: $k = k + 1$
6: **end while**

$Nk(x)$ represents the k-th neighborhood structure of x. By exploring broader neighborhoods when stuck in local optima, the algorithm intensifies the search when a better solution is found and diversifies the search when local exploration is exhausted. This systematic adjustment ensures robustness, enabling the algorithm to escape from local optima and discover globally superior solutions.

4.2 Spiral Mechanism

The second stage of PPE remains close to only the nearest optimal solution even when convergent evolution is ineffective. In this approach, position updates rely solely on previous movement direction. This will hinder the algorithm's convergence speed, so we propose adopting a spiral search mechanism. Unlike conventional search strategies that often rely on linear or random exploration, the spiral mechanism introduces a dynamic and expansive search trajectory. This approach allows the algorithm to explore the solution space more effectively, avoiding duplicate paths and improving global search capabilities. The spiral mechanism also helps maintain population diversity, which is crucial for escaping local optima.

This mechanism offers a broader range of exploration opportunities for the Phasmatodea, enabling it to better adjust its position through multiple search paths. Simultaneously, this mechanism fully utilizes the top k best positions from the existing Phasmatodea population, thereby enhancing the capability to discover the global solution throughout the optimization process. The spiral mechanism leverages a logarithmic spiral model to ensure diverse exploration paths. The position update formula is:

$$D = abs(Hx_L - x^t) \tag{15}$$

$$z = \exp(b * u) * \cos(u * 2 * \pi) \tag{16}$$

$$x^{t+1} = D * z + Hx_L \tag{17}$$

where b is the spiral shape constant, which is set to a value of 1 and z represents the spiral exploration factor. u denotes the path coefficient, which is a random number in $[-1, 1]$. The exponential term $\exp(b * u)$ allows the search radius to expand or contract dynamically, providing a fine balance between exploration and exploitation. Meanwhile, the cosine term $\cos(u * 2 * \pi)$ ensures variability in the search direction, enabling broader coverage of the solution space and improving the global search performance.

Individuals of Phasmatodea use a spiral advance strategy during search. Unlike straight lines, spiral advancement uses a straight line as the central axis, which makes the individual move in a spiral when moving towards the target individual. This strategy minimizes the possibility of generating duplicate individuals. The integration of a spiral mechanism into the position updating process enhances the Phasmatodea's ability to explore unknown regions, effectively improving the algorithm's global search performance.

4.3 Balancing Factor

We use the value of the fitness function to form a balancing factor, and this dynamic adjustment strategy enhances the algorithm's search effectiveness and efficiency across various search stages. This way, the SPPE will become more flexible, which can improve the algorithm's search efficiency. The balancing factor dynamically adjusts the exploration and exploitation tendencies of the algorithm based on fitness values. This mechanism ensures a more adaptive search process, allowing SPPE to maintain a fine balance between global and local optimization. It improves convergence speed and enhances the algorithm's robustness in different stages of the search.

$$w = \frac{1}{\exp\left(\frac{fit(x_i)}{sum(fit)}\right)} \tag{18}$$

Ensures that individuals with better fitness have a stronger influence during population updates. This mechanism reduces the premature convergence common in other algorithms by preventing excessive exploitation of local optima. The updated population parameter:

$$p^{t+1} = w * a^{t+1} p^t (1 - p^t) \tag{19}$$

The population quantity p^{t+1} incorporates this balancing factor to maintain diversity while focusing on promising areas of the search space. This balance between exploration and exploitation enhances the convergence rate while maintaining sufficient diversity to avoid stagnation in suboptimal regions. Introducing a balance factor into the process of updating the population quantity helps to fully utilize effective information within the population and enhancing its ability to escape local optimal solutions. Through this approach, significant improvements in optimization speed and accuracy can be achieved. Algorithm 2 outlines the pseudocode of the SPPE algorithm.

4.4 Complexity Analysis

The time complexity of the algorithm mainly depends on population initialization and iterative optimization. The population initialization time complexity of the SPPE algorithm is $O(p * d)$, where p is the population size, and d is the dimension of the solution space. In each iteration, the SPPE algorithm includes four key steps: population position updating in the PPE algorithm, the spiral mechanism, VNS, and balancing factor updates.

Among them, the population position updating and spiral mechanism both involve position updates, with a time complexity of $O(p * d)$; the balancing factor update only requires simple arithmetic operations, resulting in a time complexity of $O(1)$. The VNS involves checking multiple neighborhood structure. Assuming the number of neighborhood structures is k, the overall complexity of VNS is $O(k * d)$. In SPPE, the probability of triggering VNS in each iteration is 0.7, and three neighborhood structures are used ($k = 3$), so the complexity of VNS is $O(0.7 * 3 * d)$.

In summary, the total time complexity of each SPPE iteration is $3 * O(p * d) + O(0.7 * 3 * d) + O(1) = O(p * d)$. Considering the algorithm runs for I iterations, the total time complexity is $O(p * d * I)$. This complexity is equivalent to the standard PPE algorithm's time complexity of $2 * O(p * d * I) = O(p * d * I)$.

To further demonstrate the computational performance of SPPE, we analyzed its runtime under the same experimental conditions and compared it with other algorithms. The runtime results (in seconds) are as follows: SPPE: 490.20, PPE: 420.19, PSO: 130.38, HHO: 560.90, GWO: 140.67, AOA: 145.80, GA: 207.3, BOA: 422.86. Although the runtime of SPPE is slightly higher than its predecessor PPE due to the additional computational costs of the spiral mechanism and VNS, SPPE still demonstrates competitiveness compared to other algorithms. For instance, SPPE outperforms HHO in terms of runtime while offering better robustness and convergence. Although algorithms like PSO, GWO, and GA have shorter runtimes, SPPE's enhanced optimization capability proves that the moderate increase in computational cost is justified.

Similar to PPE, SPPE's space complexity is determined by population storage ($O(p * d)$) and historical best positions ($O(d)$). Auxiliary components like the spiral mechanism, VNS, and balancing factor require $O(1)$ space. Thus, the total space complexity of SPPE is $O(p * d)$.

In conclusion, compared to PPE, SPPE achieves significant improvements in robustness and convergence while maintaining a time complexity of $O(p * d * I)$. Additionally, its computational cost is comparable to other popular algorithms, making it an efficient solution for complex optimization problems.

Algorithm 2: Pseudo-code for the SPPE algorithm

Input: Number of iterations (M_It), number of dimensions (Dim), the size of a population (N_p)
Output: The optimal solution produced by each iteration ($best\_fit$)

Calculating the fitness function (*fit(x)*), set g_{best} and Hx
for $ite = 1 : M\_It$ **do**
  Use ev_i generate new population (x_{new})
  for $i = 1 : N_p$ **do**
    if $f(x_{new}) \leq f(x_{old})$ **then**
      if $rand < 0.7$ **then**
        $x = x_{new}$
      else
        VNS factor
      end if
    else
      if $rand < p$ **then**
        $x = x_{new}$
      else
        use Eq. (17) to update x
      end if
    end if

(Continued)

Algorithm 2 (continued)

```
        if The distance between the two populations is less than G then
            Update ev_i via Eq. (10)
        end if
    end for
end for
```

4.5 Experiments and Data Evaluation

In this section, Our experimental tests performed on a computer with an operating system of Windows 11 and software of MATLAB R2022a are tested using 28 benchmark functions from the CEC2013. Additionally, a more intuitive comparison of the algorithms was conducted through curve analysis of the fitness values. We compared the SPPE algorithm with the PPE algorithm, AOA algorithm, PSO algorithm, HHO algorithm, GWO algorithm, GA algorithm, and BOA algorithm. The parameter configurations for these algorithms are shown in Table 1.

Table 1: Parameter settings for the algorithms

Algorithm	Initial setting of parameter values
SPPE	$N_p = 20, a = 1.1$
PPE	$N_p = 20, a = 1.1$
PSO	$N_p = 20, c1 = 2, w = 0.9, c2 = 2$
HHO	$N_p = 20, E_0 : [-1, 1]$
GWO	$N_p = 20$
AOA	$N_p = 20, mu = 0.499, alpha = 5, mop_{max} = 1, mop_{min} = 0.2$
GA	$N_p = 20, crossoverrate = 0.8, mutationrate = 0.05$
BOA	$N_p = 20, powerexponent = 0.1, modularmodality = 0.01, probabilityswitch = 0.8$

To ensure fairness, each algorithm was subjected to 1000 function iterations with a total of 30 independent function runs, and the solution space range was set to [−100, 100]. We evaluated the algorithms in 10, 30, and 50 dimensions, recording the mean and variance of their results. Additionally, we conducted the Friedman test and obtained the Friedman rankings, which are displayed in Tables 2–4. In these tables, the bold data values represent the best results in each function. Observing the tables reveals SPPE's outstanding performance across all dimensions, with its efficacy progressively enhancing as the dimensions expand. We conducted a Wilcoxon rank-sum test at a significance level of 0.05 to compare the optimization results of SPPE with those of other algorithms. In the table, a "+" indicates that the results of the SPPE algorithm outperform those of its comparison algorithms, "=" signifies that the results are similar, and "−" indicates that the SPPE algorithm's results are inferior to those of its comparison algorithms. The results in the table indicate that the SPPE algorithm has a significant advantage in seeking optimal solutions compared to other algorithms, outperforming most of them. From the results presented in Tables 2–4, it can be observed that, in most cases, there is a statistically significant difference between the SPPE algorithm and the competing algorithms in the Wilcoxon signed-rank test. Furthermore, when such significant differences exist, the SPPE algorithm demonstrates a clear statistical advantage over the competing algorithms. This further validates the effectiveness of the SPPE's improvement strategies and its superiority under various test conditions.

Table 2: Results on CEC2013 when D = 50

		SPPE	PPE	PSO	HHO	GWO	AOA	GA	BOA
F1	Mean	**−1397.031**	−1392.3803	185002.287	−1075.3865	14565.7407	73802.181	127178.025	78480.9247
	Std	1.52231359	1.87025254	14335.2676	115.039877	6247.86825	6766.04324	22462.2745	5065.11436
F2	Mean	**33470196**	49359211.9	7684780888	101861481	197090738	2095202977	1371471948	3593281065
	Std	9220931.05	10196846.2	2181278382	28629163.6	68864643.2	713088786	718416525	1097074247
F3	Mean	**7.946E+09**	1.6342E+10	8.1344E+21	4.4269E+10	7.7472E+10	4.5242E+13	4.8144E+13	1.9736E+21
	Std	3832176835	4315434996	2.7822E+22	1.3705E+10	2.409E+10	4.768E+13	1.2023E+14	4.9218E+21
F4	Mean	**59898.742**	68442.3282	567184.855	83951.2612	181766.83	78886.4315	255260.778	104733.184
	Std	14128.1983	11268.9706	864242.131	8293.48939	49152.9014	7957.4943	40951.0067	16746.6273
F5	Mean	**−977.1497**	−868.84949	108129.293	−622.31207	3432.38209	22766.4833	41779.4327	26517.5611
	Std	8.80260199	21.1638953	32477.7201	148.041728	2157.60991	8425.6295	21521.0768	6260.27532
F6	Mean	**−783.4757**	−678.90512	39629.7048	−601.274	91.209867	7453.87871	11035.0221	11074.0168
	Std	67.8895299	46.2122903	6246.01279	51.7166139	409.206544	1806.78546	4141.59723	1864.60777
F7	Mean	**−683.9244**	−643.93638	19670171.1	−128.87343	−617.12795	3144.10469	4532.46577	70413.3942
	Std	19.1850276	36.715322	26437518.5	642.762823	34.8732922	3814.38938	8872.14813	66720.376
F8	Mean	−678.77778	−678.77073	−678.63018	**−678.7882**	−678.78087	−678.73145	−678.71358	−678.76157
	Std	0.03801589	0.04703793	0.04571112	0.05639984	0.03528534	0.06117103	0.05522852	0.03961746
F9	Mean	**−544.1318**	−537.23039	−516.14265	−528.25055	−538.96987	−526.95541	−516.26608	−525.04115
	Std	6.02248468	4.7813727	2.06001674	2.81573269	4.94967593	3.32320995	2.73056693	2.31253669
F10	Mean	**−433.4597**	−326.05723	29713.5107	17.9333812	1703.57388	10988.0495	13722.2196	13966.3598
	Std	17.6855861	32.1762948	5806.13209	130.267091	443.585143	2078.08514	4498.0974	1588.61181
F11	Mean	**−171.7039**	46.2665481	2593.27018	307.516825	135.192083	715.264733	1488.87339	770.733991
	Std	54.0574552	61.2536097	368.697886	47.1457228	61.3413463	99.3690661	290.078229	55.0311892
F12	Mean	**63.81003**	323.199149	2472.57247	772.969022	368.812632	894.014667	1554.74072	970.666194
	Std	65.05993	75.7097871	225.61595	121.549847	92.6144898	115.619415	393.477109	54.716409
F13	Mean	**375.87743**	584.072799	2342.87226	972.596659	513.657543	976.07423	1571.19411	1062.8686
	Std	87.2377791	74.3250426	290.270391	111.179539	61.7859329	99.8449376	295.503653	61.4049712
F14	Mean	**3777.4713**	5355.63816	14866.8554	8539.07544	12545.3459	13667.4793	16099.9081	15088.8423
	Std	585.630383	629.811333	366.089185	1345.19689	1882.98419	1319.35043	1190.07123	555.083235
F15	Mean	**9147.5285**	9376.14745	16942.829	12220.9457	14000.2747	14912.2175	15664.2267	15729.822
	Std	1244.245	986.270376	521.128808	1043.27375	1370.18553	581.06236	1012.76301	402.113079
F16	Mean	203.12866	**202.23749**	206.186936	203.309516	204.204309	203.929069	204.937161	204.488055
	Std	1.00426266	0.47438864	0.6012589	0.56215979	0.33499587	0.60699248	0.55669978	0.34779578
F17	Mean	**580.65172**	916.650199	6024.06614	1640.46786	1177.96135	1670.95101	4440.89869	1684.53256
	Std	44.3342113	78.2407373	619.778393	85.4182794	137.660427	76.1355154	691.705435	58.8324778
F18	Mean	**884.38638**	1236.85237	6026.84717	1753.37179	1312.54709	1779.23097	4669.90236	1810.61017
	Std	85.0075358	121.077595	574.819326	82.2321586	117.610611	53.5697894	654.71418	56.6835969
F19	Mean	**520.71666**	551.440053	40246615.2	597.576574	57253.5774	1233368.41	3235044.35	1504445.77
	Std	4.83743393	12.4091781	18314484.6	16.9815502	57675.6733	501626.709	2762254.45	409047.805
F20	Mean	**624.31527**	624.487112	625	624.656622	624.431965	624.817117	625	624.991173
	Std	0.80004538	0.28169586	1.558E-11	0.19461768	0.43476846	0.18434554	4.9393E-11	0.04613965
F21	Mean	1631.56879	**1631.4833**	12901.2712	2332.53986	4461.0082	5191.02774	10956.1369	5285.91961
	Std	238.136896	204.006072	1250.68549	638.434051	235.951334	66.5265499	1569.66363	64.5351444
F22	Mean	**6031.9214**	9831.42722	17749.1567	12148.7788	15191.8692	15492.0153	18694.2786	17562.6118
	Std	645.404482	1156.17559	397.647726	957.112956	1883.82831	628.415868	709.505769	413.192302
F23	Mean	**12167.138**	12928.6642	18468.2023	14846.6797	16080.8154	17239.3532	18096.3287	17727.8478
	Std	1617.40332	1263.48795	432.502109	1135.7741	1234.60957	408.426477	622.207752	550.616801
F24	Mean	**1359.1297**	1405.31287	1945.07779	1481.79859	1383.41674	1713.36065	1771.69551	1930.46488
	Std	17.0420667	19.2520903	187.275686	39.4155415	14.7712894	260.022375	138.292989	191.135115
F25	Mean	1542.91587	1614.75213	1661.48507	1610.19171	1532.01606	**1527.6798**	1797.99176	1614.10928
	Std	19.0160363	28.0265683	2.73691037	45.1468577	11.612587	55.245384	56.8220741	21.6564751
F26	Mean	1614.53643	**1589.4397**	1948.01313	1630.65059	1618.30876	1685.53768	1745.46582	1676.26455
	Std	85.9657819	101.057302	206.064726	125.646061	92.703452	68.4739306	75.8509378	69.8651564
F27	Mean	**3130.7868**	3482.971	5319.40901	3821.80651	3275.49318	4031.72168	4331.92741	4860.70669
	Std	146.644759	145.015878	680.066496	196.0415	128.290402	230.941712	250.692381	547.320462
F28	Mean	**2561.171**	7913.5416	19227.2095	11175.057	6741.81241	11785.9689	15078.7364	11896.9674
	Std	1523.48062	939.568388	2261.86343	742.913113	787.023501	777.147475	1475.40452	806.038848

(Continued)

Table 2 (continued)

	SPPE	PPE	PSO	HHO	GWO	AOA	GA	BOA
+ / = / −		25/0/3	28/0/0	25/0/3	26/0/2	28/0/0	28/0/0	27/0/1
Friedman rank	1	2	8	4	3	5	7	6

Table 3: Results on CEC2013 when D = 30

		SPPE	PPE	PSO	HHO	GWO	AOA	GA	BOA
F1	Mean	**−1399.9231**	−1399.6391	106144.547	−1360.5568	5425.18677	47238.7312	42111.39	55933.3301
	Std	0.04358879	0.1340234	12685.0444	8.45560904	3459.96877	6111.04593	13032.9454	4828.35335
F2	Mean	**18851148.7**	19182860.6	3206702353	53340866.9	92842916.5	625662787	487997514	1528066935
	Std	4326940.05	4363869.84	1161914233	24206009.8	43814084.4	260393616	300997838	733436346
F3	Mean	**2644294369**	5224945688	9.5429E+23	3.3583E+10	4.3147E+10	2.7492E+17	1.1984E+14	1.7891E+21
	Std	2093693686	2319529109	5.1914E+24	1.7143E+10	2.5312E+10	6.0818E+17	6.2457E+14	3.4466E+21
F4	Mean	**38713.9426**	42664.4533	670355.516	52158.2556	108569.953	54217.3218	168641.551	64700.713
	Std	10357.8909	9106.07608	1331622.26	3979.22965	23059.7746	5309.09511	35172.2841	3631.26338
F5	Mean	**−998.99385**	−986.64799	94285.3268	−770.88257	1806.5044	25968.0108	17998.6613	39775.4203
	Std	0.6493884	10.1991097	58038.8469	88.9645519	1440.59453	11468.3562	10154.608	13462.7215
F6	Mean	**−822.93244**	−790.97756	27494.1092	−715.91887	−442.58198	7772.94401	3044.19079	15230.5877
	Std	21.6463724	30.3148666	7389.90373	44.7419747	225.021717	2433.46687	2712.57476	2582.84049
F7	Mean	**−706.95212**	−670.24458	158067798	52534.6154	−631.28169	769252.762	10581.7676	6654785.81
	Std	27.4245767	26.7265997	321816864	161745.36	39.9171103	2385412.11	25150.2659	10227890.2
F8	Mean	−678.93522	**−678.94375**	−678.74731	−678.9903	−678.93659	−678.89571	−678.87775	−678.92483
	Std	0.03985151	0.04535341	0.07606742	0.05854216	0.05075957	0.07145631	0.06197968	0.04177734
F9	Mean	**−570.62564**	−567.62071	−554.39387	−560.20661	−565.69042	−559.76487	−553.30121	−558.08407
	Std	2.93597517	2.94023572	1.55244671	2.91978022	5.26951222	1.81242385	1.76860594	1.4634282
F10	Mean	**−487.91465**	−482.60962	15137.4292	−331.21397	664.180982	6306.92782	5232.98964	9937.3204
	Std	4.1378444	7.51481174	3795.17027	80.1249441	348.752358	1194.87634	2482.19782	1073.3585
F11	Mean	**−301.79945**	−162.1433	1340.14442	38.1245532	−95.289031	353.700147	243.776741	493.99544
	Std	18.9269736	41.7627922	384.576069	69.9044285	42.2617693	91.942204	241.301041	76.6889971
F12	Mean	**−140.76935**	66.0885277	1427.36368	355.601558	32.1912768	407.670511	450.380647	551.653212
	Std	44.9401219	65.3941402	296.562194	104.83889	61.181121	109.231719	210.358207	94.6888988
F13	Mean	**51.8584486**	212.405961	1458.23435	515.051765	142.526449	486.646139	567.936076	659.266532
	Std	61.4526264	58.2394442	263.152504	99.2285326	49.6558851	74.1269711	165.128688	74.8064214
F14	Mean	**1714.13076**	2249.28697	9227.63019	4100.31796	6999.27851	7145.69503	7940.9627	8269.91908
	Std	541.097164	466.105778	400.441588	662.804298	1204.55891	400.402047	805.945241	346.194669
F15	Mean	**4328.25573**	4546.17527	9404.71619	5535.38853	7452.11246	8007.72505	8312.98067	8215.70369
	Std	820.505572	553.073706	364.704884	697.496053	1039.69831	341.893335	614.264292	371.782324
F16	Mean	202.110149	**201.331159**	204.908314	202.571682	203.283987	203.238081	204.040661	203.301986
	Std	0.83168363	0.4353729	0.8585207	0.46704164	0.38920196	0.41539335	0.48893782	0.42546125
F17	Mean	**426.955646**	516.946326	3324.70135	1104.95845	724.720633	1114.23937	1971.61295	1173.12143
	Std	24.1340927	43.7839829	385.611349	92.801311	88.2484518	104.677109	458.290051	60.8081622
F18	Mean	**614.767107**	755.030829	3301.07182	1236.67646	843.60875	1225.59116	2041.69776	1253.95878
	Std	48.7104414	60.2380404	369.887583	87.1482253	71.6549352	104.38952	351.439781	52.9599057
F19	Mean	**507.542776**	517.264248	17969440.6	547.147404	13470.212	655437.739	337173.838	774042.33
	Std	1.85284648	5.16327954	9429165.14	9.3427267	20155.5887	273399.69	701485.661	271936.94
F20	Mean	**614.85548**	614.879175	615	614.969467	614.991348	614.910671	615	615
	Std	0.35368431	0.22777215	0.00323264	0.10614674	0.02114589	0.17503164	5.5865E-10	9.8426E-10
F21	Mean	1051.84224	**1050.41183**	7106.16898	1136.58973	2426.16548	3126.0113	4821.56614	3258.57671
	Std	73.1619711	75.6882402	907.241367	58.5187011	317.643	78.4127361	955.846409	79.8859659
F22	Mean	**3058.46514**	4257.13154	10801.6197	6900.16146	8351.01189	9370.46363	10003.7943	9836.61216
	Std	568.50808	723.941672	452.534624	858.014308	1246.37965	441.456095	742.33992	282.61895
F23	Mean	**5866.23393**	6980.20592	11138.6294	8066.30376	9036.98849	9619.65452	9867.78566	9804.03233
	Std	942.782076	872.856642	456.092274	857.658076	1054.93878	513.426375	807.122595	345.741991
F24	Mean	**1277.99979**	1297.15594	1746.84392	1365.90846	1292.66426	1404.53838	1493.93297	1479.88698
	Std	15.4384825	14.5898342	199.085101	66.1671419	9.13490859	58.3076135	123.164945	95.7528126
F25	Mean	**1410.01033**	1457.61372	1493.69632	1459.86125	1416.84254	1417.91691	1566.22095	1475.12301
	Std	14.8799132	20.9713269	8.65564105	22.2285661	8.53614633	19.8138092	35.3490149	24.4406925

(Continued)

Table 3 (continued)

		SPPE	PPE	PSO	HHO	GWO	AOA	GA	BOA
F26	Mean	**1502.95786**	1517.34589	1693.50666	1581.12264	1524.14731	1552.56268	1628.69313	1569.40235
	Std	85.3594315	90.3042796	109.947248	72.0892356	76.5700995	66.4644816	104.587485	73.7942564
F27	Mean	**2341.09455**	2525.93975	3388.10431	2741.84945	2448.81361	2822.63572	2922.18775	3205.25199
	Std	92.310362	95.92028	248.721887	95.4005448	86.4270953	142.451467	83.474519	242.2879
F28	Mean	**2086.62036**	5147.12838	11667.3386	6617.24961	4285.56267	6583.83092	8756.94067	7435.62314
	Std	850.032482	910.132122	1676.25728	702.208675	671.426669	467.024219	1372.08981	563.206872
	+ / = / −		24/0/4	28/0/0	27/0/1	26/0/2	27/0/1	27/0/1	27/0/1
Friedman rank		1	2	8	4	3	5	6	7

Table 4: Results on CEC2013 when D = 10

		SPPE	PPE	PSO	HHO	GWO	AOA	GA	BOA
F1	Mean	**−1400**	−1400	16424.772	−1399.527	−1065.479	4282.5059	−678.3268	8585.8291
	Std	2.269E-05	0.0002776	6280.2412	0.1892219	528.07565	1821.6114	209.49352	2851.4525
F2	Mean	**570123.35**	581059.98	177077297	4767514.8	7574418.4	8373027.6	31540251	32979713
	Std	444688.99	326670.61	128330624	2057881.5	3758814	2716702.1	21446365	21774051
F3	Mean	**198862889**	294867946	1.13E+14	1.165E+09	2.348E+09	4.972E+09	1.195E+10	1.519E+14
	Std	474208651	327109190	4.423E+14	2.986E+09	2.268E+09	3.364E+09	7.77E+09	8.121E+14
F4	Mean	4610.3533	**3319.4097**	582587.5	12160.828	33200.4	8398.1139	45603.14	15245.087
	Std	2584.808	2214.8809	1053961.9	2150.7334	12621.115	3014.1627	20606.29	2865.8461
F5	Mean	−998.7058	**−999.9938**	8551.3996	−998.2598	−796.757	1666.8088	−338.9682	5358.6123
	Std	7.0727536	0.0034221	4347.545	2.9162481	215.35838	1916.7523	663.07184	3241.7697
F6	Mean	**−886.7198**	−863.1054	938.54304	−852.0109	−843.8212	−621.4268	−414.5304	−38.93042
	Std	24.089371	32.610924	866.06956	35.835428	35.984056	129.33724	106.01236	296.83482
F7	Mean	**−769.7513**	−756.0424	3722.2288	−639.3495	−742.3379	−710.3642	−468.5904	6637.0784
	Std	19.694547	26.349078	10974.266	109.58718	25.987011	21.898113	236.13317	12040.281
F8	Mean	−679.5125	−679.5517	−679.1831	**−679.5916**	−679.5468	−679.5249	−679.4893	−679.5503
	Std	0.0860187	0.0981662	0.160564	0.0951949	0.0250958	0.0989505	0.0677557	0.1051814
F9	Mean	**−594.6924**	−593.6155	−587.0309	−591.352	−593.0222	−591.2128	−587.8901	−591.2671
	Std	1.5596603	1.5071589	1.0380978	1.4842474	1.667648	1.2604298	1.0845926	0.7937569
F10	Mean	**−499.204**	−498.5972	1668.3104	−495.0345	−432.1961	−11.47359	−153.8978	761.12257
	Std	0.6594964	0.9846833	882.60445	4.1058259	63.921026	194.66076	70.282888	366.41758
F11	Mean	**−396.3653**	−395.0034	−105.6964	−342.9767	−359.8746	−316.3936	−157.7912	−253.6183
	Std	2.1229356	3.0791158	80.011381	25.040599	15.670916	24.542001	56.124692	23.596046
F12	Mean	**−275.6537**	−254.0874	−23.03635	−188.7999	−254.5355	−205.9597	−104.2751	−172.8273
	Std	13.522706	24.741517	74.925996	33.258927	15.598271	26.666662	40.036958	24.432186
F13	Mean	**−165.0595**	−150.2625	87.054396	−76.83841	−145.4304	−109.2314	−59.36277	−61.84365
	Std	12.618306	15.155859	76.879171	33.474822	16.222556	23.997882	23.324328	26.595076
F14	Mean	**194.75875**	281.63346	2403.2428	788.92441	1165.5675	1295.5057	601.78193	1741.2031
	Std	179.24065	138.8978	215.9695	238.92372	497.33864	197.27225	307.86926	174.70883
F15	Mean	**946.32516**	1005.3311	2542.1246	1078.3739	1489.5285	1428.4334	1536.5351	1815.8467
	Std	290.85958	236.45737	313.14825	329.85469	429.36489	213.90082	324.63645	246.8766
F16	Mean	200.7893	**200.47465**	203.18249	200.96704	201.59308	201.13698	201.67583	201.528
	Std	0.3988056	0.2359898	0.630261	0.2710986	0.2684241	0.3314584	0.3863822	0.336939
F17	Mean	316.1828	**316.01023**	786.05318	398.34045	357.56808	390.62909	412.00752	423.56367
	Std	2.8926237	2.7407037	117.78932	25.242059	12.41579	26.723466	36.398796	20.538157
F18	Mean	**434.01739**	438.05797	903.14941	523.04955	462.72497	492.23559	524.7113	532.57158
	Std	15.519585	11.939926	117.11392	33.123398	13.203751	21.153622	39.144237	14.873761
F19	Mean	**500.69816**	501.14748	419392.51	507.51581	554.8706	7398.6055	511.34044	17577.993
	Std	0.2511203	0.6710526	552868.93	2.3078968	268.84627	7929.8113	4.6144139	12229.22
F20	Mean	**603.38003**	603.55616	604.89749	604.12577	603.79699	604.05429	604.94755	604.77493
	Std	0.461087	0.6268992	0.1689292	0.405238	0.3965978	0.3157149	0.1202624	0.2603904
F21	Mean	**1090.189**	1100.1939	2059.6276	1090.7241	1117.5904	1213.5205	1260.728	1332.2941
	Std	54.798871	4.751E-05	388.76581	38.209262	25.01492	42.530861	89.10182	63.177437
F22	Mean	**1139.9552**	1343.117	3529.4318	2071.8344	2454.7889	2657.7495	2565.0545	2975.9783
	Std	205.49582	263.86667	210.99042	373.39409	608.31481	194.66072	493.26752	202.22631
F23	Mean	**2079.9068**	2380.2709	3701.2218	2637.9849	2620.7698	2878.3788	3002.1244	3150.724

(Continued)

Table 4 (continued)

		SPPE	PPE	PSO	HHO	GWO	AOA	GA	BOA
	Std	415.75503	330.93998	214.50147	322.04511	454.56966	260.77083	389.44411	184.08688
F24	Mean	**1214.2757**	1223.9446	1281.1252	1228.9307	1221.3674	1230.157	1245.7107	1204.4938
	Std	18.908776	27.202298	46.370267	5.0190718	5.1630693	5.347367	11.08615	24.664811
F25	Mean	**1313.236**	1320.2569	1351.1963	1330.5068	1319.991	1326.4939	1343.62	1317.7469
	Std	6.5427338	6.9356803	6.3674059	5.8544835	4.277398	3.8728487	4.692356	22.452396
F26	Mean	**1373.2004**	1398.7698	1604.4618	1494.2224	1382.6981	1423.2073	1483.8992	1406.6081
	Std	73.273468	72.4976	120.76685	61.548318	28.767885	27.823754	58.786645	28.395067
F27	Mean	**1699.7103**	1712.4717	2564.4012	1974.6895	1837.0272	1907.9407	2044.4832	2030.5317
	Std	67.687327	61.615282	269.91913	134.53454	77.422618	70.323808	83.907654	83.89589
F28	Mean	**1929.5398**	2046.0716	3321.7786	2296.4772	1969.5118	2285.7285	2478.8937	2371.5291
	Std	166.15661	230.19883	262.0526	189.52659	211.89573	65.933934	184.58304	143.20777
	+ / = / −		20/0/8	28/0/0	27/0/1	27/0/1	27/0/1	27/0/1	26/0/2
Friedman rank		1	2	8	4	3	5	7	6

Finally, we illustrate the convergence curves of the SPPE and other algorithms in Figs. 1 and 2 for a dimensionality of 50. It is clear that the SPPE algorithm demonstrates superior convergence in F1, F6, F10, F12, F21, F27, and F28 compared to the other algorithms, as it reaches the optimal value more rapidly. The substantial advantage exhibited by the SPPE algorithm over the others underscores its superior capability in exploring and evading local optima. In functions F9, F15, F22, and F23, while SPPE may not initially surpass PPE and GWO, its curve continues to decline while PPE converges. This suggests that SPPE retains significant potential in the later stages of algorithmic exploration. In summary, SPPE's capability to avoid local optima and effectively find the global optimum has seen significant enhancement following the implemented improvements.

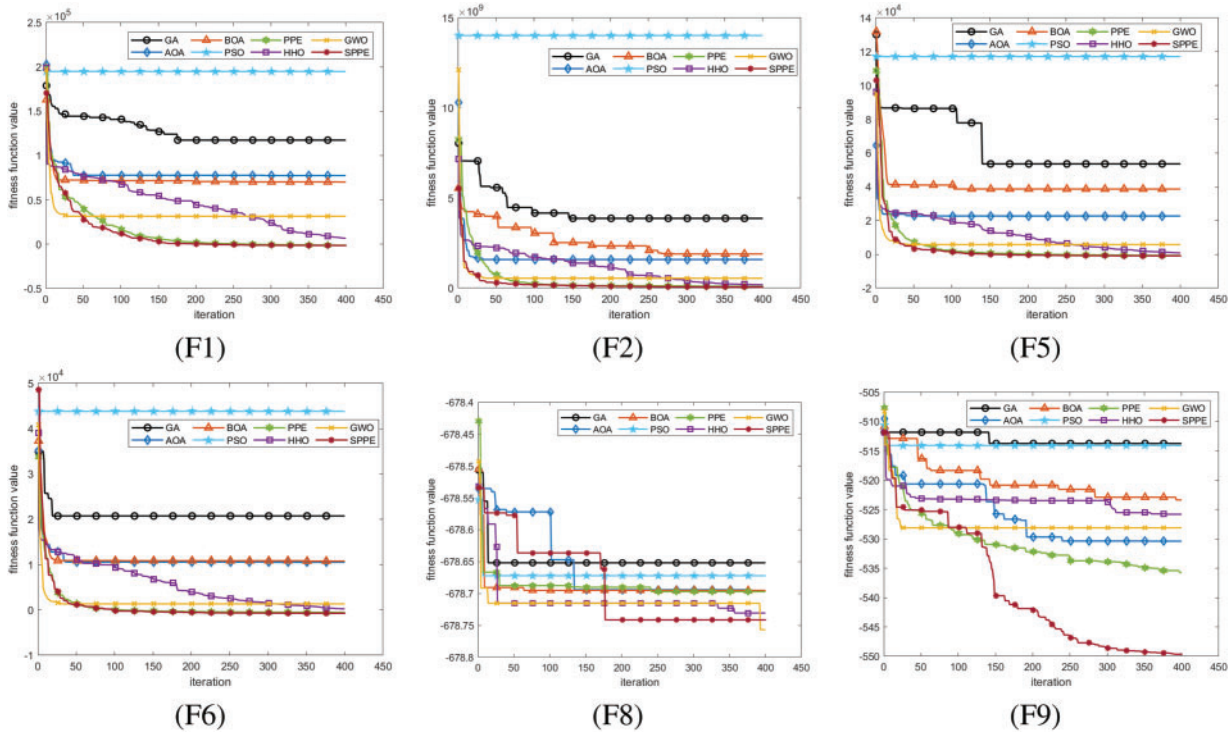

Figure 1: (Continued)

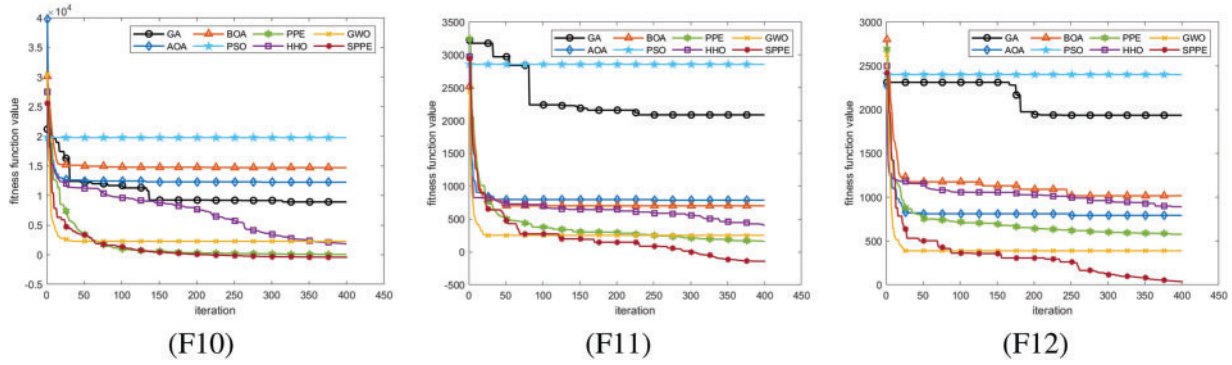

Figure 1: Partial convergence curves of CEC2013 unimodal functions and basic multimodal functions

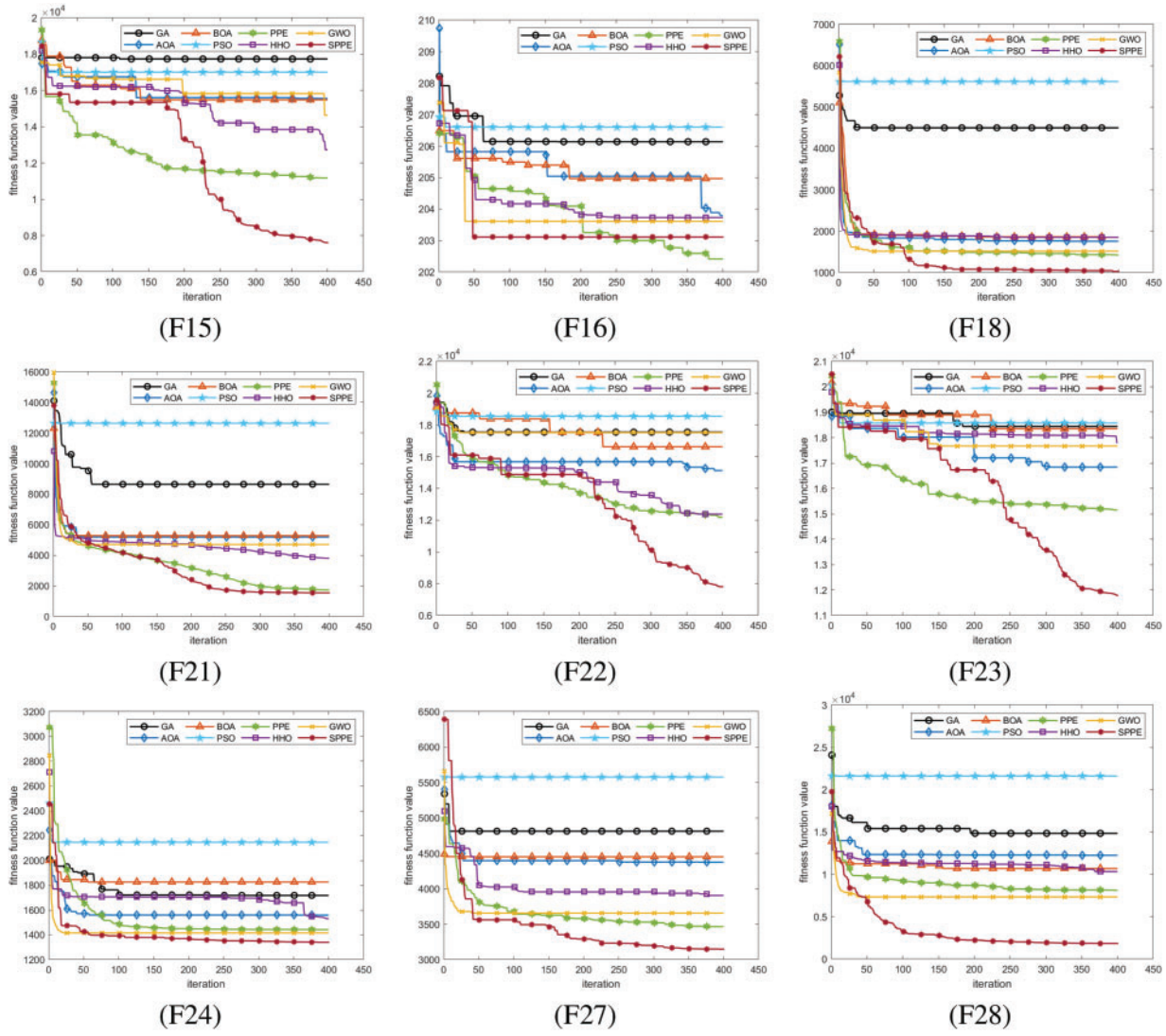

Figure 2: Partial convergence curves of CEC2013 basic multimodal functions and composition functions

4.6 Ablation Study

In order to verify the effectiveness and impact of the improved strategy proposed in this paper on the performance improvement of PPE algorithm, we conducted the following experiments: SPPE represents the PPE algorithm that combines multiple strategies, while SPPE1, SPPE2, and SPPE3 represent the PPE algorithm that uses spiral mechanism, VNS, and balancing factor to optimize a single strategy, respectively. Through these ablation experiments, we aim to evaluate the specific effects of each improvement strategy on the performance of PPE algorithms. The results are shown in Table 5. From the comprehensive evaluation of mean and standard deviation, the three single-strategy optimized PPE algorithms perform significantly better than the original PPE algorithm in 25 out of 28 functions, which indicates that the proposed improvement strategies effectively enhance the performance of the algorithms.

Table 5: Experimental results of SPPE with single-strategy improvement algorithm

		PPE	PPE1	PPE2	PPE3	SPPE
F1	Mean	−1392.3803	**−1396.462**	−1303.5212	−1389.7405	−1397.0307
	Std	1.87025254	0.23013516	29.4120828	3.5413881	1.52231359
F2	Mean	49359211.9	**33704868**	51551544.1	47810856.6	33470195.8
	Std	10196846.2	9463142.1	17660454.9	13751161.8	9220931.05
F3	Mean	1.6342E+10	**1.031E+10**	2.9533E+10	1.5673E+10	7946038298
	Std	4315434996	6720820089	1.3205E+10	6610536731	3832176835
F4	Mean	68442.3282	**66254.694**	88152.2155	75541.9807	59898.7422
	Std	11268.9706	16420.5557	17278.2507	9228.55691	14128.1983
F5	Mean	−868.84949	**−961.0164**	−886.25335	−875.61977	−977.14966
	Std	21.1638953	12.9062241	16.7144081	21.8468404	8.80260199
F6	Mean	−678.90512	**−775.2217**	−670.23805	−661.41531	−783.47575
	Std	46.2122903	38.4117615	46.4295207	52.6839515	67.8895299
F7	Mean	**−643.9364**	−627.00925	−630.63246	−615.39991	−683.92441
	Std	36.715322	19.2723848	42.9903827	53.4524997	19.1850276
F8	Mean	−678.77073	−678.77274	−678.77174	**−678.7728**	−678.77778
	Std	0.04703793	0.04433078	0.03344835	0.04203007	0.03801589
F9	Mean	**−537.2304**	−523.61471	−537.15063	−536.76342	−544.13175
	Std	4.7813727	4.26804068	5.37376746	4.12144016	6.02248468
F10	Mean	−326.05723	−302.50696	**−374.606**	−328.32742	−433.4597
	Std	32.1762948	20.863925	55.5346043	48.6980097	17.6855861
F11	Mean	46.2665481	**−20.95406**	38.4231372	79.9396773	−171.70395
	Std	61.2536097	75.8146426	61.0600545	64.8352922	54.0574552
F12	Mean	323.199149	**72.112184**	580.51846	356.939505	63.8100303
	Std	75.7097871	73.6053483	151.905962	79.1251987	65.05993
F13	Mean	584.072799	**393.52317**	731.231849	612.864022	375.877432
	Std	74.3250426	66.0210656	149.247723	122.160121	87.2377791
F14	Mean	5355.63816	4418.70549	**3782.682**	5635.34419	3777.47131
	Std	629.811333	848.035828	724.225729	783.526432	585.630383
F15	Mean	9376.14745	**9211.6369**	10258.2517	9308.05434	9147.52853
	Std	986.270376	1010.25287	1128.36272	877.503801	1244.245
F16	Mean	202.237491	204.910709	202.461706	**202.0707**	203.12866
	Std	0.47438864	0.99595074	0.58988355	0.83854705	1.00426266
F17	Mean	916.650199	643.302491	**624.29592**	954.386442	580.651717

(Continued)

Table 5 (continued)

		PPE	PPE1	PPE2	PPE3	SPPE
	Std	78.2407373	47.697251	11.1636177	88.0247353	44.3342113
F18	Mean	1236.85237	993.898848	**991.38011**	1264.64669	884.386379
	Std	121.077595	91.7983123	95.870048	111.05549	85.0075358
F19	Mean	551.440053	524.039135	**522.02506**	550.310798	520.716659
	Std	12.4091781	6.54767867	0.67873647	8.35304328	4.83743393
F20	Mean	624.487112	624.441664	**623.6761**	624.662364	624.315267
	Std	0.28169586	0.44744773	0.18838842	0.30372005	0.80004538
F21	Mean	1631.48333	**1627.6339**	1780.23537	1635.78445	1631.56879
	Std	204.006072	275.307293	225.15921	266.010779	238.136896
F22	Mean	9831.42722	**7470.4985**	9472.81515	9620.53571	6031.92143
	Std	1156.17559	1358.8164	1184.64244	1258.37074	645.404482
F23	Mean	12928.6642	**11692.156**	13792.1896	12827.6906	12167.1377
	Std	1263.48795	1255.61092	1183.24549	1336.47987	1617.40332
F24	F24	1405.31287	**1364.0946**	1409.91489	1406.97327	1359.12975
	Std	19.2520903	21.6455686	25.6354095	23.6300985	17.0420667
F25	Mean	1614.75213	**1554.0337**	1660.48712	1643.61499	1542.91587
	Std	28.0265683	21.9751805	49.743744	36.1297213	19.0160363
F26	Mean	**1589.4397**	1602.73696	1600.04149	1625.81634	1614.53643
	Std	101.057302	92.3364841	120.720005	91.0900627	85.9657819
F27	Mean	3482.971	3151.16755	3577.04168	**3141.7961**	3130.78682
	Std	145.015878	118.969756	197.293298	170.077039	146.644759
F28	Mean	7913.5416	**2777.4587**	6967.5306	7890.6452	2561.17104
	Std	939.568388	1125.40079	2721.22281	663.937736	1523.48062

Note: In the table, bold entries indicate the optimal values.

On unimodal functions and composition functions, SPPE1 reaches the optimal solution among the 4 algorithms, which is superior to other single-strategy optimization PPE algorithms and similar to the data obtained by the SPPE algorithm. This indicates that the spiral mechanism optimization strategy has significant effect in improving the performance of the algorithm on single peaked functions, especially in enhancing the algorithm's ability to find the optimal solution.

On basic multimodal functions, SPPE2 and SPPE3 perform comparably on functions F7 F10, but on other functions, the notation of SPPE2 outperforms most of the other PPE algorithms optimized by a single strategy. Therefore, this verifies that the VNS optimization strategy is significantly effective in improving the algorithm's global exploration and local exploitation.

The experimental data of SPPE3 on functions F8, F16, F27 outperforms other improved algorithms. Although SPPE3 has no obvious shortcomings in improving the algorithm's optimization searching ability or local exploitation ability compared to SPPE1 and SPPE3, it helps to balance the algorithm's performance in a comprehensive way.

In the above experimental data, the SPPE algorithm is less than or equal to other improved algorithms except in 4 functions. It shows that the fusion of multiple strategies makes the PPE optimization algorithm combine the advantages of each improvement strategy to improve the performance of the algorithm, and rely on the effect of different strategies to neutralize the shortcomings of each improvement strategy, avoiding the influence of the shortcomings of the improvement strategy on the algorithm.

5 SPPE Applied to Data Clustering

5.1 SPPE-K-Means

To address the dependency of K-means on the initial cluster centers, we propose an improved K-means algorithm that leverages the rapid convergence of the SPPE algorithm and its capability to efficiently find the global optimal solution. The SPPE algorithm optimizes the position of Phasmatodea through iterative updates, and using the final optimal solution as the clustering center, instead of the clustering center obtained by the traditional algorithm after random initialization which has uncertain factors. To validate the clustering performance of SPPE in this paper, classic datasets from the UCI repository including Wine, Iris, Zoo, Glass, Pima, Ionosphere, and Vehicle were selected for experimentation. The dataset's comprehensive information is provided in Table 6.

Table 6: Information on the experimental dataset

Dataset	Number of samples	Number of attributes	Number of categories
Ionosphere	351	34	2
Glass	214	9	6
Iris	178	13	3
Zoo	101	16	7
Wine	150	4	3
Pima	768	8	2
Vehicle	846	18	4

5.2 Analysis of Clustering Experiment Results

In this simulation experiment, GA, PSO, GWO, PPE, HHO, AOA, BOA and SPPE are introduced for comparison, and the algorithms are iterated for 100 times and run independently for 10 times, and the experimental findings are presented in Tables 7–9.

Table 7: WCSS metric values for different algorithms on each dataset

		GA	PSO	GWO	PPE	HHO	AOA	BOA	SPPE
Wine	Mean	1074747167	1023126880	29295409.3	90605057.5	46524302.7	29539220.8	56269960.6	**18907733**
	Std	264200697	147967878	6738203.93	59760327.4	30059233.2	13689041.7	18072689.5	1063610.24
Iris	Mean	1078.35308	1076.72054	296.19085	266.367679	326.919305	517.879426	579.042695	**156.43405**
	Std	179.875371	231.285195	107.884534	269.55373	245.362891	147.831228	164.269975	38.359326
Zoo	Mean	13145.0101	12863.9816	1195.95357	3790.57588	**582.11333**	677.85017	925.549847	723.981388
	Std	1652.93858	1420.38352	292.031783	4258.46032	85.0683406	140.216101	63.9148596	78.0274691
Glass	Mean	938434.472	759303.324	9944.19096	197751.585	2361.9365	157690.659	139874.997	**2266.138**
	Std	211369.424	210048.694	3954.18822	377240.585	823.79519	96116.8174	99586.5277	376.150996
Pima	Mean	399.292039	418.208256	157.085871	133.767382	208.451289	214.321593	224.334146	**133.13366**
	Std	91.2762013	69.1434394	10.2629981	7.11376689	24.5434324	26.7289556	22.5241903	44.7431245
Ionosphere	Mean	5637.65527	5689.37483	3576.58439	2760.14604	3395.20926	3662.17161	3753.32245	**2591.3896**
	Std	396.702148	241.760686	3355.76735	71.147509	33.3540917	80.7763106	107.42683	216.652481
Vehicle	Mean	5340.18999	5047.03627	2809.8837	1632.15444	2741.74071	3147.54261	3205.19453	**1496.9494**
	Std	601.056512	554.450656	350.19696	351.790043	161.962347	144.963233	262.413113	154.230178

Note: In the table, bold entries indicate the optimal values.

In this paper, we utilize Within-Cluster Sum of Squares (WCSS), Silhouette Coefficient (Sc) and the Davies-Bouldin Index (DBI). Sc is used to evaluate the goodness of clustering effect, DBI index is used as a metric to evaluate the performance of clustering algorithms, and WCSS is a widely used method of evaluating

the effectiveness of clustering.

$$WCSS = \sum_{i=1}^{k} \sum_{x \in U_i} |x - c_i|^2 \tag{20}$$

where U_i represents the set of all samples in the i-th cluster. The smaller the WCSS, the better the clustering performance of the algorithm. Sc combines the tightness within a cluster and its separation from other clusters. The value of Silhouette Coefficient ranges from −1 to 1, the closer to 1, the better the clustering result.

$$Sc(i) = \frac{b(i) - a(i)}{max(a(i), b(i))} \tag{21}$$

The average distance value of a sample to other points in the same cluster is $a(i)$ and $b(i)$ is the average distance value of a sample to samples in different clusters closest to it. The DBI index assesses the closeness between clusters and the proximity of sample points within clusters once the clustering process is finished.

$$DB = \frac{1}{k} \sum_{i=1}^{k} \max \frac{S_i + S_j}{d(i,j)} \tag{22}$$

$$S_i = \sqrt{\frac{1}{n_i} \sum_{a=1}^{n_i} \left\| x_a^{(i)} - c_i \right\|^2} \tag{23}$$

where k denotes the number of clusters, S_i signifies the average distance from all samples in the i-th cluster to its center, n_i indicates the number of samples in the i-th cluster, and $d(i, j)$ is the distance between the centroids of the i-th and the j-th clusters. The DBI index represents the ratio of the tightness within clusters to the separation between clusters. A higher level of tightness within clusters, coupled with greater separation between them, indicates a more effective clustering outcome.

Table 7 displays the WCSS indicator values of the algorithms on each dataset. On the Zoo dataset, the results of SPPE are not as good as those of the HHO algorithm. However, in datasets such as Iris and Wine where data separation is not distinct, the WCSS results of SPPE-K-means clustering algorithm are improved. Meanwhile, on complex datasets like Pima and Glass, the clustering performance of SPPE-K-means is notably superior to other algorithms. This supports the effectiveness of the enhanced algorithm presented. Table 8 displays the DBI metric values for the algorithm across the datasets. On each dataset, the results of SPPE consistently yield the minimum value, indicating that incorporating the SPPE algorithm into K-means improves its clustering effectiveness. This result verifies that the SPPE-K-means algorithm has a good clustering effect on these seven datasets, effectively reduces the dependence of K-means on the initial clustering centers, and shows that K-means combined with the SPPE algorithm can improve its clustering effect. The Sc results, as shown in Table 9, indicate that the SPPE algorithm outperforms other algorithms on most datasets, except for the Pima dataset. This demonstrates that the intra-cluster samples are tightly distributed, and the clustering performance is significant, providing a strong basis for evaluating the clustering effectiveness. In summary, the SPPE algorithm can be effectively used in the field of clustering analysis and has strong applicability.

Table 8: DBI metric values for different algorithms on each dataset

		GA	PSO	GWO	PPE	HHO	AOA	BOA	SPPE
Wine	Mean	6.388506	5.907917	1.733354	2.8504704	2.7284255	0.894031	2.110252	**0.6356308**
	Std	0.529427	0.402513	0.205393	0.2629728	0.9585203	0.457415	0.400036	0.0926289
Iris	Mean	1.6368825	1.575565	0.856628	0.775189	0.83433	1.044958	1.211281	**0.635631**
	Std	0.3554493	0.280914	0.06401	0.153117	0.084156	0.113488	0.161626	0.092629
Zoo	Mean	11.58504	11.82228	5.17913	8.324481	2.198299	4.216268	7.733347	**0.333515**

(Continued)

Table 8 (continued)

		GA	PSO	GWO	PPE	HHO	AOA	BOA	SPPE
	Std	0.985365	0.965011	1.20708	1.64106	0.725014	4.243484	3.049385	0.541015
Glass	Mean	3.650136	3.723107	1.9927	2.67871	2.33893	0.53537	3.201357	**0.47783**
	Std	0.212855	0.240576	0.083606	0.27222	0.17016	0.04577	0.165428	0.43041
Pima	Mean	1.238404	1.261189	0.889104	0.850765	0.928978	0.962727	1.072583	**0.684489**
	Std	0.103682	0.086077	0.030425	0.174447	0.058496	0.054619	0.045898	0.094428
Ionosphere	Mean	1.663545	1.668663	1.237264	1.295554	1.234572	1.61163	1.411567	**1.025201**
	Std	0.115572	0.086534	0.024973	0.109426	0.067426	0.099324	0.061377	0.050053
Vehicle	Mean	1.698139	1.642265	1.316946	1.291805	1.359096	1.546008	1.440858	**1.11879**
	Std	0.095696	0.108278	0.037065	0.102189	0.10184	0.094903	0.074888	0.222149

Note: In the table, bold entries indicate the optimal values.

Table 9: Sc metric values for different algorithms on each dataset

		GA	PSO	GWO	PPE	HHO	AOA	BOA	SPPE
Wine	Mean	0.3247723	0.3472307	0.5176163	0.4528987	0.4796872	0.4629137	0.3142687	**0.6865432**
	Std	0.1695174	0.2572736	0.0992017	0.2651527	0.1540728	0.3364921	0.0263721	0.289012
Iris	Mean	0.4768328	0.5173519	0.6698765	0.6709377	0.5376892	0.6982731	0.5569483	**0.7590121**
	Std	0.2946822	0.351275	0.0448405	0.109661	0.1693762	0.3556292	0.0672984	0.0452918
Zoo	Mean	0.2174824	0.3768295	0.4241561	0.4749019	0.5293847	0.5984726	0.3276037	**0.6378901**
	Std	0.5847248	0.2984732	0.0512099	0.0804871	0.3748293	0.2423565	0.1628471	0.1780123
Glass	Mean	0.3748279	0.3929385	0.4094093	0.3799591	0.5842938	0.5290847	0.4664703	**0.7679102**
	Std	0.3638295	0.1947228	0.1420999	0.1662631	0.2984729	0.3748209	0.2980293	0.1372345
Pima	Mean	0.3784074	0.2984029	0.2263243	0.2890801	0.5934255	**0.6329385**	0.3638295	0.5413451
	Std	0.2984249	0.1638295	0.0216127	0.0386702	0.1947289	0.1794729	0.3584743	0.2379567
Ionosphere	Mean	0.4729385	0.3782669	0.511336	0.5146962	0.5794043	0.5944253	0.4290475	**0.7525608**
	Std	0.1298047	0.0945024	0.0080495	0.0230494	0.2298476	0.2636295	0.1794428	0.1270461
Vehicle	Mean	0.3942043	0.3382947	0.3446542	0.4442173	0.5729385	0.5925639	0.4374829	**0.6131979**
	Std	0.5847629	0.2924463	0.0561124	0.0884927	0.3684729	0.2984729	0.6382947	0.0248901

Note: In the table, bold entries indicate the optimal values.

6 Conclusion and Future Work

The SPPE-based K-means algorithm proposed in this paper, which obtains optimized cluster centers, addresses the issue of K-means' significant dependency on the initial cluster centers. Firstly, VNS is introduced to quickly lock the high quality solutions. Secondly, by integrating the proposed spiral factor updating formula, individual search space is expanded, enhancing global search capability, thereby improving search efficiency and convergence speed. Finally, the balancing factor fully utilizes the effective information from the Phasmatodea population, making the SPPE algorithm more flexible, enhancing its exploration capabilities, and establishing a more balanced SPPE algorithm for exploration and exploitation.

Subsequently, we carried out extensive experiments on the SPPE algorithm with the CEC2013 benchmark functions. The experimental findings indicate that the SPPE algorithm offers notable benefits regarding performance and greatly enhances convergence. In addition, we also conducted clustering experiments based on the UCI dataset. The experimental findings indicate that K-means clustering, when based on the SPPE algorithm, demonstrates enhanced optimization capability and offers greater clustering stability.

Future research can focus on optimizing SPPE's performance by developing adaptive parameter adjustment methods to dynamically optimize key parameters (e.g., spiral mechanism, balancing factor) based on problem characteristics, enhancing adaptability. Additionally, as SPPE's computational complexity grows with population size and problem dimensionality, future studies will explore parallelization strategies, such as GPU acceleration or distributed frameworks, to improve scalability and efficiency for large-scale problems. These efforts will further enhance SPPE's robustness, efficiency, and practical application value.

While this study demonstrates the effectiveness of SPPE in data clustering, future research will explore its application to more complex tasks. SPPE can be applied to image segmentation, dividing images into meaningful regions based on features like pixel intensity or texture, and is well-suited for non-convex, high-dimensional search spaces. It could also analyze high-dimensional biological data, such as gene expression or protein interaction networks, to uncover meaningful patterns. These applications will further validate SPPE's utility across diverse and complex domains, enhancing its relevance in both theory and practice.

Acknowledgement: The authors are grateful to editor and reviewers for their valuable comments.

Funding Statement: The authors received no specific funding for this study.

Author Contributions: The authors confirm contribution to the paper as follows: study conception and design: Jeng-Shyang Pan and Mengfei Zhang; data collection: Jeng-Shyang Pan, Mengfei Zhang and Xingsi Xue; analysis and interpretation of results: Mengfei Zhang, Shu-Chuan Chu and Václav Snášel; draft manuscript preparation: Xingsi Xue and Václav Snášel. All authors reviewed the results and approved the final version of the manuscript.

Availability of Data and Materials: The authors confirm that the data supporting the findings of this study are availale within the article. The original dataset can be found at: https://archive.ics.uci.edu/datasets (accessed on 26 December 2024).

Ethics Approval: Not applicable.

Conflicts of Interest: The authors declare no conflicts of interest to report regarding the present study.

References

1. Saxena A, Prasad M, Gupta A, Bharill N, Patel OP, Tiwari A, et al. A review of clustering techniques and developments. Neurocomputing. 2017;267(10):664–81. doi:10.1016/j.neucom.2017.06.053.
2. Murtagh F, Contreras P. Algorithms for hierarchical clustering: an overview. Wiley Interdiscip Rev: Data Min Knowl Discov. 2012;2(1):86–97. doi:10.1002/widm.53.
3. Starczewski A, Scherer MM, KsiaŻek W, Debski M, Wang L. A novel grid-based clustering algorithm. J Artif Intell Soft Comput Res. 2021;11(4):319–30. doi:10.2478/jaiscr-2021-0019.
4. Cheam A, Marbac M, McNicholas P. Model-based clustering for spatiotemporal data on air quality monitoring. Environmetrics. 2017;28(3):e2437. doi:10.1002/env.2437.
5. Kriegel HP, Kröger P, Sander J, Zimek A. Density-based clustering. Wiley Interdiscip Rev: Data Mining Knowl Discov. 2011;1(3):231–40. doi:10.1002/widm.30.
6. Ahmed M, Seraj R, Islam SMS. The k-means algorithm: a comprehensive survey and performance evaluation. Electronics. 2020;9(8):1295.
7. Sinaga KP, Yang MS. Unsupervised K-means clustering algorithm. IEEE Access. 2020;8:80716–27.
8. Poggiali A, Berti A, Bernasconi A, Del Corso GM, Guidotti R. Quantum clustering with k-Means: a hybrid approach. Theor Comput Sci. 2024;992:114466.
9. Zhang C, Ouyang D, Ning J. An artificial bee colony approach for clustering. Expert Syst Appl. 2010;37(7):4761–7.
10. Zhang X, Lin Q, Mao W, Liu S, Dou Z, Liu G. Hybrid Particle Swarm and Grey Wolf Optimizer and its application to clustering optimization. Appl Soft Comput. 2021;101:107061.
11. Sharma V, Tripathi AK. A systematic review of meta-heuristic algorithms in IoT based application. Array. 2022;14(1):100164. doi:10.1016/j.array.2022.100164.
12. Vasconcelos J, Ramirez JA, Takahashi R, Saldanha R. Improvements in genetic algorithms. IEEE Trans Magn. 2001;37(5):3414–7. doi:10.1109/20.952626.
13. Whitley D. A genetic algorithm tutorial. Stat Comput. 1994;4(2):65–85. doi:10.1007/BF00175354.
14. de Souza RCT, de Macedo CA, dos Santos Coelho L, Pierezan J, Mariani VC. Binary coyote optimization algorithm for feature selection. Pattern Recognit. 2020;107(5):107470. doi:10.1016/j.patcog.2020.107470.

15. Mareli M, Twala B. An adaptive Cuckoo search algorithm for optimisation. Appl Comput Inform. 2018;14(2):107–15. doi:10.1016/j.aci.2017.09.001.
16. Das S, Mullick SS, Suganthan PN. Recent advances in differential evolution-an updated survey. Swarm Evol Comput. 2016;27:1–30.
17. Opara KR, Arabas J. Differential evolution: a survey of theoretical analyses. Swarm Evol Comput. 2019;44:546–58.
18. Alia OM, Mandava R. The variants of the harmony search algorithm: an overview. Artif Intell Rev. 2011;36:49–68.
19. Ma H. An analysis of the equilibrium of migration models for biogeography-based optimization. Inf Sci. 2010;180(18):3444–64.
20. Kennedy J, Eberhart R. Particle swarm optimization. In: Proceedings of ICNN'95-International Conference on Neural Networks; 1995; Perth, WA, Australia: IEEE. Vol. 4, p. 1942–8.
21. Wang D, Tan D, Liu L. Particle swarm optimization algorithm: an overview. Soft Comput. 2018;22(2):387–408. doi:10.1007/s00500-016-2474-6.
22. Wu D, Jiang N, Du W, Tang K, Cao X. Particle swarm optimization with moving particles on scale-free networks. IEEE Trans Netw Sci Eng. 2018;7(1):497–506. doi:10.1109/TNSE.2018.2854884.
23. Gupta S, Deep K. A novel random walk grey wolf optimizer. Swarm Evol Comput. 2019;44(4):101–12. doi:10.1016/j.swevo.2018.01.001.
24. Mirjalili S, Mirjalili SM, Lewis A. Grey wolf optimizer. Adv Eng Softw. 2014;69:46–61. doi:10.1016/j.advengsoft.2013.12.007.
25. Mirjalili S, Lewis A. The whale optimization algorithm. Adv Eng Softw. 2016;95(12):51–67. doi:10.1016/j.advengsoft.2016.01.008.
26. Yang W, Xia K, Fan S, Wang L, Li T, Zhang J, et al. A multi-strategy whale optimization algorithm and its application. Eng Appl Artif Intell. 2022;108:104558. doi:10.1016/j.engappai.2021.104558.
27. Arora S, Singh S. Butterfly optimization algorithm: a novel approach for global optimization. Soft Comput. 2019;23:715–34. doi:10.1007/s00500-018-3102-4.
28. Hashim FA, Hussain K, Houssein EH, Mabrouk MS, Al-Atabany W. Archimedes optimization algorithm: a new metaheuristic algorithm for solving optimization problems. Appl Intell. 2021;51:1531–51. doi:10.1007/s10489-020-01893-z.
29. Alabool HM, Alarabiat D, Abualigah L, Heidari AA. Harris hawks optimization: a comprehensive review of recent variants and applications. Neural Comput Appl. 2021;33:8939–80. doi:10.1007/s00521-021-05720-5.
30. Dorigo M, Blum C. Ant colony optimization theory: a survey. Theor Comput Sci. 2005;344(2–3):243–78. doi:10.1016/j.tcs.2005.05.020.
31. Parpinelli RS, Lopes HS, Freitas AA. Data mining with an ant colony optimization algorithm. IEEE Trans Evol Comput. 2002;6(4):321–32. doi:10.1109/TEVC.2002.802452.
32. Mavrovouniotis M, Li C, Yang S. A survey of swarm intelligence for dynamic optimization: algorithms and applications. Swarm Evol Comput. 2017;33:1–17. doi:10.1016/j.swevo.2016.12.005.
33. Song PC, Chu SC, Pan JS, Yang H. Simplified Phasmatodea population evolution algorithm for optimization. Complex Intell Syst. 2022;8(4):2749–67. doi:10.1007/s40747-021-00402-0.
34. Pan JS, Song PC, Pan CA, Abraham A. The phasmatodea population evolution algorithm and its application in 5G heterogeneous network downlink power allocation problem. J Internet Technol. 2021;22(6):1199–213. doi:10.53106/160792642021112206001.
35. Zhang AN, Chu SC, Song PC, Wang H, Pan JS. Task scheduling in cloud computing environment using advanced phasmatodea population evolution algorithms. Electronics. 2022;11(9):1451. doi:10.3390/electronics11091451.
36. Cai Z, Gong W, Ling CX, Zhang H. A clustering-based differential evolution for global optimization. Appl Soft Comput. 2011;11(1):1363–79. doi:10.1016/j.asoc.2010.04.008.
37. Nayak J, Naik B, Behera H. Cluster analysis using firefly-based K-means Algorithm: a combined approach. In: Computational Intelligence in Data Mining: Proceedings of the International Conference on CIDM; 2016 Oct 10–11; Singapore: Springer; 2017. p. 55–64.
38. Chaudhary R, Banati H. Hybrid enhanced shuffled bat algorithm for data clustering. Int J Adv Intell Paradig. 2020;17(3–4):323–41. doi:10.1504/IJAIP.2020.109513.

39. Huang SC. Color image quantization based on the artificial bee colony and accelerated K-means algorithms. Symmetry. 2020;12(8):1222. doi:10.3390/sym12081222.
40. García J, Yepes V, Martí JV. A hybrid k-means cuckoo search algorithm applied to the counterfort retaining walls problem. Mathematics. 2020;8(4):555. doi:10.3390/math8040555.
41. Pal SS, Pal S. Black hole and k-means hybrid clustering algorithm. In: Computational Intelligence in Data Mining: Proceedings of the International Conference on ICCIDM 2018; 2020; Singapore: Springer; p. 403–13.
42. Hansen P, Mladenović N, Brimberg J, Pérez JAM. Variable neighborhood search. Cham: Springer; 2019.