Electronic Research Archive

http://www.aimspress.com/journal/ERA

ERA, 31(11): 7078–7103.
DOI: 10.3934/era.2023359
Received: 04 August 2023
Revised: 19 September 2023
Accepted: 26 September 2023
Published: 06 November 2023

Research article

Adaptive clustering algorithm based on improved marine predation algorithm and its application in bearing fault diagnosis

Zhuanzhe Zhao[1,2], Mengxian Wang[3], Yongming Liu[1,4,*], Zhibo Liu[1,4,*], Yuelin Lu[1,4], Yu Chen[3,5], Zhijian Tu[6]

[1] School of Artificial Intelligence, Anhui Polytechnic University, Wuhu 241000, China
[2] Anhui Provincial Key Laboratory of Discipline Co-construction on Intelligent Equipment Quality and Reliability, Wuhu 241000, China
[3] School of Mechanical Engineering, Anhui Polytechnic University, Wuhu 241000, China
[4] Center for Robot Performance Testing and Reliability Assessment, Anhui Polytechnic University, Wuhu 241000, China
[5] Anhui Provincial Key Laboratory of Electric Drive and Control, Anhui Polytechnic University, Wuhu 241000, China
[6] Wuhu Ceprei Robotics Industry Technology Research Institute Company Limited, Wuhu 241000, China

* **Correspondence:** Email: liuym@ahpu.edu.cn, liuzhibo@mail.ahpu.edu.cn.

Abstract: In cluster analysis, determining the number of clusters is an important issue because there is less information about the most appropriate number of clusters in the real problem. Automatic clustering is a clustering method that automatically finds the most appropriate number of clusters and divides instances into the corresponding clusters. In this paper, a novel automatic clustering algorithm based on the improved marine predator algorithm (IMPA) and K-means algorithm is proposed. The new IMPA utilizes refracted opposition-based learning in population initialization, generates opposite solutions to improve the diversity of the population and produces more accurate solutions. In addition, the sine-cosine algorithm is incorporated to balance global exploration and local development of the algorithm for dynamic updating of the predator and prey population positions. At the same time, the Gaussian-Cauchy mutation is combined to improve the probability of obtaining the globally optimal solution. The proposed IMPA is validated with some benchmark data sets. The calculation results show that IMPA is superior to the original MPA in automatic clustering. In addition, IMPA is also used to solve the problem of fault classification of Xi'an Jiaotong University bearing data. The results show that the IMPA has better and more stable results than other algorithms such as the original

MPA, whale optimization algorithm, fuzzy C-means and K-means in automatic clustering.

Keywords: fault diagnosis; automatic clustering; cluster validity index; marine predator algorithm; K-means clustering

1. Introduction

Bearings are one of the important parts in machinery such as motors, gearboxes or wind turbines, and their functional state directly determines the performance and efficiency of the mechanical system [1]. Often working for long periods of time in environments with variable loads, speeds and vibrations, they are prone to failure, which can lead to significant economic losses and catastrophic accidents. Therefore, in order to detect potential faults in a timely manner and reduce their impact, advanced fault diagnosis technology must be developed [2].

In order to accurately detect bearing faults and with advances in computational intelligence technology, data-driven machine learning (ML) methods have been widely used in fault diagnosis because of their ability to characterize the intrinsic correlation between measurements and fault conditions [3]. Once a ML model is established, fault diagnosis can be effectively implemented [4]. ML techniques can be divided into supervised, unsupervised, semi-supervised and reinforcement learning. A variety of ML techniques are used for different types of problems. In general, supervised learning solves regression and classification processes, and common algorithms are k-nearest neighbor (KNN), support vector machines (SVM), decision tree (DT), random forest (BF) and neural network (NN). Liang et al. [5] integrated wavelet transform into the improved residual neural network and used the global singular value decomposition adaptive strategy for feature extraction for fault diagnosis of rolling bearings in noisy environments. The improved model has good anti-noise robustness. Xu et al. [6] developed a hybrid deep learning model based on a convolutional neural network and deep forest model. Compared with other existing methods, although the computational efficiency is significantly reduced, it has better detection accuracy. The typical unsupervised method is the clustering algorithm, including k-means clustering (KCM), fuzzy clustering algorithm and density-based clustering method (DBSCAN). Yuan et al. [7] proposed a fault diagnosis algorithm based on adaptive hierarchical clustering and subset to extract features and apply them to gearbox fault diagnosis. Hou et al. [8] proposed a clustering fault diagnosis method for rolling bearings based on ensemble empirical mode decomposition, permutation entropy, linear discriminant analysis and the Gath-Geva clustering algorithm, which has the advantage of better class clustering compactness.

Faced with the lack of labeled data for the training of fault diagnosis models, researchers often use clustering algorithms. It is a data analysis problem based on the given sample being grouped into several "classes" or "clusters" according to the similarity or distance of their features [9]. The purpose of cluster analysis is to discover the characteristics of the data or process the data through the obtained "classes" or "clusters," and it has been widely used in data analysis problems in many fields such as image segmentation [10], wind speed forecasting [11], fault diagnosis [12] and so on.

Cluster analysis algorithms are divided into two categories: partition clustering algorithms and hierarchical clustering algorithms [13]. The partition based method is a simple and commonly used clustering method that clusters objects by dividing them into mutually exclusive clusters, with each object belonging to only one cluster. The partitioning results aim to make the similarity between clusters low and the similarity within clusters high. Common partitioning methods include k-means, k-medoids, K-prototype and other methods. The application of hierarchical clustering is second only to the partition-based clustering. The core idea is to divide the data into clusters of different layers according to the hierarchy of the data set, to form a tree-shaped cluster structure. According to the process of hierarchical clustering, it can be divided into bottom-up clustering and top-down splitting clustering. Aggregation clustering is represented by balanced iterative reducing and clustering using hierarchies (BIRCH), there is a robust clustering algorithm for categorical attributes (ROCK), and split clustering is represented by a divisive analysis (DIANA) algorithm.

As an excellent clustering algorithm, the KCM [14] algorithm has the characteristics of fast computing speed, strong scalability and being simple, fast and easy to understand. It has been widely used in research of rotating machinery fault data analysis. Xue et al. [15] combined KCM with AdaBoost to optimize artificial hydrocarbon networks. They proposed an intelligent diagnosis based on double-optimized artificial hydrocarbon networks to identify mechanical faults of an in-wheel motor. Mariela et al. [16], inspired by KCM and the one nearest neighbor algorithm, put forward a framework to detect a new mode of abnormal conditions in gearboxes. Wan et al. [17] proposed a Spark-based parallel ant colony optimization KCM algorithm for rolling bearing fault diagnosis based on massive running state monitoring data of rolling bearings, to realize efficient and accurate fault diagnosis of rolling bearings.

The above research shows that KCM and rotary fault diagnosis technology have a good combination and can effectively identify the fault state. However, most of these require some domain knowledge to select the appropriate cluster number for the dataset containing data objects with different densities and sizes. The requirement of predefined cluster number makes KCM inefficient for automatic cluster analysis and prone to problems such as slow convergence and low accuracy, thus affecting the efficiency and accuracy of fault diagnosis [18]. In recent years, KCM has been combined with naturally inspired metaheuristic optimization algorithms to overcome the challenges of traditional clustering algorithms in processing automatic data clustering. Metaheuristic optimization algorithms have global search capability for optimizing the number of clusters and the clustering centroid [19]. The metaheuristic algorithm is mainly used to help the K-means algorithm to get rid of locally optimal convergence by searching the globally optimal initial cluster centroid. Its effectiveness has been widely verified in the literature. Using global-local optimization as an idea, Zhang et al. [20] constructed a hybrid genetic algorithm-quasi-Newton method optimization parameter model, which can effectively identify a faulty combustor can based on gas temperature profile. In Yang and Sutrisno [21], the symbiotic organisms search (SOS) algorithm was applied to the initial solution of automatic K-means to create subpopulations, thereby improving the quality and efficiency of the search. The subeconomies created by automatic K-means enables the clustering-based symbiotic organisms search (CSOS) algorithm to combine local and global searches on datasets.

The marine predators algorithm (MPA) is a novel and efficient metaheuristic algorithm proposed by Faramarzi et al. [22] in 2020. It is primarily inspired by the movement between predator and prey in the ocean and the optimal encounter rate strategy in biological interaction. MPA has many advantages, including fewer parameters, higher flexibility and ease of implementation [23], and it has been applied to solve many problems in different fields [24,25].

Therefore, this study aims to improve the MPA algorithm for better automatic clustering results. The initialization of MPA enters the high-speed ratio phase with random factors, greatly reducing optimization efficiency and robustness and slowing the initial convergence. In the intermediate stage of optimization, the population wandering strategy needs to be more flexible to lead to an imbalance between exploration and exploitation. On the other hand, the MPA will converge prematurely if prey forages successfully. Therefore, an improved MPA algorithm is proposed and applied to the automatic clustering of bearing faults. The main contributions of this paper can be summarized as follows:

•An improved marine predator algorithm (IMPA), combining refracted opposition-based learning (OBL), sine-cosine algorithm (SCA) and Gaussian-Cauchy mutation (GCM), is developed to improve the convergence accuracy of standard MPA and the ability to handle high-dimensional problems. The efficiency of the IMPA is tested by solving 10 classical optimization functions.

•The proposed IMPA is applied to the automatic clustering of 7 common datasets and Xi'an Jiaotong University bearing data set, and the results are compared with the whale optimization algorithm (WOA), sparrow search algorithm (SSA) and k-means and fuzzy C-means (FCM) algorithms of automatic clustering. The results show that IMPA is more competitive than other optimization methods.

The rest of this article is summarized as follows: Section 2 introduces the concept of cluster correlation, and Section 3 introduces MPA, IMPA and an improved hybrid algorithm for automatic clustering based on IMPA. Section 4 validates the hybrid algorithm. The conclusions and future research directions are presented in Section 5.

2. Cluster analysis

2.1. Description of clusters

For a given dataset $X = \{x_1, x_2, \dots, x_N\}$, the clustering algorithm divides it into *K* classes, that is, $C = \{C_1, C_2, \dots, C_K\}$. Get the partitioning matrix $U(X)$, and the partition matrix $U = [\mu_{ij}]_{K\times N}$, $i = 1,2,\dots,K$, $j = 1,2,\dots,N$, where μ_{ij} is the attachment of the sample x_j to the C_i class.

If μ_{ij} satisfies

$$\mu_{ij} = \begin{cases} 1, & \text{if } x_j \in C_i \\ 0, & \text{otherwise} \end{cases} \tag{1}$$

and the result of cluster division is satisfied, that is, $C_i \neq \emptyset$; $C_i \cap C_j = \emptyset$, $(i, = 1,2,\dots,K,\ j = 1,2,..,N$; $\bigcup_{i=1}^{K} C_i = X$, then this is called cluster analysis. Each data object is assigned to a certain category according to the principle of similarity, which is generally measured by Euclidean distance [26], that is,

$$d(\vec{x}_i, C_j) = \sqrt{\sum_{p=1}^{N}(\vec{x}_{i,p} - C_{j,p})} = \|\vec{x}_i - C_j\|. \tag{2}$$

2.2. KCM

Since the proposed new automatic clustering method combines KCM in part, the basic principle of the K-means algorithm is briefly introduced. The goal of K-means is to divide the dataset into *K* separate parts to minimize the Euclidean distance between each object and the cluster to which it belongs. The main steps of a standard K-means algorithm are usually represented as follows [27]:

1) Select *K* objects from *N* data as the initial cluster center $\{Ci\}$.

2) Calculate the Euclidean distance from each data point to *K* initial clustering centers, and reclassify

the data according to the principle of minimum distance.

3) Calculate the average of each dimension of the cluster to get the new center. Repeat steps 2) and 3) until the cluster center no longer changes or the distance at which the cluster center is less than the set threshold, and output the cluster center of the data set.

As is known from the clustering process, improper selection of initial clustering center can cause falling into local optimization. The calculation formula of cluster center C_j is as follows:

$$C_j = \frac{1}{C_j}\sum_{i=1}^{N} x_i\,, \qquad j = 1,2,\dots,\mathrm{K} \tag{3}$$

Therefore, the initialization of KCM has randomness, which may take noise or an abnormal value as the initial center, resulting in a large difference in the results of each clustering and poor algorithm robustness. At the same time, the simple updating process will cause KCM to fall into local optimality.

2.3. Automate clustering and objective functions

In order to implement automatic clustering in this paper, we need cluster center information and activation threshold or switch vector. By setting a constant value, if the activation threshold is greater than this value, it indicates that the centroid of the cluster is an effective centroid. If we have *K* centers, the activation threshold part has *K* elements, and the coordinates of the center are $K \times N$ elements, which is actually $K \times (N+1)$. Overall, we have a matrix with *K* rows and $N+1$ columns. The following matrix shows the relationship between the clustering centroid and the activation threshold:

$$\left.\begin{bmatrix} m_1 | a_1 \\ \vdots \,|\, \vdots \\ m_K | a_K \end{bmatrix}\right\} K \qquad \underset{N}{\leftrightarrow} \quad \underset{1}{\leftrightarrow} \tag{4}$$

where *K* represents the row, *m* is the centroid vector, *N* represents the column, and $a \in [0,1]$ is the activation threshold. In this article, the threshold is set to 0.4; if $a < 0.4$, the center is invalid. Figure 1 shows how to activate or deactivate a cluster center case.

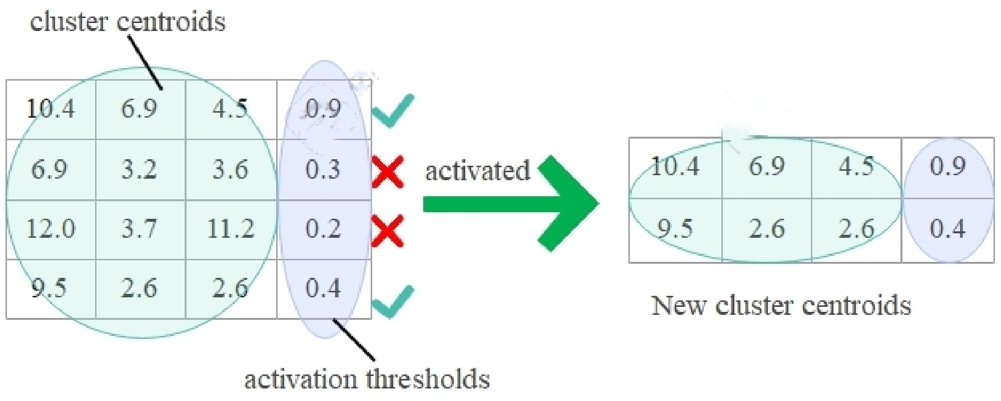


Figure 1. Illustration of the cluster centroid activated or deactivated.

In automatic clustering, the commonly used objective function is the optimization cluster validity index (CVI), used to evaluate clustering [28]. The validity index is a criterion for the degree of completion of cluster analysis. The objective function (CVI) needs to be minimized, and the lower the CVI value is, the better the clustering result, with better compactness and further cluster separation. In automatic clustering, if the number of clusters is specified, the previous objective function (Eq (2)) works fine; but if the number of clusters is unknown, another objective function must be used. To date, researchers have provided effectiveness indices for a number of clusters, such as the Davis-Bouldin index (DB) [29], Calinski-Harabasz index (CH) [30], Pakhira Bandyopadhyay Maulik index (PBM) [31], Silhouette [32] and compact separation index (CS) [33].

In this article, we use the CS index as the objective function to measure the effectiveness of the cluster. CS measures the quality of the clustering results using the ratio of scatter points within a cluster to the sum of separations between clusters. The lower the value of CS is, the better the separation, and more compact clustering is reflected. It has been reported to be a better CVI in terms of efficiently handling clusters with different densities, dimensions and sizes [34]. CS is expressed in Eq (5) below:

$$CS = \frac{\sum_{i=1}^{K}\left[\frac{1}{Q_i}\sum_{X_i \in Q} maxX_j \in Q\{V(X_i, X_j)\}\right]}{\sum_{i=1}^{K}\left[min_{j=K, j \neq i}\{V(x_i, x_j)\}\right]} \tag{5}$$

where the number of data points in cluster *C* is expressed as Q, and the distance between intra-cluster scattering X_i and inter-cluster separation X_j is expressed as the function $V(X_i, X_j)$. The distance between the data point and its center of mass is expressed as $V(x_i, x_j)$.

3. MPA

3.1. Standard MPA

MPA is a new metaheuristic algorithm that draws inspiration from nature and is based on the various foraging techniques used by marine predators and the best encounter rate policy in biological interaction. Like most metaheuristics, the initial solution is evenly distributed in the search space. The initialization formula is as follows:

$$P_{ij} = l_j + R_1(u_j - l_j) \tag{6}$$

where $u_j$and l_j are the lower and upper bound of the variable, and R_1 is a uniform random vector in the range of 0 to 1. The initialization creates the initial *Prey* matrix, as shown below:

$$Prey = \begin{bmatrix} P_{1,1} & P_{1,2} & \cdots & P_{1,d} \\ P_{2,1} & P_{2,2} & \cdots & P_{2,d} \\ \vdots & \vdots & \ddots & \vdots \\ P_{n,1} & P_{n,2} & \cdots & P_{n,d} \end{bmatrix}_{n \times d} \tag{7}$$

where *n* is the population number, and *d* is the problem dimension.

Calculate the fitness of each prey, select the most fit prey individual as the top predator, and build it into the *Elite* matrix:

$$Elite = \begin{bmatrix} P_{1,1}^{I} & P_{1,2}^{I} & \cdots & P_{1,d}^{I} \\ P_{2,1}^{I} & P_{2,2}^{I} & \cdots & P_{2,d}^{I} \\ \vdots & \vdots & \ddots & \vdots \\ P_{n,1}^{I} & P_{n,2}^{I} & \cdots & P_{n,d}^{I} \end{bmatrix}_{n\times d} \tag{8}$$

The MPA optimization process is divided into three phases based on the different velocity ratios of the predator and prey. 1) At a high-velocity ratio, the prey moves faster than the predator. 2) At a unit-velocity ratio phase, both prey and predator speeds are similar. 3) At a low-velocity ratio phase, the prey is slower than the predator. At each stage, the movement of the predator and prey in nature is imitated separately.

Phase 1: The high-velocity ratio phase at the beginning of the iteration. The best strategy for the predator is to remain completely immobile, while the prey obeys Brownian motion. The mathematical model for this phase is shown as follows:

$$\begin{cases} s_i = R_B \otimes (Elite_i - R_B \otimes Prey_i) \\ Prey_i = Prey_i + 0.5 R_v \otimes s_i \end{cases} i = 1,2,\ldots,n;\ Iter < \frac{1}{3} Max\_Iter \tag{9}$$

where s_i is the prey search step. R_B represents the Brownian motion, which is a random vector with a normal distribution. The notion $\otimes$ indicates entry-wise multiplications. $R_v \in (0,1)$ is a uniformly distributed random vector. $Iter$ denotes the current number of iterations, and Max_Iter is the maximum number of iterations.

Phase 2: The unit-velocity ratio phase is the middle of the iteration, requiring both exploration and exploitation. Thus, half of the population is used for exploitation to perform Lévy movement and the other half for exploration to practice Brownian motion. The following formulas are proposed in this regard:

$$\frac{1}{3} Max\_Iter < Iter < \frac{2}{3} Max\_Iter$$

$$\begin{cases} s_i = R_L \otimes (Elite_i - R_L \otimes Prey_i) \\ Prey_i = Elite_i + 0.5 R_v \otimes s_i \end{cases} i = 1,2,\ldots,\frac{n}{2}; \tag{10}$$

$$\begin{cases} s_i = R_B \otimes (R_B \otimes Elite_i - Prey_i) \\ Prey_i = Elite_i + 0.5 C_F \otimes s_i \end{cases} i = \frac{n}{2},\ldots,n; \tag{11}$$

$$C_F = \left(1 - \frac{Iter}{Max\_Iter}\right)^{\frac{2\times Iter}{Max\_Iter}} \tag{12}$$

where R_L is a vector of random numbers based on Lévy distribution; C_F is considered an adaptive parameter for controlling the step size of predator movement.

Phase 3: The exploitation process is mainly carried out. The best strategy for the predator is Lévy movement. Its mathematical model is

$$\begin{cases} s_i = R_L \otimes (R_L \otimes Elite_i - Prey_i) \\ Prey_i = Elite_i + 0.5 C_F \otimes s_i \end{cases} i = 1,\ldots,n;\ Iter > \frac{2}{3} Max\_Iter \tag{13}$$

In addition, environmental issues such as eddy formation or fish aggregating devices (FADs) also can cause behavioral changes in marine predators, which are considered locally optimal. So, longer jumps are used to avoid stagnation when finding the best. The model of the process is as follows:

$$Prey_i = \begin{cases} Prey_i + C_F[P_{min} + R_L \otimes (P_{max} - P_{min})] \otimes Y, R_2 \leq 0.2 \\ Prey_i + [0.2(1 - R_2) + R_2](Prey_{r1} - Prey_{r2}), R_2 > 0.2 \end{cases} \tag{14}$$

where Y is the binary vector with arrays including 0 and 1. R_2 is a uniform random number in [0,1]. $Prey_{r1}$ and $Prey_{r2}$ denote two individuals randomly selected from the prey matrix. The pseudo-code for MPA is as follows (Algorithm 1).

```
Algorithm 1
1   Initialize search agents (Prey) populations $i$ = 1,...,$n$
2   **While** termination criteria are not met
3       Calculate the fitness and construct the Elite matrix
4       **If** $Iter < Max\_Iter/3$
5           Update prey based on Eq (9)
6           **Else if** $Max\_Iter/3 < Iter < 2*Max\_Iter/3$
7               For the first half of the populations ($i = 1, ..., n/2$)
8               Update prey based on Eq (10)
9               For the other half of the populations ($i = n/2, ..., n$)
10              Update prey based on Eq (11)
11          **Else if** $Iter > 2*Max\_Iter/3$
12              Update prey based on Eq (13)
13      **End (if)**
14      Accomplish memory saving and Elite update
15      Applying FADs effect and update based on Eq (14)
16      Accomplish memory saving and Elite update
17  **End while**
```

3.2. IMPA

In this part of the study, an IMPA is provided, using refracted OBL, SCA and GCM. Initializing MPA with random factors reduces optimization efficiency and robustness, while refracted OBL is used to generate a more widely distributed initial population, which may increase the speed of convergence in the initial search. In the intermediate stages of optimization, the pace of population movement limits the population exchange, thereby reducing the search area, and the SCA helps to achieve a smoother step transition in the balance phase. If the prey succeeds in foraging, it becomes a predator, leading the population to a local optimum. When the algorithm stagnates, Gaussian-Cauchy mutation is utilized to increase the diversity of the population and make the algorithm be out of the local optimum.

3.2.1 Refracted OBL

OBL is an optimization strategy proposed by Tizhoosh [35]. The basic idea is to consider the current solution and its opposite solution in order to obtain optimal solution acquisition, which is calculated as follows:

$$x^* = l + u - x \tag{15}$$

where l and u are, respectively, the lower and upper bounds. $x*$ represents the opposite solution of x. OBL is a common means of improving population initialization [36]. It is a simple approach, but low flexibility makes it detrimental to handling dynamic changes in the population. To improve the performance of OBL, the refraction law of light is added to obtain refraction opposition-based learning (ROBL) [37]. It can help generate high-quality populations and increase population diversity, thus avoiding blind search in the initial iterations and accelerating early convergence. The principle of ROBL is shown in Figure 2, and the mathematical model is described as follows:

$$\begin{cases} \sin\alpha = (\frac{l+u}{2} - x)/h \\ \sin\beta = (x^* - \frac{l+u}{2})/h^* \end{cases} \tag{16}$$

Then, the refractive medium rate z can be obtained as

$$z = \frac{sin\alpha}{sin\beta} = \frac{h^*\big((l+u)/2 - Prey\big)}{h(Prey^* - (l+u)/2)} \tag{17}$$

Let the scaling factor be $q = \frac{h}{h^*}$, and Eq (17) is changed to

$$Prey^* = \frac{l+u}{2} + \frac{l+u}{2qz} - \frac{Prey}{qz} \tag{18}$$

Equation (13) can extend to the d-dimensional space of the MPA to get the refracting solution as follows:

$$Prey_{i,j}^* = \frac{l_j + u_j}{2} + \frac{l_j + u_j}{2q} - \frac{Prey_{i,j}}{q} \tag{19}$$

where l_j and u_j are the minimum and maximum of the j-th dimension in the current population, respectively. $Prey_{i,j}$ represents the value of the j-th dimension of the i-th particle in the current population. $Prey_{i,j}^*$ is the refracting solution of $Prey_{i,j}$.

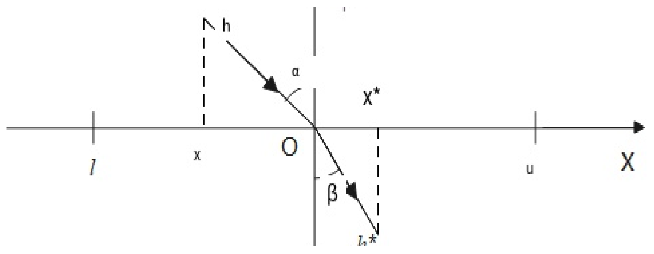


Figure 2. Refraction reverse learning schematic.

3.2.2. SCA

Based on the position update equations of MPA described in Eqs (10) and (11), MPA converges in advance during optimization iteration without fully exploring the search space. In addition, giant steps may occur in alternating Brownian and Lévy motions for optimization, traversing the optimal solution during the search process. The SCA is introduced in this paper to improve the performance of the MPA. SCA is a new intelligent algorithm proposed in 2016 by Mirjalili [38] which uses the oscillation switching characteristics of sine and cosine functions to find the optimum solution. This algorithm has the advantages of fast optimization and strong robustness. At the core of SCA is the location updating method, and its mathematical model of location updating is shown Eq (20):

$$Prey_i = \begin{cases} Prey_i + r_1 \times \sin r_2 \times |r_3 \times B - Prey_i|, & r_4 < 0.5 \\ Prey_i + r_1 \times \cos r_2 \times |r_3 \times B - Prey_i|, & r_4 \geq 0.5 \end{cases} \tag{20}$$

$$r_1 = 1 - \frac{Iter}{Max\_Iter} \tag{21}$$

where $r_2 \in [0,2\pi], r_3 \in [0,2], r_4 \in [0,1]$, and B is the target position, the global optimal position in SCA. The most critical parameter is r_1, which controls the balance between the exploration and exploitation of the algorithm. r_1 is a linearly decreasing function, and the change of function is unique and needs to be more flexible. For this purpose, the non-linear decreasing function r_1 expression in [39] is used as follows:

$$r_1 = 2e^{-\frac{Iter}{Max\_Iter}} \tag{22}$$

Since C_F is a non-linear descent curve, the rate of the population with Brownian motion decreases with the number of iterations, which follows the standard iteration law. So, only the population with Lévy motion is required to improve the search strategy. Then, the new predator position update equation is obtained as

$$While\ \frac{1}{3}Max\_Iter < Iter < \frac{2}{3}Max\_Iter$$

$$Prey_i = \begin{cases} Prey_i + r_1 + sinr_2 \times |r_3 - B - Prey_i|, r_4 < 0.5 \\ Prey_i + r_1 \times cosr_2 \times |r_3 - B - Prey_i|, r_4 \geq 0.5 \end{cases} i = 1, \ldots, \frac{n}{2} \tag{23}$$

3.2.3. GCM

In order to solve the problems of low accuracy and easily falling into a local optimum, MPA needs to incorporate a variational strategy. To this end, the GCM [40] strategy is adopted. The Gauss and Cauchy distributions are shown in Figure 3.

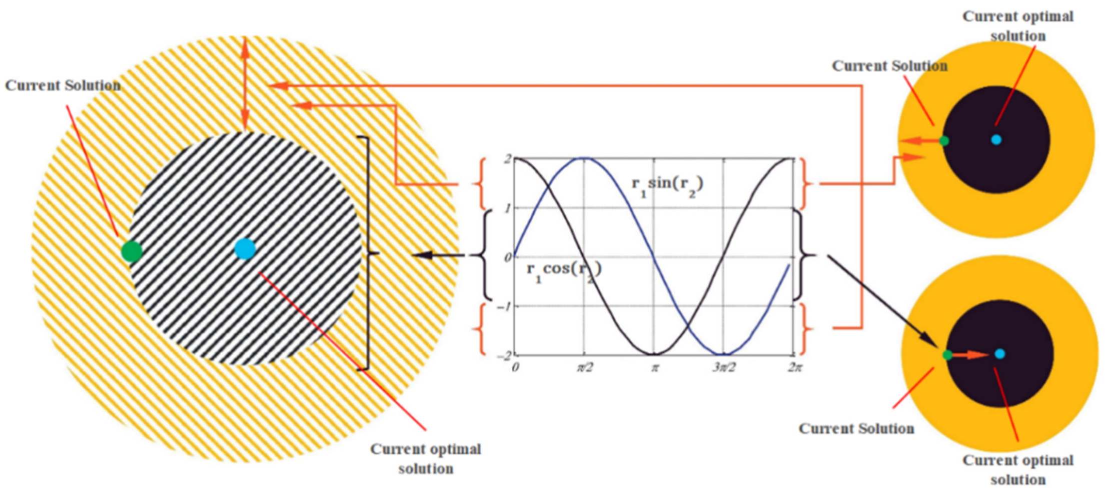


Figure 3. SCA schematic diagram.

The Cauchy distribution is small in the middle and large at both ends, having a larger search step and expanding the search range. The Gaussian distribution is highest in the middle and lowest at both ends, having a smaller search step and improving the ability to search locally. The GCM strategy is outlined as follows:

We have a Gaussian distribution with mean zero ($\mu = 0$) and unit variance ($\sigma^2 = 1$). We have a Cauchy distribution with position parameter ($x_0 = 0$) and scale parameter ($\gamma = 1$). Therefore, the GCM strategy is used in MPA to improve accuracy, which is described as follows:

$$GCM = \begin{cases} f_g(x;\mu,\sigma) = \dfrac{1}{\sigma\sqrt{2\pi}} e^{\frac{(x-\mu)^2}{2\sigma^2}} \\ f_c(x;x_0,\gamma) = \dfrac{1}{\pi}\dfrac{\gamma}{\gamma^2 + (x-x_0)^2} \end{cases} - \infty < x < \infty \tag{24}$$

$$\text{Prey} = \begin{cases} Prey * (1 + C(1,0)), R_3 \le 1 - \dfrac{Iter}{Max\_Iter} \\ Prey * (1 + G(0,1)), R_3 > 1 - \dfrac{Iter}{Max\_Iter} \end{cases} \tag{25}$$

where R_3 is a random number uniformly distributed between 0 and 1. $C(1,0)$ is a random variable based on the Cauchy distribution, and G(1,0) is based on the Gaussian one. $1 - Iter/Max\_Iter$ decreases with increasing $Iter$. In the early iterations, the value of $Iter$ is small, and the predator performs a Cauchy distribution to expand the search. Later in the iteration, the value of $Iter$ increases, and $1 - Iter/Max\_Iter$ becomes smaller for the Gaussian distribution. The two variant strategies are used alternately, which amounts to an adaptive dynamic parameter that increases with the number of iterations. It increases the opportunity that the algorithm will jump out of its stagnation and accelerate its convergence later. The pseudo-code of IMPA is as follows (Algorithm 2):

Algorithm 2

```
1  Initialize the population to construct the Prey matrix, determine the relevant parameters,
   and calculate the individual fitness
2  Apply the ROBL to generate the opposite prey position and calculate the individual fitness
3  Compare the fitnesses and sort them from small to large, and choose the top half
   corresponding population positions to construct the Prey matrix
4  While ($Iter < Max\_Iter$)
5     calculate the fitness of each agent
6     construct the Elite matrix and accomplish memory saving
7  for each agent
8     If $Iter < Max\_Iter /3$
9        update prey based on Eq 9
10    else if $Max\_Iter /3 < Iter < 2* Max\_Iter /3$
11          if $i <= n/2$
12                update prey based on Eq 23
13          else if $n/2 < i < n$
14                update prey based on Eq 11
15          end if
16    else if $Iter > 2* Max\_Iter /3$
17          update prey based on Eq (13)
18    end if
19 applying GCM and update based on Eq 25
20 calculate the fitness and update the Elite matrix
21 accomplish memory saving
22 applying FADs effect and update based on Eq 14
23 $Iter = Iter +1$
24 End while
```

3.3 Overall scheme of AC-IMPA

In summary, for the AC-IMPA method, after iterative optimization of the algorithm, the position vector that best fits the fitness function is selected as the cluster center representing a specific cluster number. According to the effective clustering centroid obtained, a better cluster grouping can be obtained. Figure 4 shows the flow chart of AC-IMPA, visually showing the whole process.

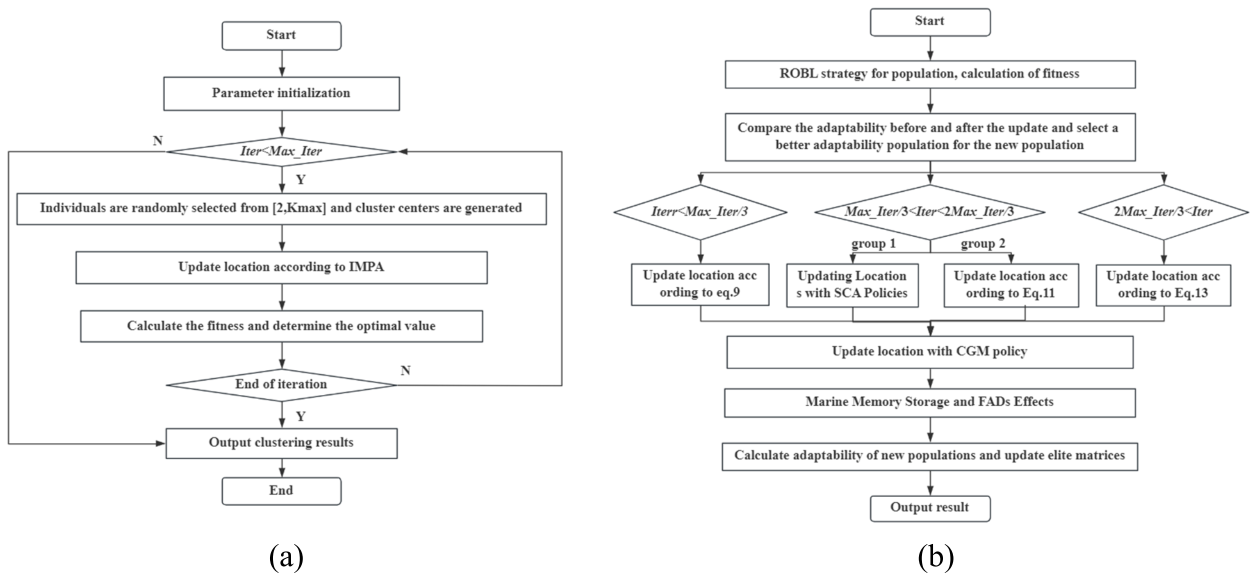

Start
Parameter initialization
Iter<Max_Iter
N
Y
Individuals are randomly selected from [2,Kmax] and cluster centers are generated
Update location according to IMPA
Calculate the fitness and determine the optimal value
End of iteration
N
Y
Output clustering results
End
(a)
Start
ROBL strategy for population, calculation of fitness
Compare the adaptability before and after the update and select a better adaptability population for the new population
Iterr<Max_Iter/3
Max_Iter/3<Iter<2Max_Iter/3
2Max_Iter/3<Iter
group 1
group 2
Update location according to eq.9
Updating Locations with SCA Policies
Update location according to Eq.11
Update location according to Eq.13
Update location with CGM policy
Marine Memory Storage and FADs Effects
Calculate adaptability of new populations and update elite matrices
Output result
(b)

Figure 4. (a) Overall flow diagram of automatic clustering. (b) Overall flow diagram of IMPA.

4. Experimental results and discussion

This section is divided into two parts. One is the result of IMPA in test functions, and the other is the application of AC-IMPA in common datasets and rotating fault datasets. The simulation results and discussion of the proposed algorithm are introduced and compared with other results in the literature. The experimental running environment is a 64-bit Windows 11 operating system, the CPU is an Intel(R) Core (TM) i7-12650H, and the main frequency is 2.3 GHz. The algorithm is written based on MATLAB R2021b.

4.1. Test functions

This section uses test functions to evaluate the performance of the proposed IPA model. The experiments were performed on 10 standard benchmark functions, including unimodal, multimodal, and fixed dimension multimodal. The unimodal test functions (F1–F4) are used to test the exploitation ability of the algorithm, while multimodal functions (F5–F19) are used to evaluate the exploration performance. The fixed dimension test function (F10) shows the exploration capability in low dimensions [41]. The detailed descriptions and related information are given in Table 1. To ensure fairness of the comparison between algorithms, the basic parameters of the algorithms are set to the same values, including the population size n = 30, the maximum number of iterations $Max\_Iter = 500$, and the dimensionality of the test functions d =30.

To verify the overall performance of IMPA, six algorithms are selected for comparison, including the basic MPA, the nonlinear marine predator algorithm (NMPA) [42], the quasi-opposition learning and the Q-learning based marine predators algorithm (QQLMPA) [43], WOA [44], the pelican optimization algorithm (POA) [45] and particle swarm optimization (PSO) [46]. The internal parameters of each basic algorithm are set as shown in Table 2. All improved MPA algorithms have the same parameters as MPA.

Table 1. The benchmark functions.

Fun No.	Name	Dim	Range	Optimal Value
F1	Sphere	30	[−50,100]	0
F2	Schwefel 2.22	30	[−10,7]	0
F3	Rosenbrock	30	[−30,30]	0
F4	Quartic	30	[−1.28,1]	0
F5	Rastrigin	30	[−10,5.12]	0
F6	Ackley	30	[−32,50]	0
F7	Griewank	30	[−600,500]	0
F8	Alpine	30	[−10,40]	0
F9	Penalized	30	[−50,100]	0
F10	Schaffer	2	[−10,10]	0

*Note: See Appendix for the specific formula

Table 2. Algorithm parameter setting.

Algorithm	Parameter	Value
MPA	constant number *P*	0.5
	the probability of *FADs* effect on the optimization process *FADs*	0.2
IMPA	refractive index *q*	1000
NMPA	topology fully connected Inertia factor	0.5
QQLMPA	Discount factor	0.8
POA	constant *R*	0.2
	random number *I*	1 or 2
WOA	Convergence constant *a*	[2,0]
PSO	constriction factor $c_1 = c_2$	1.496
	inertia weight *w*	0.9
K-MEANS	distance	Euclidean distance
SSA	the alarm value	[0,1]
	the safety threshold	[0.5,1]

Table 3 shows the comparison results of 7 algorithms such as IMPA, MPA, QQLMPA on the benchmark functions. Figure 5 shows the convergence curves of these meta-heuristic algorithms. According to the experimental results in Table 3 and Figure 5, it can be seen that IMPA and NMPA can achieve the theoretically optimal value in three different evaluation metrics for two unimodal test functions F1–F2 and one multimodal function F8. As seen from the figure, IMPA can reach the ideal value before 150 iterations, indicating the effectiveness of the ROBL strategy. However, for F3 and F4, it is pretty challenging to find the global optimal solution. F3 is known as the banana-type valley function, and the shape of F4 is parabolic. They both have many local optimizations, which can easily lead the algorithm to falling into local optima and stagnate the optimization search. In F5 and F8, all three metrics of the algorithm are 0 except for WOA and PSO. F6 is characterized by an almost flat outer region with a large hole in the center. This function poses a risk for optimization algorithms to be trapped in one of its many local minima. All three improved MPA algorithms have a mean value of 8.88×10^{-16} and a mean difference of 0. However, IMPA can approach the ideal optimum faster in early iterations. There are two round spikes outside the optimal value of F10, and the result is a large fluctuation between the locally optimal position and the globally optimal position. IMPA can find the optimal point for F12 quickly. Generally, for other different types of test functions, IMPA is at the bottom of its iterative curve most of the time. It shows high convergence efficiency and verifies the effectiveness of the algorithm optimization strategy.

Table 3. Results of comparison with other basic algorithms.

Fun No.		MPA	IMPA	NMPA	QQLMPA	POA	WOA	PSO
	Best	1.09×10^{-24}	**0**	0	1.27×10^{-110}	2.21×10^{-116}	2.30×10^{-87}	2.30×10^{-02}
F1	Ave	4.86×10^{-23}	**0**	0	2.76×10^{-76}	3.28×10^{-99}	9.93×10^{-69}	6.84×10^{-02}
	Std	9.74×10^{-23}	**0**	0	4.30×10^{-76}	1.71×10^{-98}	5.44×10^{-68}	2.82×10^{-02}
	Best	1.07×10^{-14}	**0**	0	5.53×10^{-56}	4.07×10^{-59}	2.87×10^{-56}	4.89×10^{-01}
F2	Ave	4.56×10^{-13}	**0**	0	8.33×10^{-41}	7.45×10^{-51}	1.49×10^{-51}	7.75×10^{-01}
	Std	4.00×10^{-13}	**0**	0	1.70×10^{-40}	3.50×10^{-50}	3.91×10^{-51}	2.21×10^{-01}
	Best	$2.45 \times 10^{+01}$	**$2.37 \times 10^{+01}$**	$2.44 \times 10^{+01}$	$2.43 \times 10^{+01}$	$2.64 \times 10^{+01}$	$2.72 \times 10^{+01}$	$2.92 \times 10^{+01}$
F3	Ave	$2.53 \times 10^{+01}$	**$2.47 \times 10^{+01}$**	$2.51 \times 10^{+01}$	$2.57 \times 10^{+01}$	$2.80 \times 10^{+01}$	$2.79 \times 10^{+01}$	$1.03 \times 10^{+02}$
	Std	4.34×10^{-01}	6.78×10^{-01}	6.95×10^{-01}	7.39×10^{-01}	7.84×10^{-01}	**4.24×10^{-01}**	$1.01 \times 10^{+02}$
	Best	1.19×10^{-04}	1.41×10^{-06}	**4.07×10^{-07}**	1.13×10^{-04}	3.42×10^{-05}	1.35×10^{-04}	7.87×10^{-02}
F4	Ave	1.31×10^{-03}	**5.55×10^{-05}**	1.29×10^{-04}	4.96×10^{-04}	1.95×10^{-04}	1.98×10^{-03}	1.74×10^{-01}
	Std	7.97×10^{-04}	**4.60×10^{-05}**	1.16×10^{-04}	2.59×10^{-04}	1.07×10^{-04}	1.64×10^{-03}	7.56×10^{-02}
	Best	0	**0**	0	0	0	0	$3.96 \times 10^{+01}$
F5	Ave	0	**0**	0	0	0	0	$6.91 \times 10^{+01}$
	Std	0	**0**	0	0	0	0	$1.93 \times 10^{+01}$
	Best	2.53×10^{-13}	**8.88×10^{-16}**	8.88×10^{-16}	8.88×10^{-16}	8.88×10^{-16}	8.88×10^{-16}	$8.74 \times 10^{+00}$
F6	Ave	1.33×10^{-12}	**8.88×10^{-16}**	8.88×10^{-16}	8.88×10^{-16}	3.61×10^{-15}	4.80×10^{-15}	$1.35 \times 10^{+01}$
	Std	8.76×10^{-13}	**0**	0	0	1.53×10^{-15}	3.00×10^{-15}	$1.74 \times 10^{+00}$
	Best	0	**0**	0	0	0	0	$1.49 \times 10^{+01}$
F7	Ave	0	**0**	0	0	0	0	$3.93 \times 10^{+01}$
	Std	0	**0**	0	0	0	0	$2.01 \times 10^{+01}$
	Best	0	**0**	0	0	0	0	$1.49 \times 10^{+01}$
F8	Ave	0	**0**	0	0	0	0	$3.93 \times 10^{+01}$
	Std	0	**0**	0	0	0	0	$2.01 \times 10^{+01}$
	Best	**3.28×10^{-09}**	5.02×10^{-07}	9.68×10^{-05}	1.15×10^{-04}	5.92×10^{-02}	4.54×10^{-03}	$7.85 \times 10^{+00}$
F9	Ave	5.83×10^{-05}	**9.12×10^{-06}**	8.37×10^{-03}	1.12×10^{-02}	1.82×10^{-01}	$2.60 \times 10^{+02}$	$1.48 \times 10^{+01}$
	Std	2.88×10^{-04}	**1.38×10^{-05}**	6.56×10^{-04}	8.61×10^{-03}	1.09×10^{-01}	2.57×10^{-02}	$5.36 \times 10^{+00}$
	Best	0	**0**	0	0	0	0	3.33×10^{-16}
F10	Ave	1.95×10^{-08}	**0**	0	0	0	4.21×10^{-03}	6.48×10^{-03}
	Std	1.06×10^{-07}	**0**	0	0	0	4.90×10^{-03}	4.66×10^{-03}

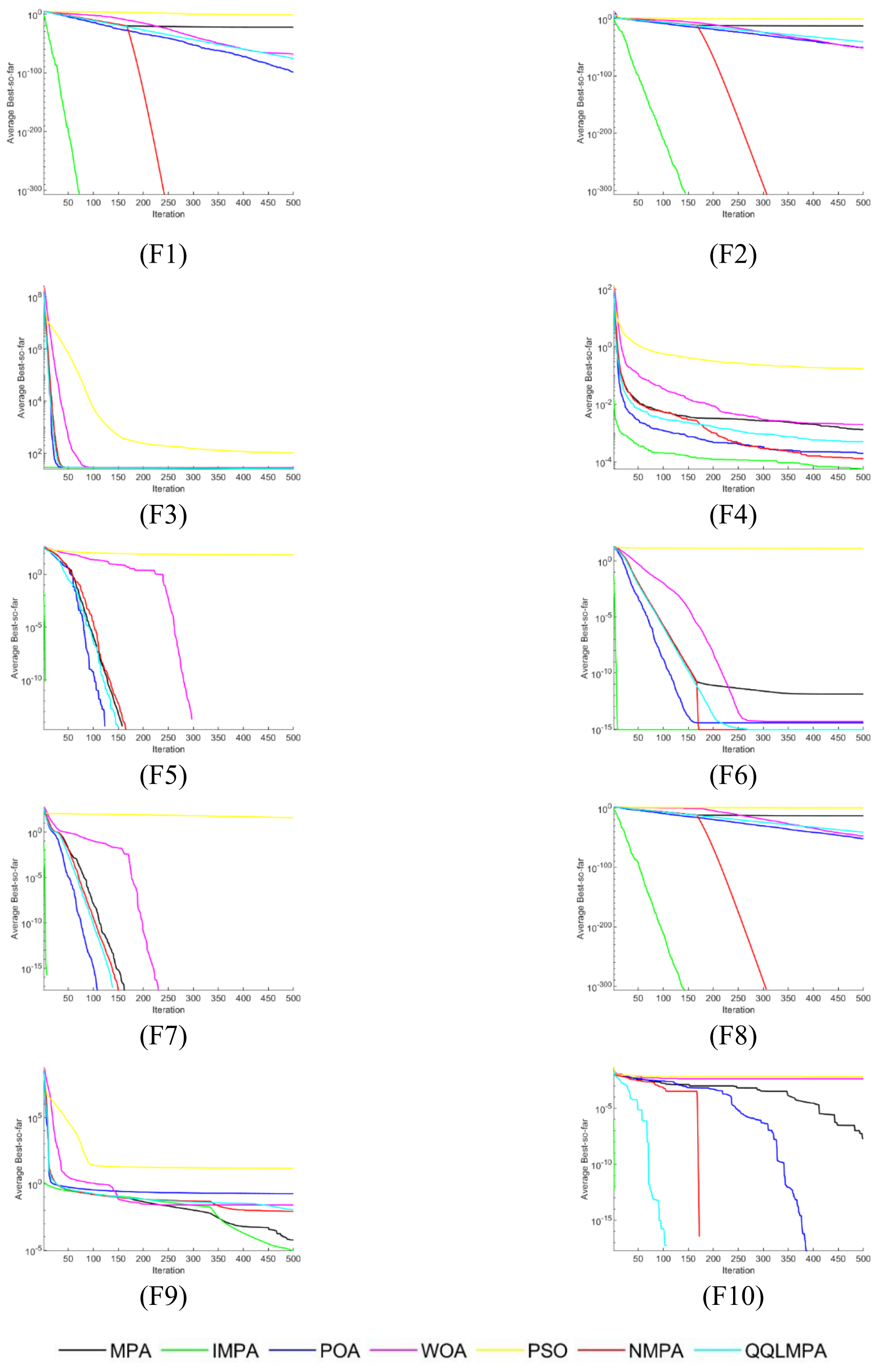


Figure 5. Convergence curves of IMPA and other metaheuristic algorithms.

The Wilcoxon signed rank test is used to demonstrate the validity of the improved algorithm. Let the best algorithm be IMPA. Paired comparisons are made between RSGMPA and MPA, POA, WOA, PSO, NMPA and QQLMPA with the significance level of p = 5%. The symbols "+", "–", and "=" indicate that IMPA performs better, worse and equivalent. As shown in Table 4, most of the p values among the 10 benchmark functions are less than 5%. This shows that the superiority of the algorithm is statistically significant. In other words, the convergence accuracy of the IMPA algorithm is higher than other algorithms [47].

Table 4. p Values for Wilcoxon signed rank test.

Fun. No.	MPA	NMPA	QQLMPA	POA	WOA	PSO
1	1.26×10^{-83}	1.89×10^{-41}	1.26×10^{-83}	1.26×10^{-83}	1.26×10^{-83}	1.26×10^{-83}
2	1.26×10^{-83}	4.36×10^{-52}	1.26×10^{-83}	1.26×10^{-83}	1.26×10^{-83}	1.26×10^{-83}
3	8.04×10^{-59}	1.26×10^{-83}	1.26×10^{-83}	1.26×10^{-83}	1.26×10^{-83}	1.26×10^{-83}
4	1.26×10^{-83}	1.26×10^{-83}	1.26×10^{-83}	1.24×10^{-83}	1.26×10^{-83}	1.26×10^{-83}
5	1.12×10^{-27}	7.91×10^{-29}	2.30×10^{-26}	6.32×10^{-22}	1.88×10^{-50}	1.26×10^{-83}
6	1.16×10^{-83}	1.20×10^{-29}	1.52×10^{-45}	3.49×10^{-90}	6.56×10^{-85}	1.26×10^{-83}
7	1.69×10^{-28}	1.58×10^{-26}	1.48×10^{-24}	1.28×10^{-19}	1.20×10^{-39}	1.26×10^{-83}
8	1.26×10^{-83}	6.35×10^{-52}	1.26×10^{-83}	1.26×10^{-83}	1.26×10^{-83}	1.26×10^{-83}
9	8.60×10^{-07}	2.55×10^{-29}	6.74×10^{-63}	1.26×10^{-83}	3.34×10^{-25}	1.26×10^{-83}
10	1.26×10^{-83}	5.62×10^{-30}	1.86×10^{-19}	2.49×10^{-65}	1.25×10^{-83}	1.26×10^{-83}
+/–/=	10/0/0	10/0/0	10/0/0	10/0/0	10/0/0	10/0/0

In addition, in order to make the statistical test results more convincing, we performed Friedman tests on the mean values of seven algorithms calculated on 10 benchmark functions. This method is mainly used to detect whether there are differences in the performances of different algorithms. The significance level was set to 0.05. When the p-value is less than 0.05, several algorithms can be considered to have statistically significant differences. Conversely, there is no statistically significant difference between algorithms [48]. According to the calculation results in Table 5, the p-value are less than 0.05, indicating that there is a significant difference between the seven algorithms.

Table 5. Friedman tests.

Algorithm	Rank Mean	Rank
MPA	4.5909	5
IMPA	1.7727	1
NMPA	2.2273	2
QQLMPA	3.9091	4
POA	3.8182	3
WOA	4.9091	6
PSO	6.7727	7
p	2.86×10^{-08}	

4.2. Datasets

To verify the overall performance of AC-IMPA, six algorithms were selected for comparison, including K-means, FCM [49], basic MPA, PSO, WOA and SSA [50]. To ensure the fairness of the comparison between the meta-heuristic algorithms, the basic parameters of the algorithms are set to the same values, population size $n = 30$, and maximum number of iterations $Max_{Iter} = 100$.

A total of 7 datasets were used, including 1 Labor dataset and 6 UCI datasets. Table 6 gives the summary of the 7 datasets, showing the data type, data set dimension, number of data points and number of clusters. UCI data sets are real classical data sets from different fields, and the selected data dimensions are different.

Table 6. Data set details.

Data Type	Data	Data dimension	Amount of data	Number of clusters	Citations
Labor	Data 1	2	299	3	[139,99,61]
UCI	Wine	13	178	3	[59,71,48]
	Balance scale	4	625	3	[49288288]
	Cancer	9	683	2	[444239]
	Jain	2	373	2	[276,97]
	Thyroid	5	215	3	[150,35,30]
	Haberman	3	306	2	[225,81]

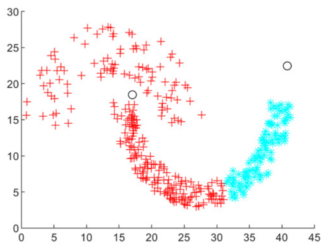
Tables 7 and 8 show the results of 50 independent runs of these algorithms based on CS. Table 6 shows the average and standard deviation of CS measurements, and Table 7 shows the cluster numbers and standard deviations determined by CS. Figure 6 shows the clustering results of MPA and IMPA on data 1. The data is shaped like two moons, representing the actual classifications. For the data at the intersection of two moons, IMPA has a better classification effect. In this paper, cluster numbers are compared first, followed by CS values. In the Cancer dataset, the numbers of clusters of K-means and AC-IMPA match the actual and are highly stable. In the Jain dataset, AC-IMPA and AC-MPA have the best cluster numbers. In the remaining 5 datasets, AC-IMPA performed best. For the values of CS table, AC-IMPA has the best average value in four datasets, Cancer, Jain, Thyroid, Haberman, and the other datasets rank high, indicating AC-IMPA's superior performance.

Table 7. Number of clusters.

		K-MEANS	AC-MPA	AC-IMPA	AC-WOA	AC-SSA	AC-PSO	FCM
Data 1	Ave	3	3.04	3	2.9	3.22	3.08	6.4
	SD	0	0.20	0	0.36	0.89	0.27	0.55
Wine	Ave	8.54	3.7	3.12	5.62	4.38	3.36	7.8
	SD	1.18	1.11	0.48	1.61	1.89	0.94	0.84
Balance	Ave	8.96	3.32	2.78	5.74	6.8	8.66	8.8
scale	SD	0.97	2.24	1.25	3.51	2.89	0.87	1.79
Cancer	Ave	2	2.1	2	2.02	2.86	2.08	2
	SD	0	0.30	0	0.14	1.98	0.44	0
Jain	Ave	7.86	2.34	2	2	3.32	5.42	8.4
	SD	0.83	0.87	0	0	1.43	1.94	0.55
Thyroid	Ave	8.58	3.16	3.08	2.54	2.76	3.84	8
	SD	1.37	0.47	0.40	1.09	1.04	1.04	0
Haberman	Ave	7.78	2	2	4.3	3.52	4.86	4
	SD	1.54	0	0	2.13	1.50	1.812	0

Table 8. CS values.

		K-MEANS	AC-MPA	AC-IMPA	AC-WOA	AC-SSA	AC-PSO	FCM
Data 1	Ave	0.6059	**0.6057**	0.6059	0.6428	0.6059	0.6062	1.2030
	SD	0	1.04×10^{-03}	0	6.05×10^{-02}	0	2.29×10^{-03}	3.17×10^{-02}
Wine	Ave	0.6095	**0.3947**	0.3960	0.4527	0.3976	0.3977	0.7313
	SD	1.27×10^{-01}	2.55×10^{-03}	1.53×10^{-03}	2.52×10^{-02}	9.81×10^{-03}	7.43×10^{-03}	1.20×10^{-02}
Balance scale	Ave	1.35078	1.0058	**0.9694**	1.1229	1.1121	1.3558	1.7521
	SD	3.22×10^{-02}	1.83×10^{-01}	1.54×10^{-01}	2.72×10^{-01}	2.10×10^{-01}	3.52×10^{-02}	1.22×10^{-01}
Cancer	Ave	1.0973	0.9592	**0.9166**	1.02334	0.9617	1.0855	1.0906
	SD	4.99×10^{-04}	1.23×10^{-01}	1.60×10^{-01}	1.19×10^{-01}	1.35×10^{-01}	7.88×10^{-02}	0
Jain	Ave	0.8725	**0.6413**	0.6547	0.65467	0.6481	0.7013	0.8764
	SD	1.47×10^{-02}	3.56×10^{-02}	5.83×10^{-16}	5.61×10^{-16}	1.95×10^{-02}	4.98×10^{-02}	6.23×10^{-03}
Thyroid	Ave	0.8873	0.2927	**0.2924**	0.3177	0.30189	0.3025	1.1892
	SD	8.13×10^{-02}	4.75×10^{-03}	6.15×10^{-03}	2.87×10^{-02}	9.15×10^{-03}	2.22×10^{-02}	0
Haberman	Ave	1.1072	**0.4674**	**0.4674**	0.5620	0.4825	0.524	1.4517
	SD	7.44×10^{-02}	5.62×10^{-16}	5.62×10^{-16}	9.09×10^{-02}	3.89×10^{-02}	5.59×10^{-02}	0


(a) MPA

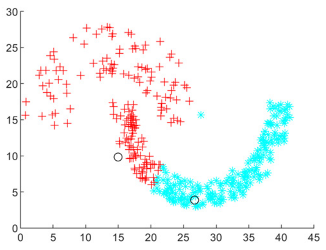

(b) IMPA

Figure 6. Clustering results of MPA and IMPA on Data 1.

The Wilcoxon signed rank test is used to demonstrate the validity of the improved algorithm. Let the best algorithm be AC-IMPA. Paired comparisons are made between AC-IMPA and AC-MPA, AC-WOA, AC-SSA and AC-PSO with the significance level of $p = 5\%$. As shown in Table 9, most of the p values among the 7 datasets are less than 5%. This shows that the superiority of the algorithm is statistically significant.

Table 9. p Values for Wilcoxon signed rank test.

	AC-MPA	**AC-WOA**	**AC-SSA**	**AC-PSO**
Data 1	4.78×10^{-09}	4.55×10^{-19}	1.21×10^{-05}	3.10×10^{-18}
Wine	1.33×10^{-02}	3.28×10^{-18}	2.71×10^{-07}	3.78×10^{-18}
Balance scale	2.71×10^{-17}	1.67×10^{-02}	2.53×10^{-05}	3.90×10^{-18}
Cancer	1.17×10^{-17}	1.14×10^{-17}	5.00×10^{-03}	3.90×10^{-18}
Jain	2.19×10^{-11}	$\mathbf{1.22 \times 10^{-01}}$	1.02×10^{-02}	3.90×10^{-18}
Thyroid	1.06×10^{-04}	3.82×10^{-18}	5.04×10^{-14}	3.88×10^{-18}
Haberman	7.23×10^{-04}	3.74×10^{-18}	4.22×10^{-18}	3.88×10^{-18}
+/−/=	7/0/0	6/1/0	7/0/0	7/0/0

4.3. XJTU-SY rolling bearing accelerator life data set of Xi'an Jiaotong University

The bearing experiment data set of Xi'an Jiaotong University was used as the data [51]. In the experiment, the sampling frequency was set at 25.6 kHz, the sampling interval was set at 1 min, and the sampling duration was set at 1.28 s each time. The bearing speed is 2100 r/min, and the radial force is 12 kN. Bearing1_1, Bearing1_4, and Bearing1_5 in the data set are used. The detailed data are described in Table 10.

Table 10. Bearing data information.

Data set	sample total	actual lifetime	failure location
Bearing1_1	123	2 h 3 min	External ring
Bearing1_4	122	2 h 2 min	Cage
Bearing1_5	52	52 min	External ring, Inner ring

Table 11 shows the adaptive cluster values and CS values of 7 algorithms on the XJTU-SY rolling bearing data set. It can be seen from the table that only the cluster numbers of AC-MPA, AC-IMPA and AC-SSA algorithms are close to the optimal category number of the data set. When K = 3, the change curves of the fitness functions of the three algorithms are shown in Figure 7, which once again verifies that the improved algorithm has a relatively high degree of accuracy in solving. Meanwhile, in order to demonstrate the effectiveness of the proposed algorithm, when K = 3, the three algorithms are run independently 10 times, and the recognition rates of all samples are shown in Table 12. It can be seen that AC-IMPA has the highest recognition rate for cluster analysis of such complex data. Although the overall recognition rate of AC-MPA algorithm is also up to 83.33%, which is close to this, the results of 10 runs show that its recognition results are less stable than the algorithm proposed in this paper.

Table 11. Clustering result.

		K-MEANS	AC-MPA	AC-IMPA	AC-WOA	AC-SSA	AC-PSO	AC-FCM
clusters	Ave	9.44	3	2.98	2.02	2.9	6.04	9.6
	SD	0.81	0.95	0.47	0.14	1.90	2.15	0.52
CS values	Ave	1.5089	0.6595	0.6335	0.7602	0.7277	0.9168	1.6050
	SD	3.51×10^{-02}	7.42×10^{-02}	5.72×10^{-02}	1.82×10^{-02}	4.25×10^{-02}	9.08×10^{-02}	1.66×10^{-02}

Table 12. Comparison of recognition rate results of the three algorithms.

category	algorithm		
	AC-MPA	AC-IMPA	AC-SSA
Fault 1	13/16 = 0.8125	16/16 = 1	9/16 = 0.5625
Fault 2	16/16 = 1	16/16 = 1	2/16 = 0.125
Fault 3	11/16 = 0.6875	16/16 = 1	13/16 = 0.1875
Entirety	83.33%	100%	56.94%

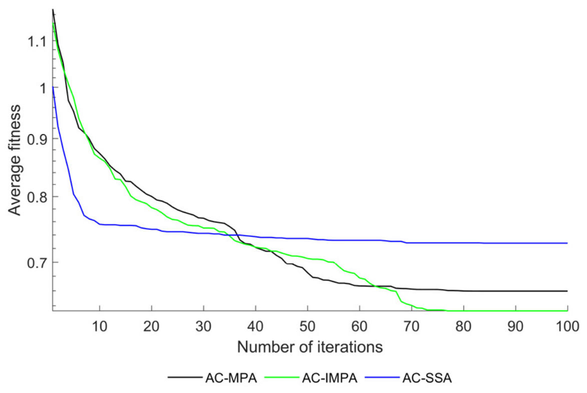


Figure 7. Convergence images of the three algorithms.

5. Conclusions and future research directions

In order to solve the clustering problem of bearing fault datasets more effectively, a new automatic clustering algorithm (AC-IMPA) based on improved MPA and K-means algorithm is proposed. In this algorithm, the improved MPA algorithm aims to provide a better initial center of mass for the K-means algorithm, that is, the predator is closer to the center of the data and is able to use this information to move faster and find a better solution than the original predator's movement. For validation, the proposed AC-IMPA algorithm is tested using some benchmark datasets and compared with some automatic clustering meta-heuristic algorithm algorithms, AC-MPA, AC-SSA, FCM, K-MEANS and AC-WOA. The results show that AC-IMPA can obtain better and more stable results in cluster number. Finally, AC-IMPA is applied to the problem of automatic classification of bearing fault, and it is found that the number of clusters obtained by the proposed new method is closer to the true facts.

Although the proposed automatic clustering method AC-IMPA shows advantages in the experiment, it also has some weaknesses. First, the advantage of cluster quality is not obvious on high cluster number data sets, because the number of clusters is small, and once the data has errors, it is easy to limit the performance of the algorithm. Another is that the clustering stability of AC-IMPA is not strong enough. In addition, in future research, more, other clustering effectiveness indicators can be attempted for automatic clustering to make the clustering effect better. Therefore, AC-IMPA provides a new and effective automatic clustering mode, but it still needs in-depth analysis and further improvement.

Use of AI tools declaration

The authors declare they have not used artificial intelligence (AI) tools in the creation of this article.

Acknowledgments

This research was funded by the Key Project of Scientific Research of Anhui Provincial Education Department, China, grant No. 2022AH050995 and No. 2022AH050975. It was supported by Anhui Province Intelligent Mine Technology and Equipment Engineering Laboratory Open Fund, grant No.

AIMTEEL202201; Project of Science and Technology Bureau of Wuhu City, grant No. KZ32022014; Open Fund of Anhui Key Laboratory of Electric Transmission and Control, grant No. DQKJ202208; Research Project of Anhui Polytechnic University, grant No. Xjky2022007; Research Start-up Fund Project of Anhui Polytechnic University, grant No. 2022YQQ004; Applied Basic Research Project of Wuhu City, grant No. 2022jc20.

Conflict of interest

The authors declare there is no conflict of interest.

References

1. P. Liang, L. Xu, H. Shuai, X. Yuan, B. Wang, L. Zhang, Semisupervised subdomain adaptation graph convolutional network for fault transfer diagnosis of rotating machinery under time-varying speeds, *IEEE/ASME Trans. Mechatron.*, (2023), 1–2. https://doi.org/10.1109/TMECH.2023.3292969
2. S. Gawde, S. Patil, S. Kumar, P. Kamat, K. Kotecha, A. Abraham, Multi-fault diagnosis of Industrial Rotating Machines using Data-driven approach: A review of two decades of research, *Eng. Appl. Artif. Intell.*, **123** (2023), 106139. https://doi.org/10.1016/j.engappai.2023.106139
3. X. Xu, S. Hu, P. Shi, H. Shao, R. Li, Z. Li, Natural phase space reconstruction-based broad learning system for short-term wind speed prediction: Case studies of an offshore wind farm, *Energy*, **262** (2023), 125342. https://doi.org/10.1016/j.energy.2022.125342
4. M. Liang, K. Zhou, Probabilistic bearing fault diagnosis using Gaussian process with tailored feature extraction, *Int. J. Adv. Manuf. Technol.*, **119** (2022), 2059–2076. https://doi.org/10.1007/s00170-021-08392-6
5. P. Liang, W. Wang, X. Yuan, S. Liu, L. Zhang, Y. Cheng, Intelligent fault diagnosis of rolling bearing based on wavelet transform and improved ResNet under noisy labels and environment, *Eng. Appl. Artif. Intell.*, **115** (2022), 105269. https://doi.org/10.1016/j.engappai.2022.105269
6. Y. Xu, Z. Li, S. Wang, W. Li, T. Sarkodie-Gyan, S. Feng, A hybrid deep-learning model for fault diagnosis of rolling bearings, *Measurement*, **169** (2021), 108502. https://doi.org/10.1016/j.measurement.2020.108502
7. H. Yuan, Y. Tang, H. Hao, Y. Zhao, Y. Zhang, Y. Chen, Intelligent detection method of gearbox based on adaptive hierarchical clustering and subset, *Comput. Intell. Neurosci.*, **2022** (2022), 6464516. https://doi.org/10.1155/2022/6464516
8. J. Hou, Y. Wu, H. Gong, A. S. Ahmad, L. Liu, A novel intelligent method for bearing fault diagnosis based on EEMD permutation entropy and GG clustering, *Appl. Sci.*, **10** (2020), 386. https://doi.org/10.3390/app10010386
9. Y. Cheng, W. Jia, R. Chi, A. Li, A clustering analysis method with high reliability based on wilcoxon-mann-whitney testing, *IEEE Access*, **9** (2021), 19776–19787. https://doi.org/10.1109/ACCESS.2021.3053244
10. M. A. Mahdi, K. M. Hosny, I. Elhenawy, Scalable clustering algorithms for big data: A review, *IEEE Access*, **9** (2021), 80015–80027. https://doi.org/10.1109/ACCESS.2021.3084057

11. X. Xu, S. Hu, H. Shao, P. Shi, R. Li, D. Li, A spatio-temporal forecasting model using optimally weighted graph convolutional network and gated recurrent unit for wind speed of different sites distributed in an offshore wind farm, *Energy*, **284** (2023), 128565. https://doi.org/10.1016/j.energy.2023.128565
12. K. He, X. Niu, X. Min, F. Min, ERCP: Speedup path planning through clustering and presearching, *Appl. Intell.*, **53** (2023), 12324–12339. https://doi.org/10.1007/s10489-022-04137-4
13. L. Abualigah, A. Diabat, Z. W. Geem, A comprehensive survey of the harmony search algorithm in clustering applications, *Appl. Sci.*, **10** (2020), 3827. https://doi.org/10.3390/app10113827
14. A. M. Ikotun, A. E. Ezugwu, L. Abualigah, B. Abuhaija, J. Heming, K-means clustering algorithms: A comprehensive review, variants analysis, and advances in the era of big data, *Inf. Sci.*, **622** (2023), 178–210. https://doi.org/10.1016/j.ins.2022.11.139
15. H. Xue, Z. Song, M. Wu, N. Sun, H. Wang, Intelligent diagnosis based on double-optimized artificial hydrocarbon networks for mechanical faults of in-wheel motor, *Sensors*, **22** (2022), 6316. https://doi.org/10.3390/s22166316
16. C. Mariela, S. René-Vinicio, C. Diego, A semi-supervised approach based on evolving clusters for discovering unknown abnormal condition patterns in gearboxes, *J. Intell. Fuzzy Syst.*, **34** (2018), 3581–3593. https://doi.org/10.3233/JIFS-169535
17. L. Wan, G. Zhang, H. Li, C. Li, A novel bearing fault diagnosis method using spark-based parallel aco-k-means clustering algorithm, *IEEE Access*, **9** (2021), 28753–28768. https://doi.org/10.1109/ACCESS.2021.3059221
18. A. M. Ikotun, M. S. Almutari, A. E. Ezugwu, K-means-based nature-inspired metaheuristic algorithms for automatic data clustering problems: Recent advances and future directions, *Appl. Sci.*, **11** (2021), 11246. https://doi.org/10.3390/app112311246
19. A. M. Ikotun, A. E. Ezugwu, Enhanced firefly-K-means clustering with adaptive mutation and central limit theorem for automatic clustering of high-dimensional datasets, *Appl. Sci.*, **12** (2022), 12275. https://doi.org/10.3390/app122312275
20. Y. Zhang, M. Martínez-García, R. S. Kalawsky, A. Latimer, Grey-box modelling of the swirl characteristics in gas turbine combustion system, *Measurement*, **151** (2020), 107266. https://doi.org/10.1016/j.measurement.2019.107266
21. C. Yang, H. Sutrisno, A clustering-based symbiotic organisms search algorithm for high-dimensional optimization problems, *Appl. Soft Comput.*, **97** (2020), 106722. https://doi.org/10.1016/j.asoc.2020.106722
22. A. Faramarzi, M. Heidarinejad, S. Mirjalili, A. H. Gandomi, Marine Predators Algorithm: A nature-inspired metaheuristic, *Expert Syst. Appl.*, **152** (2020), 113377. https://doi.org/10.1016/j.eswa.2020.113377
23. M. A. Al-Betar, M. A. Awadallah, S. N. Makhadmeh, Z. A. A. Alyasseri, G. Al-Naymat, S. Mirjalili, Marine predators algorithm: A review, *Arch. Comput. Methods Eng.*, **30** (2023), 3405–3435. https://doi.org/10.1007/s11831-023-09912-1
24. M. A. Elaziz, D. Mohammadi, D. Oliva, K. Salimifard, Quantum marine predators algorithm for addressing multilevel image segmentation, *Appl. Soft Comput.*, **110** (2021), 107598. https://doi.org/10.1016/j.asoc.2021.107598
25. M. Ramezani, D. Bahmanyar, N. Razmjooy, A new improved model of marine predator algorithm for optimization problems, *Arab. J. Sci. Eng.*, **46** (2021), 8803–8826. https://doi.org/10.1007/s13369-021-05688-3

26. J. Saha, J. Mukherjee, CNAK: Cluster number assisted K-means, *Pattern Recognit.*, **110** (2021), 107625. https://doi.org/10.1016/j.patcog.2020.107625
27. R. Ghezelbash, A. Maghsoudi, E. J. M. Carranza, Optimization of geochemical anomaly detection using a novel genetic K-means clustering (GKMC) algorithm, *Comput. Geosci.*, **134** (2020), 104335. https://doi.org/10.1016/j.cageo.2019.104335
28. A. Dey, S. Bhattacharyya, S. Dey, D. Konar, J. Platos, V. Snasel, et al., A review of quantum-inspired metaheuristic algorithms for automatic clustering, *Mathematics*, **11** (2023), 2018. https://doi.org/10.3390/math11092018
29. D. L. Davies, D. W. Bouldin, A cluster separation measure, *IEEE Trans. Pattern Anal. Mach. Intell.*, **2** (1979), 224–227. https://doi.org/10.1109/TPAMI.1979.4766909
30. J. C. Dunn, Well-separated clusters and optimal fuzzy partitions, *J. Cybern.*, **4** (1974), 95–104. https://doi.org/10.1080/01969727408546059
31. M. K. Pakhira, S. Bandyopadhyay, U. Maulik, Validity index for crisp and fuzzy clusters, *Pattern Recognit.*, **37** (2004), 487–501. https://doi.org/10.1016/j.patcog.2003.06.005
32. K. Zhou, Enhanced feature extraction for machinery condition monitoring using recurrence plot and quantification measure, *Int. J. Adv. Manuf. Technol.,* **123** (2022), 3421–3436. https://doi.org/10.1007/s00170-022-10392-z
33. C. Chou, M. Su, E. Lai, A new cluster validity measure and its application to image compression, *Pattern Anal. Appl.*, **7** (2004), 205–220. https://doi.org/10.1007/s10044-004-0218-1
34. A. Dey, S. Dey, S. Bhattacharyya, J. Platos, V. Snasel, Quantum inspired meta-heuristic approaches for automatic clustering of colour images, *Int. J. Intell. Syst.*, **36** (2021), 4852–4901. https://doi.org/10.1002/int.22494
35. H. R. Tizhoosh, Opposition-based learning: A new scheme for machine intelligence, in *International Conference on Computational Intelligence for Modelling, Control and Automation and International Conference on Intelligent Agents, Web Technologies and Internet Commerce (CIMCA-IAWTIC'06)*, (2005), 695–701. https://doi.org/10.1109/CIMCA.2005.1631345
36. X. Dong, Y. Liu, C. Deng, Improved differential evolution algorithm and its application in complex function optimization, in *The 26th Chinese Control and Decision Conference (2014 CCDC)*, (2014), 3698–3701. https://doi.org/10.1109/CCDC.2014.6852822
37. P. Shao, Z. Wu, X. Zhou, D. C. Tran, FIR digital filter design using improved particle swarm optimization based on refraction principle, *Soft Comput.*, **21** (2017), 2631–2642. https://doi.org/10.1007/s00500-015-1963-3
38. S. Mirjalili, SCA: A Sine Cosine Algorithm for solving optimization problems, *Knowl.-Based Syst.*, **96** (2016), 120–133. https://doi.org/10.1016/j.knosys.2015.12.022
39. Y. Liu, L. Ma, Sine cosine algorithm with nonlinear decreasing conversion parameter, *Comput. Eng. Appl.*, **53** (2017), 1–5.
40. X. Chen, A. Shen, Self-adaptive differential evolution with Gaussian–Cauchy mutation for large-scale CHP economic dispatch problem, *Neural Comput. Appl.*, **34** (2022), 11769–11787. https://doi.org/10.1007/s00521-022-07068-w
41. W. Yang, K. Xia, S. Fan, L. Wang, T. Li, J. Zhang, Y. Feng, A multi-strategy whale optimization algorithm and its application, *Eng. Appl. Artif. Intell.*, **108** (2022), 104558. https://doi.org/10.1016/j.engappai.2021.104558

42. A. S. Sadiq, A. A. Dehkordi, S. Mirjalili, Q. Pham, Nonlinear marine predator algorithm: A cost-effective optimizer for fair power allocation in NOMA-VLC-B5G networks, *Expert Syst. Appl.*, **203** (2022), 117395. https://doi.org/10.1016/j.eswa.2022.117395
43. S. Zhao, Y. Wu, S. Tan, J. Wu, Z. Cui, Y. Wang, QQLMPA: A quasi-opposition learning and Q-learning based marine predators algorithm, *Expert Syst. Appl.*, **213** (2023), 119246. https://doi.org/10.1016/j.eswa.2022.119246
44. S. Mirjalili, A. Lewis, The whale optimization algorithm, *Adv. Eng. Software*, **95** (2016), 51–67. https://doi.org/10.1016/j.advengsoft.2016.01.008
45. P. Trojovský, M. Dehghani, Pelican optimization algorithm: A novel nature-inspired algorithm for engineering applications, *Sensors*, **22** (2022), 855. https://doi.org/10.3390/s22030855
46. A. Khare, S. Rangnekar, A review of particle swarm optimization and its applications in Solar Photovoltaic system, *Appl. Soft Comput.*, **13** (2013), 2997–3006. https://doi.org/10.1016/j.asoc.2012.11.033
47. A. H. Elsheikh, M. A. Elaziz, Review on applications of particle swarm optimization in solar energy systems, *Int. J. Environ. Sci. Technol.*, **16** (2019), 1159–1170. https://doi.org/10.1007/s13762-018-1970-x
48. S. K. Sahoo, A. K. Saha, S. Nama, M. Masdari, An improved moth flame optimization algorithm based on modified dynamic opposite learning strategy, *Artif. Intell. Rev.*, **56** (2023), 2811–2869. https://doi.org/10.1007/s10462-022-10218-0
49. H. Wang, J. Wang, G. Wang, A survey of fuzzy clustering validity evaluation methods, *Inf. Sci.*, **618** (2022), 270–297, https://doi.org/10.1016/j.ins.2022.11.010
50. J. Xue, B. Shen, A novel swarm intelligence optimization approach: sparrow search algorithm, *Syst. Sci. Control Eng.*, **8** (2020), 22–34. https://doi.org/10.1080/21642583.2019.1708830
51. B. Wang, Y. Lei, N. Li, N. Li, A hybrid prognostics approach for estimating remaining useful life of rolling element bearings, *IEEE Trans. Reliab.*, **69** (2020), 401–412. https://doi.org/10.1109/TR.2018.2882682

Supplementary

Name	Formula
Sphere	$F_1(x) = \sum_{i=1}^{30} x_i^2$
Schwefel 2.22	$F_2(x) = \sum_{i=1}^{30}\lvert x_i\rvert + \prod_{i=1}^{30}\lvert x_i\rvert$
Rosenbrock	$F_3(x) = \sum_{i=1}^{29}[(1-x_i)^2 + 100(x_{i+1}-x_i^2)^2]$
Quartic	$F_4(x) = \sum_{i=1}^{30} ix_i^4 + random[0,1)$
Rastrigin	$F_5(x) = \sum_{i=1}^{30}[x_i^2 - 10\cos(2\pi x_i) + 10]$
Ackley	$F_6(x) = -20exp\left(-0.2\sqrt{\frac{1}{30}\sum_{i=1}^{30} x_i^2}\right) - exp\left(\frac{1}{30}\sum_{i=1}^{30}\cos 2\pi x_i\right) + 20 + e$
Griewank	$F_7(x) = \frac{1}{4000}\sum_{i=1}^{30} x_i^2 - \prod_{i=1}^{30}\cos\left(\frac{x_i}{\sqrt{i}}\right) + 1$
Alpine	$F_8(x) = \sum_{i=1}^{30}\lvert x_i\sin(x_i) + 0.1x_i\rvert$
Penalized	$F_9(x) = \frac{\pi}{30}\{10sin^2(\pi y_i) + \sum_{i=1}^{29}(y_i-1)^2[1+10sin^2(\pi y_{i+1})] + (y_n-1)^2\} + \sum_{i=1}^{30} u(x_i,10,100,4)$ $u(x_i,a,k,m) = \begin{cases} k(x_i-a)^m, & x_i > a, \\ 0 & ,-a \le x_i \le a, \\ k(-x_i-a)^m, x_i < -a \end{cases}$ $y_i = 1 + \frac{x_i+1}{4}$
Schaffer	$F_{10}(x) = 0.5 + \frac{sin^2(x_1^2-x_2^2)-0.5}{[1+0.001(x_1^2+x_2^2)]^2}$