Knowledge-Based Systems 144 (2018) 153–173

Contents lists available at ScienceDirect

Knowledge-Based Systems

journal homepage: www.elsevier.com/locate/knosys

A new improved fruit fly optimization algorithm IAFOA and its application to solve engineering optimization problems

Lei Wu[a,b], Qi Liu[a], Xue Tian[c], Jixu Zhang[d], Wensheng Xiao[a,*]

[a] College of Mechanical and Electronic Engineering, China University of Petroleum, Qingdao 266580, China
[b] School of Civil and Environmental Engineering, Maritime Institute @NTU, Nanyang Technological University, Singapore 639798, Singapore
[c] Marine Design & Research Institute of China, Shanghai 200011, China
[d] CNOOC Safety Technology Services Company Limited, Tianjin 300456, China

ARTICLE INFO

Article history:
Received 14 July 2017
Revised 24 November 2017
Accepted 27 December 2017
Available online 27 December 2017

Keywords:
Fruit fly optimization algorithm
Optimal search direction
Iteration step value
Crossover and mutation operations
Multi-sub-swarm
Engineering optimization problem

ABSTRACT

Nature-inspired algorithms are widely used in mathematical and engineering optimization. As one of the latest swarm intelligence-based methods, fruit fly optimization algorithm (FOA) was proposed inspired by the foraging behavior of fruit fly. In order to overcome the shortcomings of original FOA, a new improved fruit fly optimization algorithm called IAFOA is presented in this paper. Compared with original FOA, IAFOA includes four extra mechanisms: 1) adaptive selection mechanism for the search direction, 2) adaptive adjustment mechanism for the iteration step value, 3) adaptive crossover and mutation mechanism, and 4) multi-sub-swarm mechanism. The adaptive selection mechanism for the search direction allows the individuals to search for global optimum based on the experience of the previous iteration generations. According to the adaptive adjustment mechanism, the iteration step value can change automatically based on the iteration number and the best smell concentrations of different generations. Besides, the adaptive crossover and mutation mechanism introduces crossover and mutation operations into IAFOA, and advises that the individuals with different fitness values should be operated with different crossover and mutation probabilities. The multi-sub-swarm mechanism can spread optimization information among the individuals of the two sub-swarms, and quicken the convergence speed. In order to take an insight into the proposed IAFOA, computational complexity analysis and convergence analysis are given. Experiment results based on a group of 29 benchmark functions show that IAFOA has the best performance among several intelligent algorithms, which include five variants of FOA and five advanced intelligent optimization algorithms. Then, IAFOA is used to solve three engineering optimization problems for the purpose of verifying its practicability, and experiment results show that IAFOA can generate the best solutions compared with other ten algorithms.

1. Introduction

Due to the characteristics of high robust and excellent optimization ability, nature-inspired algorithms such as genetic algorithm (GA), artificial bee colony optimization (ABC), particle swarm optimization (PSO), bat algorithm (BA), and ant colony optimization (ACO) have been applied widely in solving mathematical and engineering problems [1]. As a novel evolutionary computation and optimization approach, fruit fly optimization algorithm (FOA) was proposed by Pan [2] in 2012 based on the foraging behavior of fruit fly. As we know, the fruit fly has obvious advantages over other creatures in olfactory and visual sensory perception, therefore, it can search for food easily [3]. Since FOA was presented, it has gained much attention and been successfully applied in many areas in recent years, such as continuous mathematical function optimization [4], design of tubular linear synchronous motor [5], optimization of flow shop rescheduling problem [6,7], web auction logistics service [8], medical diagnosis [9], forecasting power loads [10], confirming neural network parameters [11], as well as many other problems in scientific and engineering fields [12]. Many researches have proven that FOA has significant advantages in terms of convergence and robustness [3].

However, similar with other nature-inspired algorithms, FOA also has its own shortcomings. For example, it often derives a local extreme when solving high-dimensional functions and large-scale combinational optimization problems [3]. For the purpose of improving the search efficiency and global search ability, many variants of FOA were designed [13]. According to the improved points

* Corresponding author.
E-mail address: xiaows@upc.edu.cn (W. Xiao).

https://doi.org/10.1016/j.knosys.2017.12.031
0950-7051/

the researchers focus on, these variants can be divided into several categories.

First, some researchers proposed new mechanisms to adjust the search scope of the fruit flies. Actually, the search scope depends on the search radius (or called iteration step value) which is a fixed value in original FOA. Pan et al. [4] put forward an improved fruit fly optimization (IFFO), in which a new control parameter is introduced to tune the search scope around swarm location adaptively. In detail, after setting the maximum and minimum radiuses of search scope, the iteration step value is decreasing along with the increase of iteration number. Then, another improvement, which mainly focus on setting the maximum and minimum radiuses and a random value on interval [0, 1], was introduced by Liu et al. [14]. Zuo et al. [15] also presented an adaptive strategy in which a standard deviation δ is used to represent the search range. The standard deviation δ is equal to the product of $log(m+1)/m^{\alpha}$ and $(X_i - X_b)$ (m is the iteration number, α=2, X_i is the position of the i-th individual, X_b is the best location of the current population). Recently, a new method which introduces a variation coefficient and a disturbance coefficient was proposed by Xu et al. [16]. According to this method, the individual whose fitness value is lower than a certain value will search for appropriate solution in a bigger range during every iterative procedure, and if the best location of the fruit fly swarm remains unchanged for several iterations, the disturbance coefficient will take effect and the individual will search for optimum in a much bigger scope (10 times of the original search scope). What's more, Hu et al. [17] introduced fruit fly optimization algorithm with decreasing step (SFOA). In SFOA, the current step value R_i can be calculated according to a formula $R_i = R - R/(1 + e^{(6-12m/M)})$, in which R is the initial step value, M is maximum iteration number, m is the current iteration number.

Second, multi-swarm mechanism is another research topic for FOA. Due to its wonderful performance, multi-swarm approach have been applied in many intelligent algorithms. Thus, many researchers combined multi-swarm mechanism with FOA and obtained good results. Yuan et al. [18] presented multi-swarm FOA (MFOA). In MFOA, the swarm is split into several sub-swarms (usually 4 to 10), and then the sub-swarms move independently in the search space with the aim of simultaneously exploring global optimum. However, there is no information exchange between these sub-swarms. A bimodal mechanism was introduced by Wu et al. [19]. The idea of bimodal mechanism was borrowed from the labor division of natural swarm. According to the fruit fly's osphretic and visual functions in foraging process, the fruit fly population can be divided into search group and discovery group. Wang et al. [13] proposed a concept named "swarm collaboration", a certain proportion of individuals whose fitness values are better than the average fitness value, are allowed to gather towards the current optimum location, and the others fly randomly in the initial search region without the loss of generality. Besides, another concept named multi-population parallel computing was presented by Li et al. [20]. In this concept, the whole fruit fly swarm is divided into four sub-populations, and each sub-population evolves separately and the elitism strategy is introduced to preserve the best individuals. Niu et al. [21] proposed a novel FOA which divides the fruit fly swarm into two parts: one part is used to search for the food within a small range close to optimal solution; the other part is used to do the searching work within a larger scope, avoiding falling into local optimum.

Third, in order to improve the ability of jumping out the local optimum, many researchers introduced mutation operation for FOA. Zhang et al. [22] proposed a novel multi-scale cooperative mutation fruit fly optimization algorithm (MSFOA) which employs multi-scale cooperative mutation and the Gaussian mutation operator. Wang and Liu [23] presented adaptive mutation fruit fly optimization algorithm (AM-FOA) which selects a corresponding number of fruit files from the population and makes mutation and then updates the global optimum when the algorithm trapping in local optimum. Ye et al. [24] introduced a mutation probability rate mr (set to 0.8) to allow some individuals to search for optimum in a bigger range. Niu et al. [21] used Cauchy mutation to make fruit fly variants for the sake of improving the convergence performance and optimization capabilities of the algorithm. Pan [25] proposed a modified fruit fly optimization algorithm (MFOA) by introducing an escape parameter for the fitness function for the purpose of escaping from the local extreme.

What's more, it is a popular topic to combine FOA with other intelligent algorithms. Si et al. [26] employed an improved FOA in combination with the least squares support vector machine (LSSVM) to solve the identification problem of shearer cutting pattern and constructed experiment. Wu et al. [27] proposed a normal cloud model based on FOA (CMFOA) to improve the convergence performance of original FOA, and numerical results show that CMFOA can obtain competitive solutions. Kanarachos et al. [28] introduced a contrast-based fruit fly optimization algorithm (c-mFOA) to solve efficient truss optimization. Niu et al. [29] proposed differential evolution FOA (DFOA) by modifying the expression of the smell concentration judgment value and introducing a differential vector to replace the stochastic search. Wang et al. [3] proposed LP-FOA (FOA with level probability policy) in which a level probability policy and a new mutation parameter are developed to balance the population diversity and stability. Yuan et al. [30] proposed chaotic-enhanced fruit fly optimization algorithm (CFOA), which employs chaotic sequence to enhance the global optimization capacity of original FOA. Zheng and Wang [31] presented a two-stage adaptive fruit fly optimization algorithm (TAFOA). At the first stage, a heuristic is proposed to generate an initial solution with high quality, and at the second stage, the initial solution is adopted as the initial swarm center for further evolution. Lei et al. [32] proposed fruit fly optimization clustering algorithm (FOCA) which combines FOA and gene expression profiles for identifying the protein complexes in dynamic protein-protein interaction networks. Meng and Pan [33] proposed an improved fruit fly optimization algorithm (IFFOA) to solve the multi-dimensional knapsack problems. In IFFOA, the parallel search is employed to balance exploitation and exploration and a modified harmony search algorithm (MHS) is applied to add cooperation among swarms.

In order to overcome the disadvantages of original FOA while retaining its merits, a new improved fruit fly optimization algorithm named IAFOA is proposed in this paper. In particular, the main contributions of IAFOA can be summarized as follows:

(1) Present a new concept named optimal search direction, and an adaptive selection mechanism for the search direction. The optimal search direction is calculated based on the experience of the previous iteration generations. In fact, the optimal search direction can be considered as a guide to judge which direction is more suitable to find a good solution. Therefore, if a direction is near to the optimal search direction, fruit flies in the following generation will fly towards it with a large probability. However, if a direction is far away from the optimal search direction, individuals in the next generation will fly towards it with a small probability. The nearness between a direction and the optimal search direction can be measured by the angle between the two directions. In order to quantitate the probability of individual flying towards different directions, a formula which is used to calculate the value of probability is given.
(2) Propose an adaptive adjustment mechanism for the iteration step value. The iteration step value is designed to be related to the iteration number and the changes of best smell concentrations during the iteration procedure. Most impor-

tantly, the iteration step value can change adaptively. At the beginning stage of the iteration procedure, a larger iteration step value is applied in order to obtain a rapid convergence speed; as the iteration procedure going on, a smaller iteration step value is used to ensure that the fruit flies search for the global optimum accurately; especially at the ending stage of the optimization process, if the best smell concentrations of several adjacent generations remain unchanged, the iteration step value will increase appropriately so that the individuals can search for the global optimum in a larger scope. From the introductions of the mechanisms which are used to adjust the search scope in current literatures, we know that most of these mechanisms suggest that the search radius should become small along with the increase of iteration number, and increase crudely once the swarm falls into a local extreme. Compared with the current methods, the adaptive adjustment mechanism for the iteration step value allows the search range to become bigger or smaller adaptively based on the iteration number and the changes of the best locations during the iteration procedure.

(3) Combine FOA with an adaptive crossover and mutation mechanism. Different from the current crossover and mutation operations for FOA, the adaptive crossover and mutation mechanism, which is inspired by arc tangent function, allows the crossover probability and mutation probability to change along with the fitness value of the individual. The bigger the fitness value is, the smaller the crossover probability and mutation probability are. It means that the individuals with lower fitness values are operated with greater crossover probability and mutation probability, and the individuals with higher fitness values are operated with lower crossover probability and mutation probability. Obviously, this mechanism can help to improve the diversity of the swarm. At the same time, these better individuals will be retained with a bigger probability so that the whole swarm will not degenerate.

(4) Introduce multi-sub-swarm mechanism to divide the whole swarm into two sub-swarms with equal population size. In the first sub-swarm, the individuals search for the optimal solution according to the rules of the improved FOA which includes the adaptive selection mechanism for the search direction and the adaptive adjustment mechanism for the iteration step value. In the second sub-swarm, the individuals will be operated by the adaptive crossover and mutation mechanism. At the end of each iteration, 1/2 individuals of each sub-swarm exchange with each other. Compared with the current combinations of FOA and multi-swarm mechanism, which just divide the whole swarm into several sub-swarms that move independently in the search scope, or assign different tasks to different the sub-swarms, the most important characteristics of the multi-sub-swarm mechanism proposed in this paper are the cooperation and exchange between two sub-swarms during every iteration procedure. Obviously, this mechanism can spread optimization information among individuals and quicken the convergence speed.

The experiment results based on a group of 29 benchmark functions show that IAFOA is more effective and efficient when solving the high-dimension functions compared with five variants of FOA and five well-known intelligent algorithms. Besides, IAFOA is used to solve three engineering optimization problems, which are oil compression spring design, welded beam design and speed reducer design. Experiment results show that IAFOA can achieve better solutions than the other algorithms for the three engineering optimization problems.

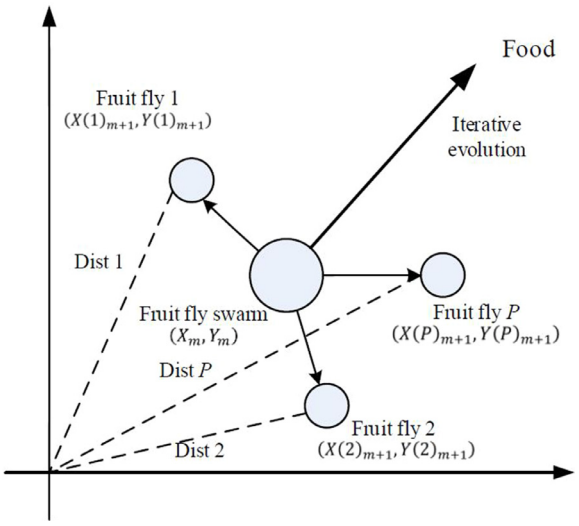

Fig. 1. Iteration procedure of original FOA.

The rest of this paper is organized as follows. Section 2 introduces the methods. The original FOA is explained and IAFOA is presented after introducing the four mechanisms in detail. The computational complexity analysis and convergence analysis of IAFOA are also shown in Section 2. In Section 3, experimental designs and numerical analyses based on a group of 29 benchmark functions are illustrated. IAFOA is used to solve engineering optimization problems in Section 4. Finally, Section 5 gives the concluding remarks.

2. Methods

2.1. Original FOA

Fruit fly is an insect that exists widely in temperate and tropical climate zones and is superior to other species in osphresis and vision. During hunting for food, fruit fly initially smells a particular odor by using its osphresis organs, sends and receives information from its neighbors and compares the smell concentration (or called fitness value) and the current best location. Fruit fly identifies the fitness value by taste, and flies towards the location with best fitness value. They use their sensitive vision to find food and fly towards that direction further [13].

Pan and other researchers have given the detailed descriptions of evolutionary steps of original FOA. According to current literatures, the iteration procedure of original FOA is shown in Fig. 1, and the pseudo code of original FOA is shown in Fig. 2 (when searching for maximum). Since all the individuals will gather at the best location in every iteration, the supreme advantage of FOA is the rapid convergence speed. At the same time, original FOA is very easy to fall into the local extreme when solving complex problems.

In original FOA, the smell concentration of individual $Smell(i)_m$ can be calculated by submitting the smell concentration judgment value $S(i)_m$ into the smell concentration judgment function (or called fitness function). Based on the iteration procedure of original FOA, it is obvious that the numerical value of $Dist_i$ is positive. Since $S(i)_m$ is the reciprocal of $Dist_i$, it is impossible to obtain a negative value for $S(i)_m$. It means that FOA cannot search for global optimum in negative domain. Actually, the issue has been reported by many researchers [18]. In this paper, similar to the methods proposed by Yuan et al. [18] and Pan et al. [4], both the distance $Dist_i$ and smell concentration judgment value $S(i)_m$ are removed, the fitness value of fruit fly is evaluated di-

Algorithm: Original fruit fly optimization algorithm
Parameters: Initial position of the swarm (X, Y), population size $\boldsymbol{P}$, maximum iteration number $\boldsymbol{M}$, initial iteration step value $\boldsymbol{R}$
Output: The best smell concentration and the coordinate of best location

// Initialization
Set (X, Y), $\boldsymbol{P}$, $\boldsymbol{M}$ and $\boldsymbol{R}$
m=0
$X(i)_0 = X + rand(\ \ ) \cdot \boldsymbol{R},\ Y(i)_0 = Y + rand(\ \ ) \cdot \boldsymbol{R},\ i = 1,2,\cdots,\boldsymbol{P}$
$Dist_i = \sqrt{X(i)_0{}^2 + Y(i)_0{}^2}$
$S(i)_0 = 1/Dist_i$
$Smell(i)_0 = F(S(i)_0)$
// Calculate initial smell concentration
$[bestSmell, bestIndex] = max(Smell)$
$Smellbest = bestSmell$
$X_0 = X(bestIndex), Y_0 = Y(betIndex)$
while m<$\boldsymbol{M}$
  //Osphresis searching process
  $X(i)_m = X(bestIndex) + rand(\ \ ) \cdot \boldsymbol{R},\ Y(i)_m = Y(bestIndex) + rand(\ \ ) \cdot \boldsymbol{R},\ i = 1,2,\cdots,\boldsymbol{P}$
  $Dist_i = \sqrt{X(i)_m{}^2 + Y(i)_m{}^2}$
  $S(i)_m = 1/Dist_i$
  $Smell(i)_m = F(S(i)_m)$
  $[bestSmell, bestIndex] = max(Smell)$
  //Vision searching process
  if $bestSmell > smellBest$
    $smellBest = bestSmell$
    $X_m = X(bestIndex), Y_m = Y(betIndex)$
  end if
  $m = m + 1$
end while
output results

Fig. 2. Pseudo code of original FOA.

rectly in the decision space of the function to avoid the limitations of the original definition of smell concentration judgement value [15]. In detail, assume that there is a multidimensional function optimization problem $F(X)$ with n variables $x^1, x^2, \cdots, x^n$, and the i-th fruit fly in the m-th iteration generation is denoted by $X(i)_m = (x(i)_m^1, x(i)_m^2, \cdots, x(i)_m^n)$, the fitness value of $X(i)_m$ can be calculated as follows:

$$Smell(i)_m = F(X(i)_m) \tag{1}$$

2.2. A new improved FOA (IAFOA)

In this section, four improvements, named adaptive selection mechanism for the search direction, adaptive adjustment mechanism for the iteration step value, adaptive crossover and mutation mechanism, multi-sub-swarm mechanism, are introduced. Then, a new improved FOA called IAFOA is proposed by combining the four mechanisms with FOA.

2.2.1. An adaptive selection mechanism for the search direction

In original FOA, the foraging behavior of fruit fly is random, it means that the search direction and search radius are uncertain. However, in the actual nature world, the animal swarm should be intelligent and can give a guide for the individual behavior. Under the guidance, the individuals should fly towards these directions which are more possible to find a better result. Thus, an adaptive selection mechanism for the search direction is proposed in this section. In detail, after calculating the best locations of the previous iteration generations, an optimal search direction is defined and then the fruit flies in the next iteration generation can forage food towards these directions which are much nearer to the optimal search direction with much larger probability.

The adaptive selection mechanism for the search direction can be described as follows:

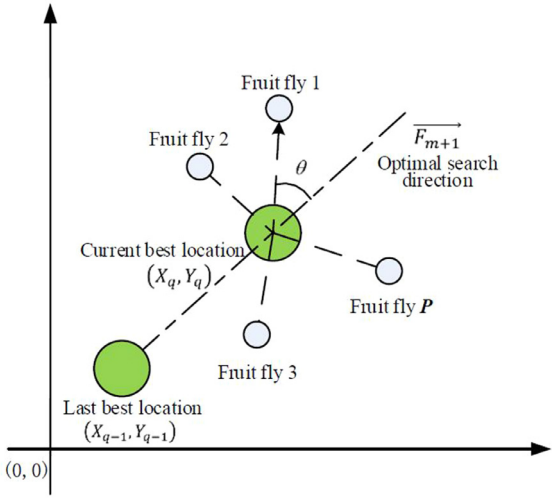

Fig. 3. The schematic diagram of the adaptive selection mechanism for the search direction.

Given the initial position of fruit fly swarm (X, Y), population size $\boldsymbol{P}$, and maximum iteration number $\boldsymbol{M}$. As shown in Fig. 3, assume that the coordinate of the best location of the m-th generation is (X_q, Y_q), and the coordinate of last best location during the optimization process is (X_{q-1}, Y_{q-1}). It is important to note that the last best location must be a location that is different with the current best location, so it may be not the best location of the (m-1)-th generation since sometimes the best locations of several adjacent generations are the same.

In fact, the experience of the previous iteration generations is very significant. Based on the changes of best locations of fruit

flies, a concept called optimal search direction, denoted by $\overrightarrow{F_{m+1}}$, is defined as follows:

$$\overrightarrow{F_{m+1}} = \left[\left(X_q - X_{q-1}\right),\ \left(Y_q - Y_{q-1}\right)\right] \tag{2}$$

As shown in formula (2), the optimal search direction is from the last best location to the current best location. Obviously, it is not sane to make all the individuals fly towards the optimal search direction in order to search for the global optimum in a wide range. However, it is a good choice to let the fruit flies in the following iteration generation fly towards these directions which are near to the optimal search direction with a big probability, and fly towards these directions which are far away from the optimal search direction with a small probability. As shown in Fig. 3, the nearness between a direction and the optimal search direction can be measured by the angle (denoted by θ) between the two directions. Obviously, the value of θ is from 0 to π, and distributes symmetrically.

In order to quantitate the probability of individual flying towards the direction with θ, a formula is given as follows:

$$g(\theta) = C - \frac{C}{\pi}\cdot\theta \tag{3}$$

in which C is a constant. Apparently, the smaller the value of θ is, the bigger the value of $g(\theta)$ is. Especially, when θ=0, $g(\theta)$=C, it means that fruit flies fly towards the optimal search direction with the maximum probability of C; when θ=π, $g(\theta)$=0, it means that fruit flies fly towards the opposite direction of the optimal search direction with the probability of 0. Obviously, we must make sure that all the fruit flies can fly towards a direction to forage food, it means that the definite integral of formula (3) with respect to θ from 0 to π is 1. Taking into account the symmetry of the distribution of θ, we have,

$$2\cdot\int_0^{\pi} g(\theta) = 2\cdot\int_0^{\pi}\left(C - \frac{C}{\pi}\cdot\theta\right) = 1 \tag{4}$$

It is easy to get,

$$C = \frac{1}{\pi} \tag{5}$$

Then, we give the final expression of $g(\theta)$,

$$g(\theta) = \frac{1}{\pi} - \frac{\theta}{\pi^2} \tag{6}$$

Actually, the aforementioned expression is based on the dimension of the optimization problem (denoted by n) is 2. By that analogy, it is also easy to understand in three-dimensional space if $n = 3$. When $n > 3$, the optimal search direction $\overrightarrow{F_{m+1}}$ can be calculated in an abstract space as follows:

$$\overrightarrow{F_{m+1}} = \left[\left(X_q^1 - X_{q-1}^1\right),\ \left(X_q^2 - X_{q-1}^2\right),\ \cdots,\ \left(X_q^n - X_{q-1}^n\right)\right] \tag{7}$$

in which X_q^n and X_{q-1}^n is the n-th variable while describing the best location of fruit flies.

The angle θ between a direction and the optimal search direction can be calculated as follows:

$$\theta = arc\cos\left[\frac{\left(X_q^1 - X_{q-1}^1\right)\cdot\left(X(p)_{m+1}^1 - X_q^1\right) + \left(X_q^2 - X_{q-1}^2\right)\cdot\left(X(p)_{m+1}^2 - X_q^2\right) + \cdots + \left(X_q^n - X_{q-1}^n\right)\cdot\left(X(p)_{m+1}^n - X_q^n\right)}{\sqrt{\left(X_q^1 - X_{q-1}^1\right)^2 + \left(X_q^2 - X_{q-1}^2\right)^2 + \cdots + \left(X_q^n - X_{q-1}^n\right)^2}\cdot\sqrt{\left(X(p)_{m+1}^1 - X_q^1\right)^2 + \left(X(p)_{m+1}^2 - X_q^2\right)^2 + \cdots + \left(X(p)_{m+1}^n - X_q^n\right)^2}}\right] \tag{8}$$

in which $(X(p)_{m+1}^1, X(p)_{m+1}^2, \cdots, X(p)_{m+1}^n)$ is a point on the direction.

Especially, when n=1, the only variable just can become bigger or smaller. Therefore, we stipulate that the adaptive selection mechanism for the search direction is no longer applicable in this case.

2.2.2. Adaptive adjustment mechanism for the iteration step value

In original FOA, the iteration step value $\boldsymbol{R}$ keeps unchanged during the whole optimization procedure. In fact, the iteration step value is the maximum radius of the search range in every iteration. Apparently, a larger iteration step value can make the search scope much larger and the convergence speed much rapider. However, when the best solution is close to the global extreme, a smaller iteration step value is more conducive to finding the global optimum in an exact range. Therefore, an adaptive adjustment mechanism for the iteration step value is proposed in this section. The iteration step value is not only related to the iteration number, but also to the changes of best smell concentrations during the iteration procedure. In detail, a larger iteration step value is applied to make sure that the swarm can converge to a good solution quickly at the beginning stage of the iteration procedure. As the iteration procedure going on, a smaller iteration step value is used to ensure that the fruit flies can search for the global optimum accurately. Especially at the ending stage of the optimization procedure, if the best smell concentrations of several adjacent generations are the same, the iteration step value will increase appropriately so that the individuals can search for the global optimum in a larger scope.

Given the initial position of fruit fly swarm (X, Y), the smell concentration of the initial position $Smell(i)_0$, and initial iteration step value $\boldsymbol{R}$. Assume that the iteration step value becomes $\boldsymbol{R}_m$ after the m-th iteration and the best smell concentration of the m-th generation is $Smell(i)_m$. Then, the iteration step value will be updated to $\boldsymbol{R}_{m+1}$ after the $(m+1)$-th iteration and the best smell concentration of the $(m+1)$-th generation is denoted by $Smell(i)_{m+1}$. $\boldsymbol{R}_{m+1}$ can be calculated as follows (when searching for maximum):

$$\boldsymbol{R}_{m+1} = \boldsymbol{R}_m\cdot\frac{Smell(i)_m}{Smell(i)_{m+1}}\cdot exp\left(\frac{Smell(i)_m}{Smell(i)_{m+1}}\cdot\frac{m+1}{m} - 1\right) \tag{9}$$

From formula (9), we can obtain the following conclusions:

(1) At the beginning stage of the iteration procedure, the best smell concentration will increase quickly, it means $\frac{Smell(i)_m}{Smell(i)_{m+1}} < 1$. Since $\frac{m+1}{m}$ is a little bigger than 1, $(\frac{Smell(i)_m}{Smell(i)_{m+1}}\cdot\frac{m+1}{m} - 1) < 0$. Thus, $\frac{Smell(i)_m}{Smell(i)_{m+1}}\cdot exp(\frac{Smell(i)_m}{Smell(i)_{m+1}}\cdot\frac{m+1}{m} - 1) < 1$. It means $\boldsymbol{R}_{m+1} < \boldsymbol{R}_m$, the iteration step value will decrease.

(2) Especially, during the first several iterations, since m is very small, $\frac{m+1}{m}$ is bigger than the ratio of $Smell(i)_{m+1}$ and $Smell(i)_m$. For example if m=2, $\frac{m+1}{m} = 1.5$. It means $\frac{Smell(i)_m}{Smell(i)_{m+1}}\cdot\frac{m+1}{m} > 1$. As a result, $\frac{Smell(i)_m}{Smell(i)_{m+1}}\cdot exp(\frac{Smell(i)_m}{Smell(i)_{m+1}}\cdot\frac{m+1}{m} - 1) > 1$. In this case, the iteration step value will increase.

(3) At the middle & later stage of the iteration procedure, the best smell concentrations of several adjacent generations change slightly or remain unchanged. It means $\frac{Smell(i)_m}{Smell(i)_{m+1}} \approx 1$. In this case, $exp(\frac{Smell(i)_m}{Smell(i)_{m+1}}\cdot\frac{m+1}{m} - 1)$ will play an important role to increase the iteration step value. Although $\frac{m+1}{m}$ is just a little bigger than 1, the value of $\boldsymbol{R}_m\cdot\frac{Smell(i)_m}{Smell(i)_{m+1}}\cdot exp(\frac{Smell(i)_m}{Smell(i)_{m+1}}\cdot\frac{m+1}{m} - 1)$ will increase observably after several iterations. Due to the increase of the iteration step value, fruit flies can search for the global optimum in a larger scope.

(4) If the value of $Smell(i)_m$ or $Smell(i)_{m+1}$ is zero, $\frac{Smell(i)_m}{Smell(i)_{m+1}} \cdot \frac{m+1}{m}$ will be zero or infinite. Obviously, it is not reasonable. Therefore, we must check the values of $Smell(i)_m$ and $Smell(i)_{m+1}$ before updating $\boldsymbol{R}_{m+1}$. If they are nonzero, $\boldsymbol{R}_{m+1}$ can be updated according to formula (9). Otherwise, let $\boldsymbol{R}_{m+1} = \boldsymbol{R}_m$, it means that the iteration step value keeps the same.

When searching for minimum, $\boldsymbol{R}_{m+1}$ will be defined as follows:

$$\boldsymbol{R}_{m+1} = \boldsymbol{R}_m \cdot \frac{Smell(i)_{m+1}}{Smell(i)_m} \cdot exp\left(\frac{Smell(i)_{m+1}}{Smell(i)_m} \cdot \frac{m+1}{m} - 1\right) \quad (10)$$

Experiments will be operated to test the effects of the adaptive adjustment mechanism for the iteration step value based on a group of benchmark functions in Section 3.

2.2.3. Adaptive crossover and mutation mechanism

In original FOA, all the fruit flies will gather at the best location in every iteration. Obviously, it is not good for promoting the population diversity which is beneficial for the exploitation ability of intelligent algorithm [34]. As we know, the crossover and mutation operators play vital roles in generating individual variants. In order to overcome the drawbacks of FOA, an adaptive crossover and mutation mechanism is introduced in this section.

Compared with the current crossover and mutation operations, the adaptive crossover and mutation mechanism, which is inspired by the arc tangent function $y = arctan(x)$, makes the crossover probability P_c and mutation probability P_m change adaptively. The individuals with lower fitness values are operated with a greater crossover and mutation probability, and the individuals with higher fitness values are operated with a lower crossover and mutation probability. Actually, the rules to calculate P_c and P_m for different individuals with different fitness values can be defined as follows:

$$P_c = \begin{cases} \frac{P_{c\ min}+P_{c\ max}}{2} + \frac{P_{c\ min}-P_{c\ max}}{\pi} \cdot 2 \cdot arctan\left(\frac{2 \cdot f' - f_{max} - f_{avg}}{f_{max} - f_{avg}}\right), & f' \geq f_{avg} \\ P_{c\ max}, \quad f' < f_{avg} \end{cases} \quad (11)$$

$$P_m = \begin{cases} \frac{P_{m\ min}+P_{m\ max}}{2} + \frac{P_{m\ min}-P_{m\ max}}{\pi} \cdot 2 \cdot arctan\left(\frac{2 \cdot f - f_{max} - f_{avg}}{f_{max} - f_{avg}}\right), & f \geq f_{avg} \\ P_{c\ max}, \quad f < f_{avg} \end{cases} \quad (12)$$

In formulas (11) and (12), $P_{c\,max}$ and $P_{c\,min}$ are the top and bottom limits of crossover probability; $P_{m\,max}$ and $P_{m\,min}$ are the top and bottom limits of mutation probability; f_{avg} is the average fitness value of all the individuals; f_{max} is the largest fitness value of all the individuals; f' is the bigger fitness value of the two individuals that take part in the crossover operation; and f is the fitness value of the mutated individual.

Based on formulas (11) and (12), the curve of crossover probability and mutation probability is plotted and shown in Fig. 4. As shown in Fig. 4, the individuals with lower fitness values will be operated with greater P_c and P_m. Especially, when the fitness value is lower than f_{avg}, the individuals will cross or mutate with the greatest probability $P_{c\,max}$ or $P_{m\,max}$. For the individuals with fitness values higher than f_{avg}, crossover probability P_c and mutation probability P_m will change based on arc tangent function $y = arctan(x)$. The greater the fitness values of individuals are, the smaller the P_c and P_m are. If the fitness value of individual is closed to f_{max}, the P_c and P_m are almost equal to $P_{c\,min}$ and $P_{m\,min}$. In this paper, $P_{c\,min}$ and $P_{m\,min}$ are set to zero in order to make sure that the best individual will not be destroyed. Obviously, the adaptive crossover and mutation mechanism can help to improve the diversity of the fruit fly swarm. At the same time, the best individual will be retained in every iteration so that the whole swarm will not degenerate.

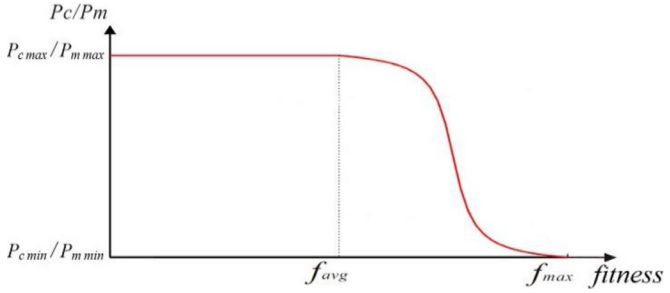

Fig. 4. Curve of P_c and P_m according to the adaptive crossover and mutation mechanism.

2.2.4. Multi-sub-swarm mechanism

Multi-sub-swarm and cooperation between sub-swarms can spread optimization information among individuals and quicken the convergence speed. In this section, a multi-sub-swarm mechanism is presented. The whole fruit fly swarm is divided into two sub-swarms with equal population size. Fruit flies in the first sub-swarm search for the optimal solution according to the rules of FOA and two mechanisms which are the adaptive selection mechanism for the search direction and the adaptive adjustment mechanism for the iteration step value. The individuals included in the second sub-swarm will cross and mutate according to the adaptive crossover and mutation mechanism. It means that the two sub-swarms search for the optimal solution in different ways. At the end of each iteration, a certain number of individuals are selected from the two sub-swarms and exchange with each other. The detailed procedure from m-th generation to $(m+1)$-th generation can be found in Fig. 5.

As shown in Fig. 5, the population size of initial swarm is $\boldsymbol{P}$, and the initial swarm is divided into two sub-swarms. At the beginning of the m-th iteration, $\boldsymbol{P/2}$ number of individuals are included in the first sub-swarm. For this sub-swarm, evaluate the smell concentration of each fruit fly, and find out the individual with the best smell concentration. Then, let other individuals fly towards the best location. Finally, update the iteration step value according to the adaptive adjustment mechanism for the iteration step value, and then fruit flies search for food based on the adaptive selection mechanism for the search direction. At the moment, $\boldsymbol{P/2}$ number of individuals exist in the first sub-swarm.

In the second sub-swarm, which includes $\boldsymbol{P/2}$ number of individuals, the individuals are operated by crossover and mutation operations. Then, $\boldsymbol{P/2}$ number of offspring individuals are generated. After evaluating the fitness value of every parent and offspring individual, $\boldsymbol{P/2}$ number of individuals with lower fitness values are removed and $\boldsymbol{P/2}$ number of individuals with higher fitness values are retained. At the moment, also $\boldsymbol{P/2}$ number of individuals exist in the second sub-swarm. The next step, which is of vital importance, is to select $\boldsymbol{P/4}$ number of individuals from the first sub-swarm and $\boldsymbol{P/4}$ number of individuals from the second sub-swarm randomly, and then exchange them.

After these operations, the $(m+1)$-th generation, which also includes two sub-swarms, is formed and ready for the next iteration. The circulatory iteration procedure will ends until meeting the stopping criterions. However, before the first iteration beginning, the two sub-swarms are created from the initial swarm randomly, and then they evolve and cooperate to produce good solution. For convenience, we can call the first sub-swarm FOA sub-swarm, and call the second sub-swarm GA sub-swarm.

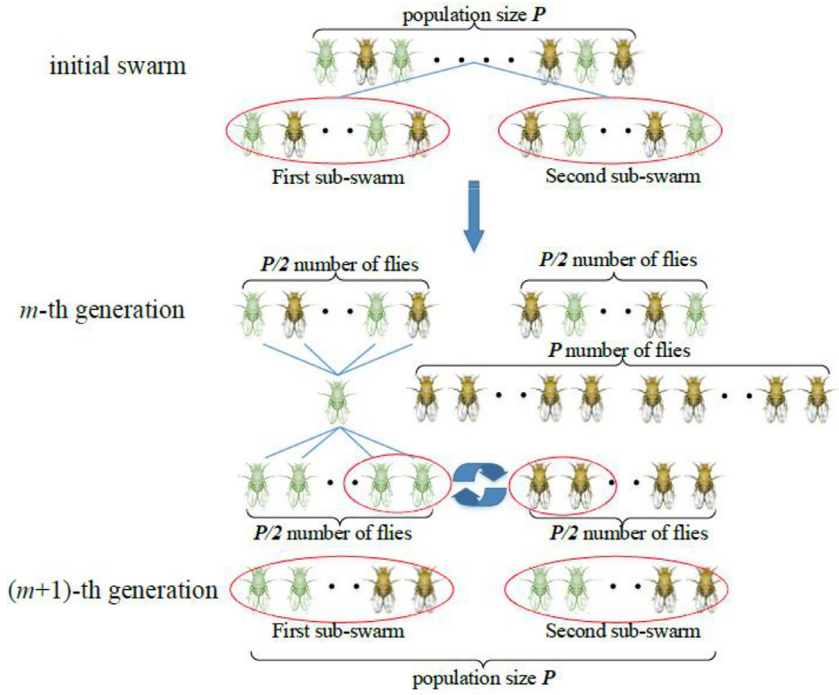

Fig. 5. The schematic diagram of multi-sub-swarm mechanism.

2.2.5. Procedure of IAFOA

Combining FOA with the four improvements, which are the adaptive selection mechanism for the search direction, the adaptive adjustment mechanism for the iteration step value, the adaptive crossover and mutation mechanism, and the multi-sub-swarm mechanism, a new improved FOA (IAFOA) is presented. The procedure of IAFOA can be described as follows:

Step 1: Initialization process. Set the initial position of swarm (X, Y), population size $\boldsymbol{P}$, maximum iteration number $\boldsymbol{M}$, and initial iteration step value $\boldsymbol{R}$.

Step 2: Individuals search for food randomly, and the location of the i-th individual is $(X(i)_1, Y(i)_1)$. Then, divide the initial swarm into two sub-swarms, and $\boldsymbol{P/2}$ number of individuals are included in each sub-swarm.

Step 3: (1) For the first sub-swarm, calculate the smell concentration of the individuals $Smell(i)_m$, in which m is the iteration number,

$$Smell(i)_m = F(X(i)_m) \tag{13}$$

Find the individual with the maximum smell concentration (when searching for maximum),

$$[bestSmell\ \ bestIndex] = max(Smell) \tag{14}$$

Record the best smell concentration value and the corresponding location (X_m, Y_m). Then, other individuals fly towards (X_m, Y_m),

$$\begin{cases} smellBest = bestSmell \\ X_m = X(bestIndex) \\ Y_m = Y(bestIndex) \end{cases} \tag{15}$$

Update the iteration step value (when searching for maximum),

$$\boldsymbol{R}_{m+1} = \boldsymbol{R}_m \cdot \frac{Smell(i)_m}{Smell(i)_{m+1}} \cdot exp\left(\frac{Smell(i)_m}{Smell(i)_{m+1}} \cdot \frac{m+1}{m} - 1 \right) \tag{16}$$

Calculate the probability of individual flying towards the direction with θ,

$$g(\theta) = \frac{1}{\pi} - \frac{\theta}{\pi^2} \tag{17}$$

Then, $\boldsymbol{P/2}$ number of individuals search for food from the location (X_m, Y_m) based on the adaptive selection mechanism for the search direction.

(2) For the second sub-swarm, individuals will be implemented crossover and mutation operations according to the adaptive crossover and mutation mechanism.

After mixing the $\boldsymbol{P/2}$ number of parent individuals and $\boldsymbol{P/2}$ number of offspring individuals, $\boldsymbol{P/2}$ number of individuals with lower fitness values will be removed, and $\boldsymbol{P/2}$ number of individuals with higher fitness values will be retained.

Step 4: Exchange individuals selected from the two sub-swarms.

Select $\boldsymbol{P/4}$ number of individuals from the first sub-swarm randomly, and exchange with $\boldsymbol{P/4}$ number of individuals which are selected from the second sub-swarm randomly.

Then, the $(m+1)$-th generation, which includes two sub-swarms, is formed and ready for the next iteration.

Step 5: Repeat step 3 to step 4 until the best fitness value meets the preset precision or the iteration number m is not less than the maximum iteration number $\boldsymbol{M}$.

Step 6: Output the results.

Based on the procedure, optimization flowchart of IAFOA is shown in Fig. 6.

In summary, IAFOA inherits the momentous advantage of original FOA which is to generate a satisfactory solution at a fast convergence speed. What's more, due to the adaptive selection mechanism for the search direction, the completely randomized searching procedure is replaced by an experience-based iteration procedure in IAFOA. Besides, the adaptive adjustment mechanism for the iteration step value can make the fruit flies search for solutions within a reasonable and changing search range. A bigger iteration step value is helpful to search for global extreme in a bigger scope, and a smaller search radius allows the individuals to forage food accurately. In addition, the adaptive crossover and mutation mechanism brings a method to improve the diversity of the swarm without destroying the best individuals. Finally, the multi-sub-swarm mechanism gives an opportunity to work cooperatively

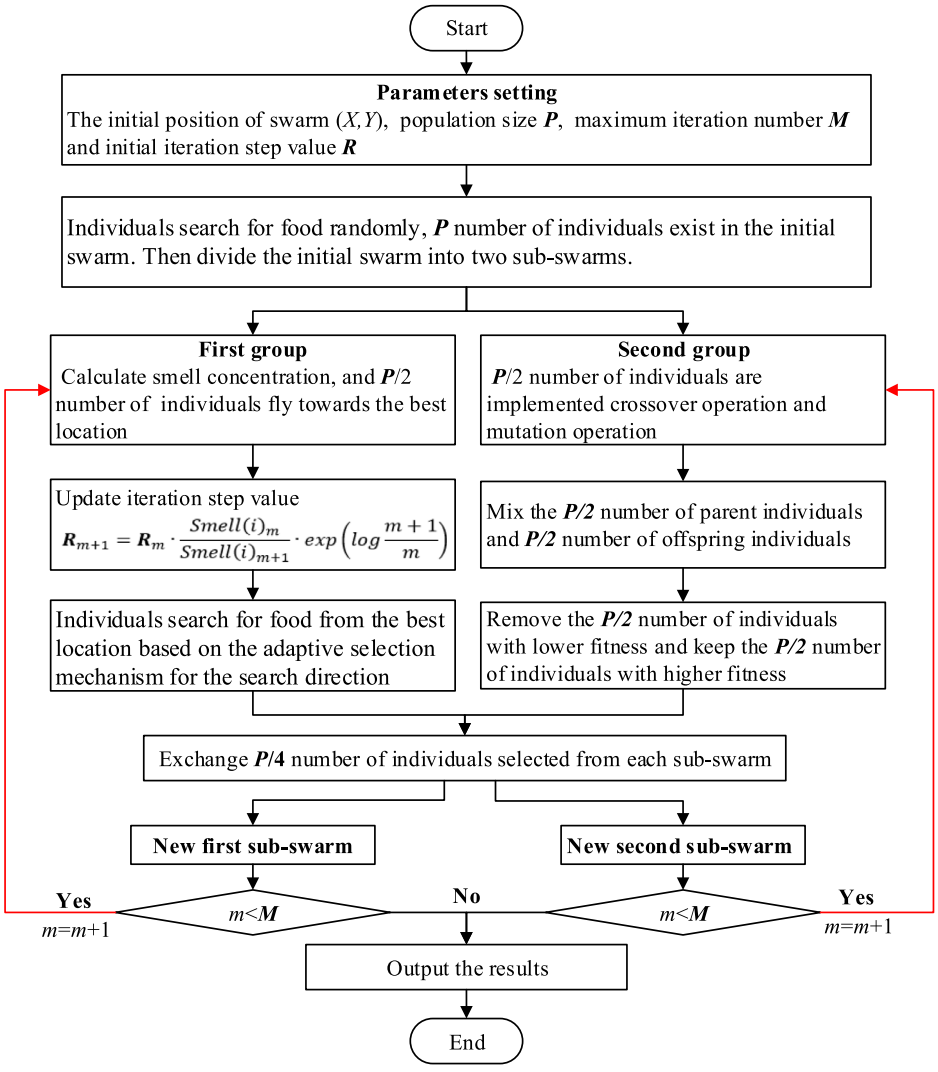

Fig. 6. Optimization flowchart of IAFOA.

for two different optimization approaches. Apparently, it is good for taking the chief and discarding the short of the two searching processes, one is based on improved FOA, and the other is based on the crossover and mutation operations.

On the whole, the adaptive selection mechanism for the search direction can accelerate the convergence speed, the adaptive adjustment mechanism for the iteration step value can make the iteration procedure be more intelligent and bring a chance to search for optimal extreme in a bigger scope especially at the later stage of the iteration procedure. The major function of the adaptive crossover and mutation mechanism is to generate different individual variants and to improve the diversity of the swarm. And multi-sub-swarm mechanism can make IAFOA obtain the advantages of two different iteration ways. Finally, the procedure of IAFOA presents a wonderful way to combine the several strategies. All the improvements can take effect and give contributions to the proposed IAFOA.

2.2.6. Computational complexity analysis

In order to analyze the efficiency of algorithm, computational complexity analysis is necessary and used to estimate the computing time [32]. In fact, the computational complexity of an intelligent optimization algorithm mainly includes two parts: algorithm execution and fitness evaluation. Since the computational complexity of algorithm execution is much larger than that of fitness evaluation, we just discuss the computational complexity of algorithm execution of IAFOA in this section [27]. Assume that the population size is ***P***, and the maximum iteration number is ***M***. According to the procedure of IAFOA, its computational complexity can be analyzed as follows:

(1) The computational complexity of initializing the locations of fruit flies is O(P);
(2) The computational complexity of dividing the whole swarm into two sub-swarms is O(P);
(3) For the first sub-swarm including ***P/2*** number of individuals, the computational complexity of updating individual positions is O($\frac{P}{2} \cdot M$), the computational complexity of calculating the iteration step values is O(M), the computational complexity of calculating the optimal search directions is O(M), and the computational complexity of selecting fly directions for individuals is O($P \cdot M$);
(4) For the second sub-swarm including ***P/2*** number of individuals, the computational complexity of calculating crossover probability and mutation probability is O($P \cdot M$), the compu-

tational complexity of carrying out crossover and mutation operations is $O(P \cdot M)$.

Usually, it is reasonable to take the highest power item as the computational complexity of algorithm. Therefore, the analyses above show that the computational complexity of IAFOA is $O(\frac{7}{2} \cdot P \cdot M)$. Compared with the computational complexity of several variants of FOA as mentioned in literature [27], such as the computational complexity of CMFOA is $O(2 \cdot P \cdot M)$, the computational complexity of MFOA is $O(5 \cdot P \cdot M)$, the computational complexity of IFFO is $O(P \cdot M)$, we can say that the computational complexity of IAFOA is acceptable. The comparison of the computing time spent by several variants of FOA when solving a group of benchmark functions, will be illustrated in the Section 3.

2.2.7. Convergence analysis of IAFOA

Due to the multi-sub-swarm mechanism, IAFOA can be considered as a combination of an improved FOA and an improved GA. According to the adaptive crossover and mutation mechanism, the individual with highest fitness value in the second sub-swarm will be operated with crossover probability and mutation probability of 0. It means that the best elite in each generation will be retained. Actually, based on the Markov chain analysis, many researchers have proved that the genetic algorithm with elitism strategy can converge to the optimal extreme theoretically [35]. However, the convergence analysis of FOA is rare in current research and literatures. Therefore, in this section, we mainly focus on two issues:

(1) Convergence analysis of IAFOA;
(2) Description why the proposed mechanisms in this paper can make contribution to the convergence of IAFOA.

Assume that a multidimensional function optimization problem $F(X) = f(x^1, x^2, \cdots, x^n)$ with n variables $x^1, x^2, \cdots, x^n$, has the global extreme $F(X^*) = f(x_*^1, x_*^2, \cdots, x_*^n)$ when $X^* = (x_*^1, x_*^2, \cdots, x_*^n)$ existing in the search space. We call $X^* = (x_*^1, x_*^2, \cdots, x_*^n)$ the global solution. For convenience, we restrict the application domain to minimization problems.

Lemma 1. *The search scope of IAFOA always can cover the global solution after some iterations.*

Proof. Let $X_m = (x_m^1, x_m^2, \cdots, x_m^n)$ be the best solution in the m-th generation, and R_m is the search radius. If $\exists i (i \in [1, 2, \cdots, n])$, let

$$\left| x_m^i - x_*^i \right| = Con > R_m \tag{18}$$

in which Con is a finite constant. We can say that the global solution is out of the search scope of the fruit flies.

According to the rules of FOA, individuals forage food in the search scope. Since the search radius is a fixed value in original FOA, it cannot become big or small. Actually, if the current best solution $X_m = (x_m^1, x_m^2, \cdots, x_m^n)$ is a local solution in the search scope with radius of R_m, due to $|x_m^i - x_*^i| > R_m$, it is impossible to let the i-th variables become x_*^i during the next iteration. Considering the fruit fly swarm will gather at the best location in every iteration, it is different to jump out the local extreme for original FOA.

However, in IAFOA, due to the adaptive adjustment mechanism for the iteration step value, the search radius will become bigger when the algorithm trapping a local extreme. Thus, $\exists m'$, let $\forall i \in (1, 2, \cdots, n)$

$$R_{m+m'} \geq \left| x_m^i - x_*^i \right| \tag{19}$$

In this case, the global solution is covered by the search scope of the individuals.

Theorem 2. *IAFOA can converge to the global extreme with a probability of 100% theoretically.*

Proof. From Lemma 1, we can know that there exist a finite number m can let the search scope of IAFOA cover the global solution. Due to the randomness of searching process, $\exists coef_i \in [-1, 1]$, let

$$\begin{cases} x_{m+1}^1 = x_m^1 + coef_1 R_m = x_*^1 \\ x_{m+1}^2 = x_m^2 + coef_2 R_m = x_*^2 \\ \qquad\qquad \vdots \\ x_{m+1}^n = x_m^n + coef_n R_m = x_*^n \end{cases} \tag{20}$$

It means that, $\forall i \in (1, 2, \cdots, n)$

$$P\left\{ \left| x_{m+1}^i - x_*^i \right| = x_m^i - x_*^i + coef_i \cdot R_m = 0 \right\} > 0 \tag{21}$$

in which $P\{\cdot\}$ is the probability of the random event $\{\cdot\}$ happens.

Besides, due to another two mechanisms, the adaptive crossover and mutation mechanism and the multi-sub-swarm mechanism, some individuals will be created by crossover and mutation operations. As we know, the crossover and mutation operations are stochastic methods. Assume that the best individuals created by crossover and mutation operations is $X_{m-s} = (x_{m-s}^1, x_{m-s}^2, \cdots, x_{m-s}^n)$, we have,

$$P\left\{ f\left(x_{m-s}^1, x_{m-s}^2, \cdots, x_{m-s}^n\right) \leq f\left(x_m^1, x_m^2, \cdots, x_m^n\right) \right\} > 0 \tag{22}$$

The proof of formula (22) can refer the convergence analysis of the genetic algorithm with elitism strategy.

Assume that $X_{m+1} = (x_{m+1}^1, x_{m+1}^2, \cdots, x_{m+1}^n)$ is the best individual in the $(m+1)$-th generation, according the rules of IAFOA we have $f(x_{m+1}^1, x_{m+1}^2, \cdots, x_{m+1}^n) \leq f(x_m^1, x_m^2, \cdots, x_m^n)$. Based on formulas (21) and (22), we can get, $\forall i \in (1, 2, \cdots, n)$

$$\lim_{m \to \infty} \left| x_m^i - x_*^i \right| = 0 \tag{23}$$

It means that IAFOA can converge to the global extreme with a probability of 100%.

Actually, the adaptive iteration step value and the adaptive selection mechanism for the search direction can promote the optimization process of IAFOA. Based on the Lemma 1, we can assume that in the m-th iteration, $\forall i \in (1, 2, \cdots, n)$, let $R_m \geq |x_m^i - x_*^i|$, and then the value of $\frac{P}{\pi R_m^2}$ ($\boldsymbol{P}$ is the population size of fruit fly swarm) is called point-face rate of the m-th iteration. Obviously, a big value of $\frac{P}{\pi R_m^2}$ means that the individuals can find the global best location with a big probability, and also can let the individuals search for the global extreme accurately. As mentioned in Section 2.2.2, at the beginning stage of optimization procedure of IAFOA, the iteration step value is becoming smaller along with the increase of the iteration number. A smaller iteration step value can make the point-face rate $\frac{P}{\pi R_m^2}$ much bigger. Therefore, the adaptive adjustment mechanism for the iteration step value can let the IAFOA converge to a better solution with a big probability.

In original FOA, the search direction of fruit fly is selected randomly. It means that every direction can be selected with the same probability. Suppose the probability of individual flying towards the direction with θ (θ is the angle between the direction and the optimal search direction) is $g(\theta)_{FOA}$ in original FOA, and the probability of individual flying towards the direction with θ is $g(\theta)$ in IAFOA, then we have,

$$\begin{cases} g(\theta)_{FOA} < g(\theta) & \theta \in [0, \pi/2) \\ g(\theta)_{FOA} \geq g(\theta) & \theta \in [\pi/2, \pi] \end{cases} \tag{24}$$

Thus,

$$\int_0^{\pi/2} g(\theta) > \int_0^{\pi/2} g(\theta)_{FOA} = \int_{\pi/2}^{\pi} g(\theta)_{FOA} > \int_{\pi/2}^{\pi} g(\theta) \tag{25}$$

Due to the adaptive selection mechanism for the search direction, more individuals will fly towards these directions with small θ. Obviously, it is beneficial to obtain a good solution at a fast speed.

Table 1
List of benchmark functions.

No.	Functions	x_i	x_*	$f(x_*)$
F1	$f(x)=\sum_{i=2}^{n} ix_i^2$	(−5.12,5.12)	0	0
F2	$f(x)=\sum_{i=2}^{n} i(2x_i^2-x_{i-1})^2+(x_1-1)^2$	(−10,10)	$2^{(2-2^i)/2^i}$	0
F3	$f(x)=-exp(-0.5\sum_{i=1}^{n} x_i^2)$	(−1,1)	0	−1
F4	$f(x)=\sum_{i=1}^{n} (10^6)^{\frac{i-1}{n-1}} x_i^2$	(−100,100)	0	0
F5	$f(x)=\sum_{i=1}^{n} ix_i^4+rand(0,1)$	(−1.28,1.28)	0	0
F6	$f(x)=\sum_{i=1}^{n-1}[100(x_{i+1}-x_i^2)^2+(x_i-1)^2]$	(−30,30)	0	1
F7	$f(x)=\sum_{i=1}^{n}(\sum_{j=1}^{i} x_j)^2$	(−100,100)	0	0
F8	$f(x)=max(\lvert x_i\rvert)$	(−100,100)	0	0
F9	$f(x)=\sum_{i=1}^{n}\lvert x_i\rvert+\prod_{i=1}^{n}\lvert x_i\rvert$	(−10,10)	0	0
F10	$f(x)=\sum_{i=1}^{n} x_i^2$	(−100,100)	0	0
F11	$f(x)=\sum_{i=1}^{n} x_i+0.5^2$	(−100,100)	0	0
F12	$f(x)=\sum_{i=1}^{n}\lvert x_i\rvert^{i+1}$	(−1,1)	0	0
F13	$f(x)=\sum_{i=1}^{n} ix_i^2$	(−10,10)	0	0
F14	$f(x)=\sum_{i=1}^{n} x_i^2-450$	(−100,100)	0	−450
F15	$f(x)=\sum_{i=1}^{n}(\sum_{j=1}^{i} x_j)^2-450$	(−100,100)	0	−450
F16	$f(x)=-20exp(-0.2\sqrt{\frac{1}{n}\sum_{i=1}^{n} x_i^2})-exp(\frac{1}{n}\sum_{i=1}^{n}\cos(2\pi x_i))+20+e$	(−32,32)	0	0
F17	$f(x)=\sum_{i=1}^{n}\lvert x_i\sin(x_i)+0.1x_i\rvert$	(−10,10)	0	0
F18	$f(x)=f_1(x_1,x_2)+f_1(x_2,x_3)+\cdots+f_1(x_n,x_1),\ f_1(x,y)=(x^2+y^2)^{0.25}[\sin^2(50((x^2+y^2)^{0.1}))+1]$	(−100,100)	0	0
F19	$f(x)=f_1(x_1,x_2)+f_1(x_2,x_3)+\cdots+f_1(x_n,x_1)$, $f_1(x,y)=0.5+(\sin^2(\sqrt{x^2+y^2})-0.5)/(1+0.001(x^2+y^2))^2$, $f(x)=\frac{\pi}{n}\{10\sin^2(\pi y_1)+\sum_{i=1}^{n-1}(y_1-1)^2[1+10\sin^2(\pi y_{1+1})]+(y_n-1)^2\}+\sum_{i=1}^{n}\mu(x_i,10,100,4)$	(−100,100)	0	0
F20	$y_i=1+0.25(x_i+1),\ \mu(x_i,a,k,m)=\begin{cases} k(x_i-a)^m, x_i>a \\ 0, -a<x_i<a \\ k(-x_i-a)^m, x_i<-a \end{cases}$	(−50,50)	−1	0
F21	$f(x)=\frac{1}{4000}\sum_{i=1}^{n} x_i^2-\prod_{i=1}^{n}\cos(\frac{x_i}{\sqrt{i}})+1$	(−600,600)	0	0
F22	$f(x)=-\sum_{i=1}^{n-1}(exp(-\frac{x_i^2+x_{i+1}^2+0.5x_ix_{i+1}}{8})\cos(4\sqrt{x_i^2+x_{i+1}^2+0.5x_ix_{i+1}}))$	(−5,5)	0	1−n
F23	$f(x)=\sum_{i=1}^{n}(x_i-1)^2-\sum_{i=2}^{n} x_ix_{i-1}$	$(-n^2,n^2)$	$i(n+1-i)$	$\frac{n(n+4)(1-n)}{6}$
F24	$f(x)=\sum_{i=1}^{n-1}(0.5+\frac{\sin^2(\sqrt{100x_i^2+x_{i+1}^2})-0.5}{1+0.001(x_i^2-2x_ix_{i-1}+x_{i+1}^2)^2})^2$	(−100,100)	0	0
F25	$f(x)=\sum_{i=1}^{n}(x_i^2-10\cos(2\pi x_i)+10)$	(−5.12,5.12)	0	0
F26	$f(y)=\sum_{i=1}^{n}(y_i^2-10\cos(2\pi y_i)+10),\ y_i=\begin{cases} x_i, \lvert x_i\rvert<0.5 \\ round(2x_i)/2, \lvert x_i\rvert\geq 0.5 \end{cases}$	(−5.12,5.12)	0	0
F27	$f(x)=1-\cos(2\pi(\sqrt{\sum_{i=1}^{n} x_i^2})+0.1\sqrt{\sum_{i=1}^{n} x_i^2})$	(−100,100)	0	0
F28	$f(x)=\sum_{i=1}^{n}\{\sum_{k=0}^{k_{max}}[a^k\cos(2\pi b^k(x_i+0.5))]\}-n\sum_{n=0}^{k_{max}}[a^k\cos(2\pi b^k\cdot 0.5)],\ a=0.5,\ b=0.3,\ k_{max}=30$	(−0.5,0.5)	0	0
F29	$f(x)=\sum_{k=1}^{n}\sum_{j=1}^{n}(y_{jk}^2/4000-\cos(y_{jk})+1),\ y_{jk}=100(x_k-x_j^2)^2+(1-x_j^2)^2$	(−100,100)	1	0

Overall, we can say that IAFOA can converge to the global extreme theoretically, and the four proposed mechanisms in this paper are helpful to address the multidimensional function optimization problem.

3. Experiments and numerical analysis

For the purpose of verifying the performance of IAFOA, a total of 29 benchmark functions including 15 unimodal functions and 14 multimodal functions, are considered in the experimental section. These benchmark functions shown in Table 1 are widely adopted in benchmarking global optimization algorithms. In fact, different functions can be used to test different abilities of algorithms, such as the unimodal functions are suitable for testing the exploitation ability of algorithms because they only have one global extreme, and multi-modal functions are suitable for testing the exploration ability of algorithms because they have a global extreme as well as many local optima [36]. In this section, all the algorithms are coded in Matlab 2012a on the computer with an Intel® Core(TM) i7-6500U CPU@2.50 GHz, 8GB RAM, Windows 7.

As we know, there are many coding methods and many variants of crossover and mutation operators. Therefore, several instructions are listed as follows:

(1) For IAFOA, the real number coding method is adopted. When optimizing the high-dimensional function, fruit fly is coded

by a numeric string which includes n (n is the dimension of the function) numbers, and the n numbers represent the n variables of the function;

(2) As to crossover operator, the partial-mapped crossover is used, and the quantity of selected points is set to $2n/10$. As to mutation operator, Gaussian mutation and Cauchy mutation are two most popular methods. Compared with Gaussian mutation, Cauchy mutation has a stronger global exploration capability, which can effectively inhibit the loss of population diversity [37]. Therefore, standard Cauchy mutation is adopted in IAFOA.

3.1. Experiments for testing the effects of adaptive adjustment mechanism for the iteration step value

In order to test the effects of adaptive adjustment mechanism for the iteration step value, a set of special experiments is carried out based on the 29 benchmark functions. Since IAFOA includes four extra mechanisms compared with original FOA, it is difficult to distinguish the effects of each mechanism by comparing the results obtained by IAFOA and FOA. Hence, a transitional algorithm named AFOA, which just combines FOA with the adaptive adjustment mechanism for the iteration step value, is used in this set of experiments.

To be fair, the common parameters of AFOA and FOA are set to the same values: population size $\boldsymbol{P}=40$, maximum iteration number $\boldsymbol{M}=1000$ and the stopping criterion is iteration number $m=\boldsymbol{M}$. The initial iteration step value $\boldsymbol{R}$ is set according to the following rules: if the limiting value of x_i shown in Table 1 is less than 10 for a function, the iteration step value $\boldsymbol{R}$ is set to the limiting value of x_i; if the limiting value of x_i shown in Table 1 is not less than 10 for a function, the iteration step value $\boldsymbol{R}$ is set to 10.

Then, AFOA and FOA are used to solve the 29 benchmark functions on the computer mentioned above. Owing to their stochastic nature, evolutionary algorithms may arrive at solutions those are better or worse than solutions they have previously reached. For this reason, it is beneficial to use statistical tools to compare the problem-solving success of an algorithm with that of another [18]. Therefore, each problem is run with 50 independent replications. The results obtained by AFOA and FOA for the 29 functions with dimensions equal to 30, are shown in Table 2 which includes the best value (Best), the worst value (Worst), the mean value (Mean) and the standard deviation (Std.) in the 50 independent replications.

As shown in Table 2, the results obtained by AFOA are much better than those obtained by FOA. Firstly, AFOA is more effective in terms of the "Best" and "Worst" results. In detail, although AFOA and FOA both can produce the best solutions which are the theoretical optima for F03, F14 and F22, AFOA generates 22 significantly better "Best" (except F07, F11, F19, F25) for the remaining 26 functions compared with FOA. What's more, AFOA generates 25 significantly better "Worst" (except F19, F21, F22, F25) for the total of 29 functions than FOA. Secondly, AFOA is highly robust in terms of the "Mean" and "Std." results. In detail, AFOA obtains 26 significantly better "Mean" (except F19, F21, F25) and 26 significantly better "Std." (except F19, F22, F25) for the total of 29 functions compared with FOA. The data shown in Table 2 can prove that AFOA is able to provide more promising performance than FOA. Obviously, this is mainly contributed by the adaptive adjustment mechanism for the iteration step value. Therefore, we can conclude that the adaptive adjustment mechanism for the iteration step value can improve the performance of FOA effectively.

Besides, for the purpose of monitoring the changes of the iteration step values, several changing curves of $\boldsymbol{R}_m(1 \le m \le M)$ are shown in Fig. 7 when solving the benchmark functions. In order to save space, just some representative curves are illustrated. As shown in Fig. 7, the iteration step value is changing along with the optimization procedure. In the beginning, the iteration step value even increases so that fruit flies can search for a good solution in a large scope and at a fast speed. Then, the iteration step value decreases rapidly since the best fitness values of individuals are becoming better quickly. At the middle & later stage of the iteration procedure, the iteration step value will increase and decrease alternately. These curves can prove that the adaptive iteration step value is more suitable for searching for global extreme efficiently.

3.2. Experiments for testing the effects of the GA sub-swarm

In order to uncover the real benefit introduced by the GA sub-swarm, another set of experiments is carried out. As mentioned above, because IAFOA includes several extra improvements compared with original FOA, another transitional algorithm called GMFOA, which combines FOA with the adaptive crossover and mutation mechanism and the multi-sub-swarm mechanism is adopted in this section. Then, GMFOA and FOA are used to solve the 29 benchmark functions to verify the advantages of GAFOA. For a fair comparison, the common parameters of GMFOA and FOA are set to the same values. In detail, population size $\boldsymbol{P}=40$, maximum iteration number $\boldsymbol{M}=1000$, the stopping criterion is iteration number $m=\boldsymbol{M}$, and the initial iteration step value $\boldsymbol{R}$ is set according to the rules mentioned in Section 3.1. What's more, the specific parameters of GMFOA are set as $P_{c\,max}=0.9$, $P_{c\,min}=0$, $P_{m\,max}=0.5$, $P_{m\,min}=0$. Each problem is run with 50 independent replications and the results obtained by GMFOA and FOA are shown in Table 3 when solving the 29 functions with dimension equal to 30.

As shown in Table 3, the performance of GMFOA is much better than that of FOA. Although both GMFOA and FOA can generate the same "Best" which are the theoretical optima for F03, F14 and F22, GMFOA produces 23 significantly better "Best" for the 26 remaining benchmark functions. In terms of "Worst", GMFOA generates 27 significantly better results (except F12 and F21) for the 29 benchmark functions compared with FOA. Based on the results of "Best" and "Worst", we can know that GMFOA is more effective than FOA. In addition, GAFOA has the significant advantages in terms of "Mean" and "Std.". In detail, GAFOA obtains 27 significantly better "Mean" (except F12 and F21) and 27 significantly better "Std." (except F12 and F21) for the total of 29 functions. It means that GMFOA is more robust than FOA. In summary, GMFOA has the ability to produce better solutions. Obviously, this is mainly contributed by the adaptive crossover and mutation mechanism and the multi-sub-swarm mechanism. The results have shown the real benefit introduced by the two mechanisms. Actually, the major function of the adaptive crossover and mutation mechanism is to improve the population diversity. As we know, better swarm diversity can bring better exploitation ability, and make the algorithm jump out the local optima effectively.

In order to evaluate and analyze the real influence on the population diversity introduced by the GA sub-swarm, some scatter plots of individual distribution at a specific phase of search (when the iteration number m=50) are shown in Fig. 8. Actually, we provide five scatter plots of individual distribution, which are obtained by the FOA sub-swarm, the GA sub-swarm, the whole swarm of IAFOA, the pure adaptive FOA and the pure adaptive GA used in IAFOA with the same parameters as mentioned above. For saving space, these scatter plots are driven by the data when solving F13. As we know, it is very difficult to display the exact location of individual when $n > 3$. Therefore, we just select two variables and show the locations in two-dimensional space. As shown in Fig. 8(a), the fruit flies in the first sub-swarm of IAFOA search for the optimum around the best location of last generation. And as shown in Fig. 8(b), the population diversity of the GA sub-swarm is much better due to the crossover and mutation operations. There-

Table 2
Comparison of AFOA and FOA on the 29 benchmark functions (n = 30).

Functions	AFOA				FOA			
	Best	Worst	Mean	Std.	Best	Worst	Mean	Std.
F01	**3.432E−09**	**2.304E−03**	**6.231E−04**	**2.038E−04**	5.249E−07	5.896E−01	4.746E−02	1.239E−01
F02	**5.304E−03**	**9.483E−01**	**5.693E−02**	**5.392E−02**	3.472E−01	5.382E+00	9.301E−01	2.482E+00
F03	**−1.000E+00**	**−9.973E−01**	**−9.998E−01**	**3.023E−03**	**−1.000E+00**	−9.966E−01	−9.994E−01	7.925E−03
F04	**7.324E−06**	**1.292E+01**	**2.903E+00**	**2.382E+00**	2.007E−04	2.415E+03	1.427E+02	3.904E+02
F05	**4.203E−06**	**9.382E−03**	**7.873E−04**	**5.284E−04**	9.533E−05	1.228E−02	3.560E−03	2.870E−03
F06	**9.394E−07**	**7.374E−02**	**4.460E−03**	**4.871E−03**	1.576E−05	2.871E+01	2.394E+00	4.989E+00
F07	6.090E−05	**4.003E+01**	**3.468E+00**	**3.980E+00**	**3.557E−05**	1.494E+02	3.788E+01	4.883E+01
F08	**9.403E−06**	**3.240E−03**	**4.590E−04**	**7.025E−04**	2.112E−05	3.791E−02	5.642E−03	6.457E−03
F09	**4.693E−07**	**8.032E−03**	**6.489E−04**	**5.609E−04**	9.265E−05	7.972E−01	1.646E−01	2.043E−01
F10	**9.023E−08**	**5.403E−03**	**4.503E−04**	**3.845E−04**	6.255E−06	1.422E−02	1.478E−03	2.723E−03
F11	8.230E−05	**2.049E−02**	**7.342E−03**	**2.983E−03**	**6.589E−06**	6.452E−01	2.997E−02	1.166E−01
F12	**5.663E−14**	**6.704E−09**	**5.303E−10**	**1.321E−09**	9.490E−12	7.481E−08	1.891E−09	1.060E−08
F13	**4.340E−08**	**5.304E−02**	**5.403E−03**	**5.034E−03**	1.084E−05	3.371E−01	3.489E−02	7.583E−02
F14	**−4.500E+02**	**−4.102E+02**	**−4.401E+02**	**8.503E+00**	**−4.500E+02**	−4.021E+02	−4.327E+02	1.201E+01
F15	**−4.403E+02**	**−3.103E+02**	**−3.928E+02**	**4.034E+01**	−4.369E+02	−2.964E+02	−3.582E+02	4.801E+01
F16	**5.940E−09**	**2.394E−04**	**3.492E−05**	**4.032E−05**	3.738E−07	5.955E−02	1.486E−02	1.355E−02
F17	**3.045E−06**	**3.052E−03**	**1.032E−04**	**3.209E−04**	1.848E−04	3.325E−02	7.360E−03	5.627E−03
F18	**5.132E−04**	**1.023E+00**	**6.034E−01**	**6.034E−01**	2.535E−02	6.216E+00	2.708E+00	1.560E+00
F19	4.098E−04	4.098E−01	8.024E−02	6.098E−02	**1.599E−05**	**7.747E−02**	**4.991E−03**	**7.242E−03**
F20	**6.483E−02**	**3.884E−01**	**4.673E−02**	**5.022E−02**	3.332E−01	1.684E+01	5.899E+00	2.905E+00
F21	**8.094E−10**	4.887E−05	7.335E−06	**2.932E−06**	1.110E−08	**3.603E−05**	**2.654E−06**	4.632E−06
F22	**−2.900E+01**	−1.304E+01	**−1.896E+01**	5.023E+00	**−2.900E+01**	**−1.553E+01**	−1.813E+01	**4.227E+00**
F23	**−1.392E+03**	**−8.023E+02**	**−1.023E+03**	**1.296E+02**	−8.700E+02	−1.685E+02	−4.698E+02	1.850E+02
F24	**7.394E−05**	**4.956E−02**	**2.343E−03**	**4.209E−03**	2.084E−04	9.237E−02	5.786E−03	8.812E−03
F25	4.453E−02	2.545E+00	7.345E−01	9.234E−01	**6.717E−03**	**2.251E+00**	**1.860E−01**	**3.822E−01**
F26	**5.441E−08**	**6.506E−02**	**3.445E−03**	**2.343E−03**	4.175E−06	8.806E−01	1.794E−01	1.532E−01
F27	**2.461E−05**	**8.334E−02**	**5.456E−03**	**2.985E−03**	4.994E−04	1.427E−01	2.026E−02	3.118E−02
F28	**6.561E−06**	**9.004E−03**	**3.095E−04**	**3.356E−04**	4.873E−03	7.339E−01	3.298E−02	7.223E−02
F29	**2.034E−01**	**2.450E+00**	**4.875E−01**	**2.445E−01**	5.593E−01	4.289E+00	8.901E−01	7.588E−01

fore, the whole swarm of IAFOA as shown in Fig. 8(c) has the advantages of the two sub-swarms. Compared with the individual distribution of the pure adaptive FOA (Fig. 8(d)), the population diversity of IAFOA has obvious advantages. And compared with individual distribution of the pure adaptive GA (Fig. 8(e)), more individuals in IAFOA can search for the optimum accurately. The scatter plots of individual distribution when the iteration number $m = 500$ are shown in Fig. 9, which can verify the conclusions obtained from Fig. 8.

3.3. Comparison of IAFOA with other algorithms

In order to verify the advantages of IAFOA, it is compared with several well-known algorithms in two groups. In the first group, several state-of-the-art variants of FOA are included. And five advanced intelligent algorithms are used for comparison in the second group. For the algorithms in group 1, the dimensions of the 29 benchmark functions are set to 30. For the algorithms in group 2, the dimensions of the functions are set to 50. All the algorithms in the two groups are carried out on the same computer mentioned above.

3.3.1. Comparison of IAFOA with state-of-the-art variants of FOA

The five state-of-the-art variants of FOA used in this section are the contrast-based fruit fly optimization algorithm (c-mFOA) proposed by Kanarachos et al. [28], the multi-scale cooperative mutation fruit fly optimization algorithm (MSFOA) proposed by Zhang et al. [22], the normal cloud model based on FOA (CMFOA) proposed by Wu et al. [27], the effective and improved FOA (IFOA) proposed by Wang et al. [13], and the improved fruit fly optimization (IFFO) proposed by Pan et al. [4].

To be fair, the common parameters used by these algorithms are set to same values, and the specific parameters for different improved FOAs are set according to the literatures first introduced them. In detail, set population size $\boldsymbol{P}$=40 and maximum iteration number $\boldsymbol{M}$=6000 for all the five variants of FOA. What's more, for c-mFOA, $K = 320$, $\kappa = 15$, the two coefficients are 0.95 and 0.92 as in [28]; for MSFOA, set $M = 6$ and $\phi = 6$ as in [22]; for CMFOA, set $En = En_{max} \times (1 - t/M)^{\alpha}$, $He = 0.1En$, $En_{max} = (UB - LB)/4$ and $\alpha = 10$ as in [27]; for IFOA, $\omega = 0.3$, $k_{max} = (UB{-}LB)/2$ and $k_{min} = 10^{-5}$ as in [13]; for IFFO, set $\lambda_{max} = (UB{-}LB)/2$ and $\lambda_{min} = 10^{-5}$ as in [4]; for IAFOA, $P_{c\,max} = 0.9, P_{c\,min} = 0, P_{m\,max} = 0.5, P_{m\,min}$=0.

The comparative results of IAFOA and the other five variants of FOA are illustrated in Table 4. Besides the results of the best value (Best), the worst value (Worst), the mean value (Mean) and the standard deviation (Std.) in the 50 independent replications, the average computing time (ACT) is also shown in Table 4 in order to test the computational complexity of IAFOA. Furthermore, we used two-tailed t-tests to compare the results produced by these improved FOAs at the 0.05 level of significance. The statistical significance level of the aggregate results are shown in the last several rows of the Table 4. The "+" indicates that a given algorithm significantly outperforms IAFOA, "−" means IAFOA is better than the given algorithm, and "≈" indicates that there is no significant difference between IAFOA and the compared algorithm. In order to highlight the overall best results, the significantly better values are marked in bold. From Table 4, we can obtain the following information and conclusions:

(1) For the total of 29 benchmark functions, IAFOA has the best performance among all the six variants of FOA. In terms of "Best" and "Worst", which reflect the efficiency of algorithm, IAFOA produces 15/16/17/21/19 significantly better, 3/4/1/2/2 significantly worse, 11/9/11/6/8 equal "Best", and generates 18/20/22/25/26 significantly better, 5/4/3/2/3 significantly worse, 6/5/4/2/1 equal "Worse" compared with c-mFOA/MSFOA/ CMFOA/IFOA/IFFO. In terms of "Mean" and "Std.", which reflect the robustness of algorithm, IAFOA produces 18/19/23/24/27 significantly better, 5/5/3/2/1 significantly worse, 6/5/3/3/1 equal "Mean", and obtains 18/20/23/25/27 significantly better, 5/4/3/2/1 signifi-

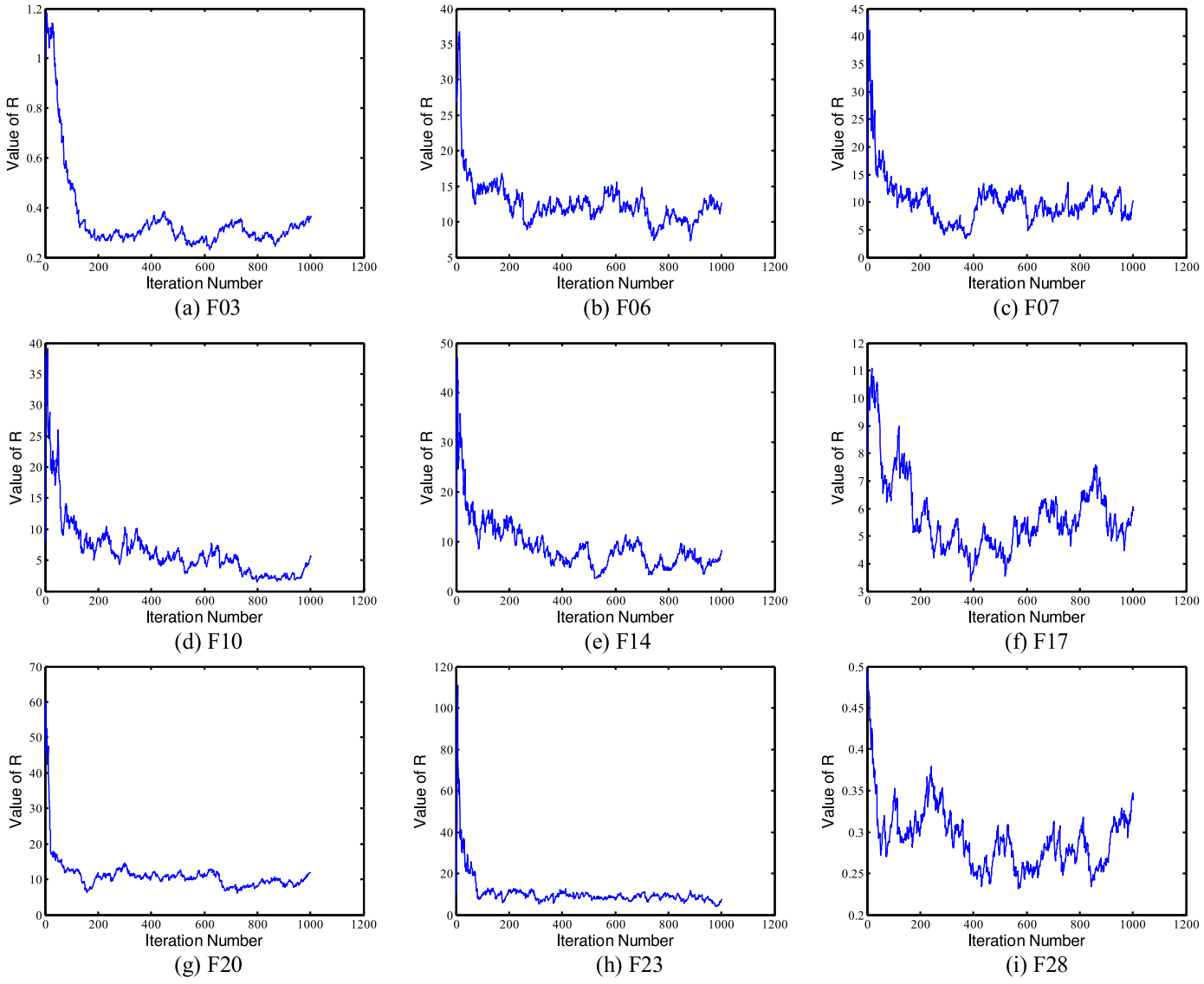

Fig. 7. Changing curves of $\boldsymbol{R}_m$.

cantly worse, 6/5/3/2/1 equal "Std." compared with *c-m*FOA/ MSFOA/ CMFOA/IFOA/IFFO.

(2) In detail, for 6 functions (F01, F03, F10, F12, F13, F14), IAFOA has the same best performance with the other one or more state-of-the-art variants of FOA. For the other 12 functions (F02, F05, F07, F08, F11, F15, F18, F21, F23, F25, F26, F29), IAFOA has significantly better performance than other five improved FOAs. It means that IAFOA can produce the best solutions for 18 benchmark functions. For the remaining 11 functions, *c-m*FOA has the best performance when solving F04, F06 and F27, MSFOA produces the best solutions for F16, F19 and F28, CMFOA has the best performance when solving F09 and F20, IFOA obtains the best solutions for F17, F22 and F24.

(3) In terms of "ACT", which reflects the computation complexity of algorithm, IAFOA spend the longest time among all the variants of FOA, while IFFO spend the shortest time. In detail, the average value of "ACT" obtained by IAFOA is 27.70 s when solving the 29 benchmark functions, while that of *c-m*FOA/ MSFOA/CMFOA/IFOA/IFFO is 21.83 s/18.28 s/17.92 s/24.11 s/10.91 s. It means that the average value of "ACT" spent by IAFOA is about 1.27/1.52/1.55/1.15/2.54 times of that used by *c-m*FOA/ MSFOA/ CMFOA/IFOA/IFFO. However, considering the best performance and outstanding solutions obtained by IAFOA, the time spent by IAFOA is acceptable.

In brief, IAFOA has the better performance than other five state-of-the-art variants of FOA when solving the 29 benchmark functions in terms of highly effective and robust within a reasonable computing time. The results obtained by IAFOA can verify the superiority of IAFOA compared with the other improved FOAs.

3.3.2. Comparison of IAFOA with advanced intelligent optimization algorithms

In the section, we intend to show how well the proposed IAFOA performs when compared with five representative intelligent optimization algorithms, such as the improved artificial fish swarm algorithm (IASFA) [38], the comprehensive learning particle swarm optimizer (CLPSO) [39], the local-best harmony search algorithm with dynamic subpopulations (DLHS) [40], the ant lion optimizer (ALO) [41] and the grey wolf optimization (GWO) [42]. Among them, IASFA is the representative of fish swarm algorithm and CLPSO is the top access article in the PSO community [27]. DLHS is a new improved algorithm of harmony search algorithm

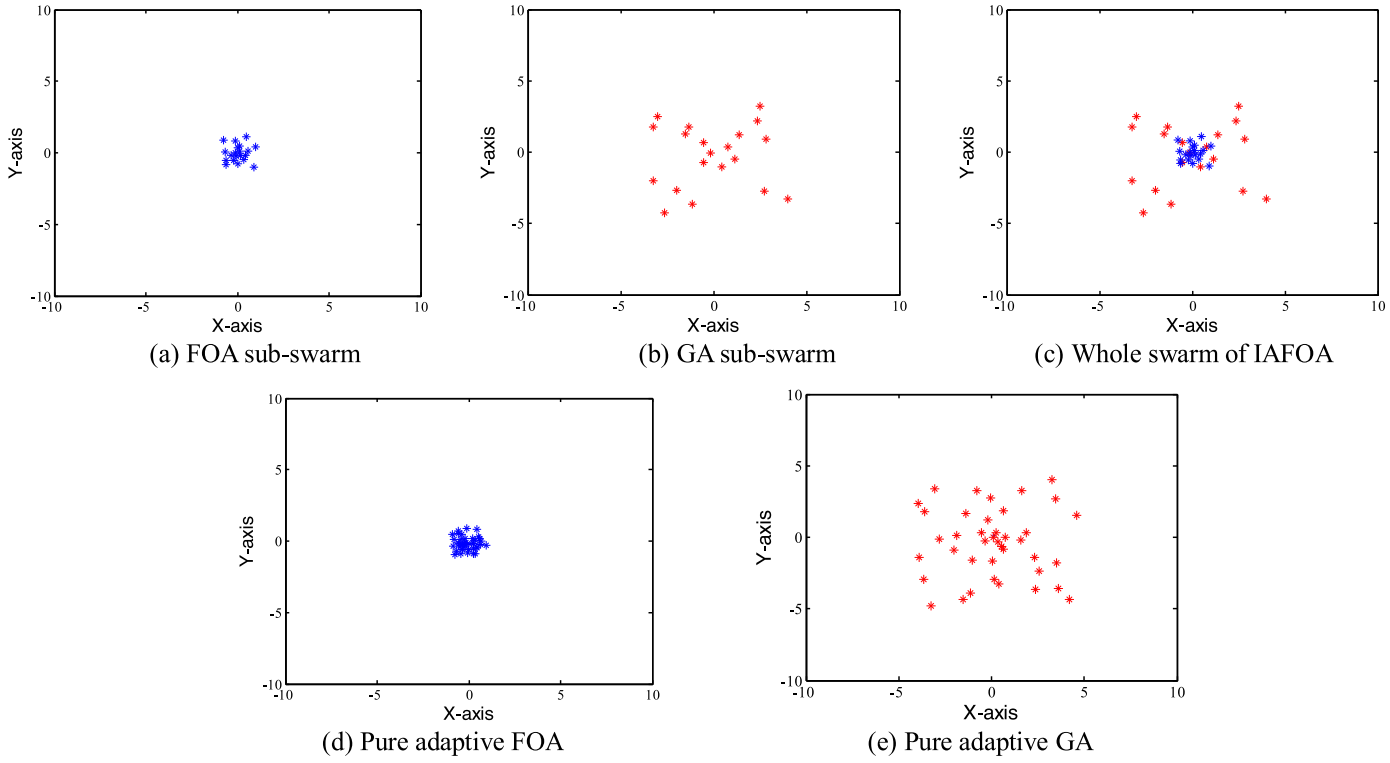

Fig. 8. Scatter plots of individual distribution when $m = 50$.

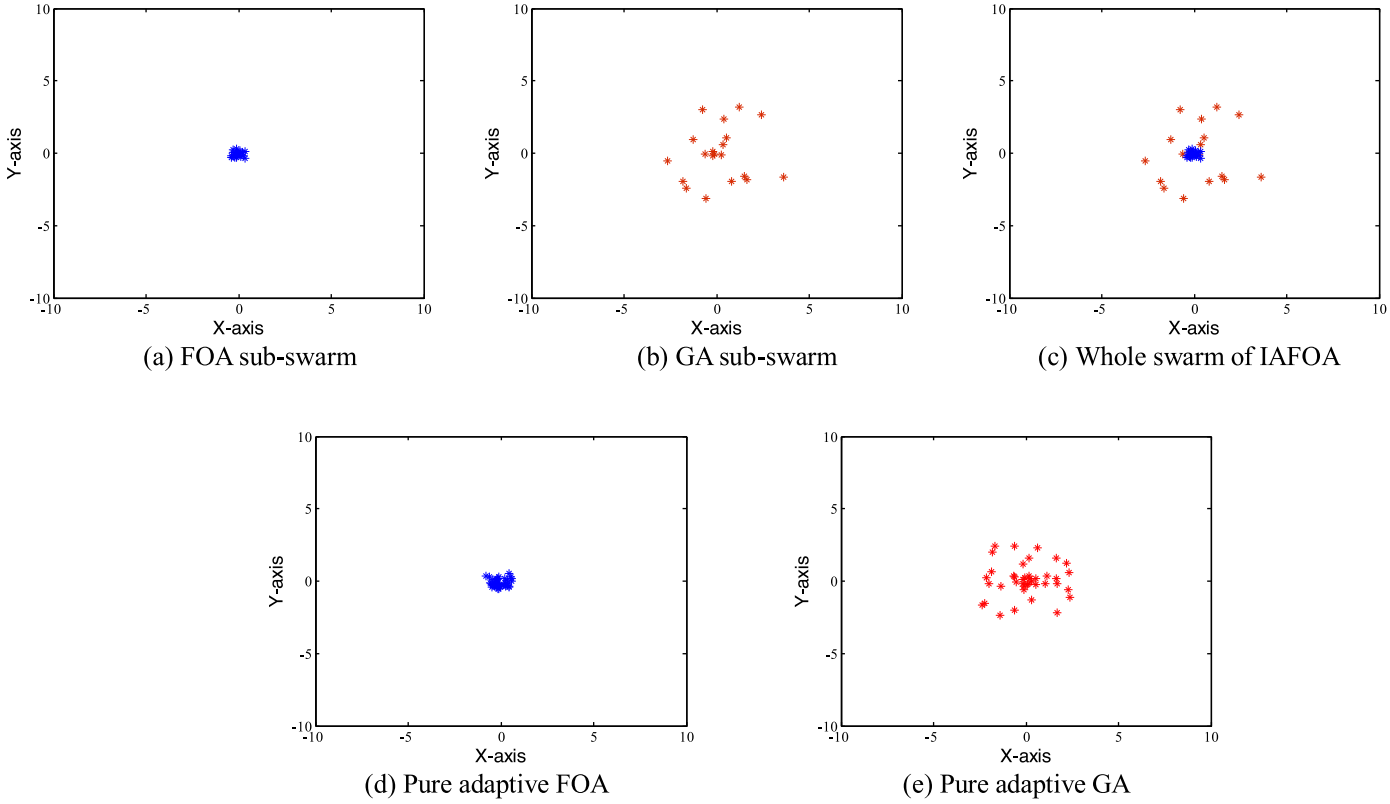

Fig. 9. Scatter plots of individual distribution when $m = 500$.

Table 3
Comparison of GMFOA and FOA on the 29 benchmark functions ($n = 30$).

Functions	GMFOA				FOA			
	Best	Worst	Mean	Std.	Best	Worst	Mean	Std.
F01	**6.231E−08**	**2.611E−04**	**3.216E−05**	**5.502E−05**	5.249E−07	5.896E−01	4.746E−02	1.239E−01
F02	**3.521E−04**	**7.182E−02**	**6.418E−03**	**7.268E−03**	3.472E−01	5.382E+00	9.301E−01	2.482E+00
F03	**−1.000E+00**	**−9.975E−01**	**−9.998E−01**	**2.681E−03**	**−1.000E+00**	−9.966E−01	−9.994E−01	7.925E−03
F04	**1.620E−05**	**1.292E−01**	**3.256E−02**	**2.265E−02**	2.007E−04	2.415E+03	1.427E+02	3.904E+02
F05	**8.162E−06**	**2.004E−03**	**3.264E−04**	**3.648E−04**	9.533E−05	1.228E−02	3.560E−03	2.870E−03
F06	**2.365E−07**	**1.592E−02**	**2.395E−03**	**2.265E−03**	1.576E−05	2.871E+01	2.394E+00	4.989E+00
F07	**1.390E−05**	**6.321E−01**	**3.289E−02**	**6.318E−02**	3.557E−05	1.494E+02	3.788E+01	4.883E+01
F08	**2.658E−07**	**9.358E−04**	**7.925E−05**	**8.647E−05**	2.112E−05	3.791E−02	5.642E−03	6.457E−03
F09	**1.365E−06**	**2.340E−02**	**3.611E−03**	**3.102E−03**	9.265E−05	7.972E−01	1.646E−01	2.043E−01
F10	**6.329E−08**	**1.280E−03**	**2.970E−04**	**1.699E−04**	6.255E−06	1.422E−02	1.478E−03	2.723E−03
F11	**3.220E−08**	**5.321E−03**	**3.841E−04**	**6.480E−04**	6.589E−06	6.452E−01	2.997E−02	1.166E−01
F12	1.236E−11	3.261E−07	4.521E−08	5.012E−08	**9.490E−12**	**7.481E−08**	**1.891E−09**	**1.060E−08**
F13	**2.330E−07**	**1.399E−03**	**6.481E−04**	**2.034E−04**	1.084E−05	3.371E−01	3.489E−02	7.583E−02
F14	**−4.500E+02**	**−4.108E+02**	**−4.409E+02**	**7.803E+00**	**−4.500E+02**	−4.021E+02	−4.327E+02	1.201E+01
F15	**−4.420E+02**	**−3.320E+02**	**−4.126E+02**	**3.125E+01**	−4.369E+02	−2.964E+02	−3.582E+02	4.801E+01
F16	**2.360E−09**	**6.445E−05**	**7.223E−06**	**6.314E−06**	3.738E−07	5.955E−02	1.486E−02	1.355E−02
F17	**7.182E−05**	**5.330E−03**	**4.112E−04**	**7.140E−04**	1.848E−04	3.325E−02	7.360E−03	5.627E−03
F18	**1.020E−04**	**6.341E−01**	**8.117E−02**	**9.150E−02**	2.535E−02	6.216E+00	2.708E+00	1.560E+00
F19	**9.630E−06**	**4.260E−02**	**1.003E−03**	**4.261E−03**	1.599E−05	7.747E−02	4.991E−03	7.242E−03
F20	**1.568E−02**	**6.328E−01**	**3.201E−02**	**1.236E−01**	3.332E−01	1.684E+01	5.899E+00	2.905E+00
F21	2.310E−08	1.477E−04	3.894E−05	1.562E−05	**1.110E−08**	**3.603E−05**	**2.654E−06**	**4.632E−06**
F22	**−2.900E+01**	**−1.755E+01**	**−2.213E+01**	**3.563E+00**	**−2.900E+01**	−1.553E+01	−1.813E+01	4.227E+00
F23	**−1.866E+03**	**−1.023E+03**	**−1.423E+03**	**1.203E+02**	−8.700E+02	−1.685E+02	−4.698E+02	1.850E+02
F24	**6.322E−06**	**8.112E−03**	**4.236E−04**	**2.130E−03**	2.084E−04	9.237E−02	5.786E−03	8.812E−03
F25	**7.209E−04**	**9.125E−01**	**8.251E−02**	**1.963E−01**	6.717E−03	2.251E+00	1.860E−01	3.822E−01
F26	7.236E−06	**2.148E−02**	**6.384E−03**	**4.112E−03**	**4.175E−06**	8.806E−01	1.794E−01	1.532E−01
F27	**9.010E−05**	**3.624E−03**	**8.630E−04**	**1.025E−03**	4.994E−04	1.427E−01	2.026E−02	3.118E−02
F28	**1.332E−05**	**5.249E−02**	**4.113E−03**	**1.632E−02**	4.873E−03	7.339E−01	3.298E−02	7.223E−02
F29	**6.149E−03**	**4.223E−01**	**1.025E−02**	**2.002E−01**	5.593E−01	4.289E+00	8.901E−01	7.588E−01

with dynamic subpopulations, and ALO and GWO are the advanced swarm intelligent optimization algorithms [43]. The parameters of these algorithms are the same as the corresponding literatures suggested. And the parameters of the IAFOA are the same as the previous.

Table 5 shows the mean value "Mean" and standard deviation "Std." by applying the six algorithms to optimize the 29 benchmark functions with dimensions equal to 50. In Table 5, the best results on each row are in bold and the aggregate results of statistical testing are shown in the last several rows. From Table 5, we can obtain the following information and conclusions:

(1) For the 15 unimodal functions, the results show that GWO has the best performance, followed by IAFOA. In detail, GWO obtains 10 best "Mean" and "Std." while IAFOA generates 8 best "Mean" and "Std." for the 15 functions. However, the performance of IAFOA is much better than that of IASFA, CLPSO, DLHA and ALO.
(2) For the 14 multimodal functions, the results show that IAFOA has the best performance. In detail, IAFOA generates 11 best "Mean" and "Std." for the 14 multimodal functions, but the numbers obtained by IASFA, CLPSO, DLHA, ALO and GWO are 0, 0, 0, 0 and 3, respectively.
(3) Overall, for the 29 benchmark functions, IAFOA produces 27/25/24/25/16 significantly better, 1/2/3/2/9 significantly worse, and 1/2/2/2/4 equal "Best" compared with IASFA/CLPSO/DLHA/ALO/GWO. On the other hand, IAFOA performs steadily well in terms of "Mean", it generates 27/25/25/25/16 significantly better, 1/2/2/2/9 significantly worse, and 1/2/2/2/4 equal "Std." compared with IASFA/CLPSO/DLHA/ALO/GWO.

Therefore, we can conclude that IAFOA can achieve excellent solutions for unimodal and multimodal functions, even though its performance for unimodal functions is a little worse than that of GWO. In summary, IAFOA is the best algorithm when solving the 29 functions compared with IASFA/CLPSO/DLHA/ALO/GWO. Actually, the super performance of the proposed IAFOA is mainly benefited from the four proposed mechanisms in this paper.

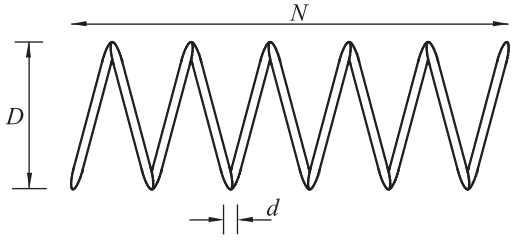

Fig. 10. Schematic diagram of coil compression spring design [46].

4. Application to solve engineering optimization problems

In order to verify the ability of IAFOA to solve the optimization problems in real-world applications [44], three well-known engineering problems, which are coil compression spring design, welded beam design and speed reducer design, are provided in this section [45,46].

4.1. Three engineering optimization problems

4.1.1. Coil compression spring design

The objective of this optimization problem is to design a coil compression spring with a minimum total weight. The coil compression spring, as shown in Fig. 10, is subjected to three design variables, which are the mean coil diameter D (x_1), the wire diameter d (x_2) and the number of active coils N (x_3), and four constrains, which are the minimum deflection, the shear stress, the surge frequency and the diameter [46].

Table 4
Comparative results of IAFOA with several variants of FOA on the 29 benchmark functions (n=30).

Function		IAFOA	*c-m*FOA	MSFOA	CMFOA	IFOA	IFFO
F01	Best	**0.000E+00**	**0.000E+00** ≈	**0.000E+00** ≈	**0.000E+00** ≈	**0.000E+00** ≈	**0.000E+00** ≈
	Worst	**0.000E+00**	**0.000E+00** ≈	4.814E−10 −	**0.000E+00** ≈	1.485E−08 −	6.248E−11 −
	Mean	**0.000E+00**	**0.000E+00** ≈	6.157E−11 −	**0.000E+00** ≈	1.464E−09 −	4.852E−12 −
	Std.	**0.000E+00**	**0.000E+00** ≈	1.164E−10 −	**0.000E+00** ≈	2.478E−09 −	1.967E−12 −
	ACT	18.83s	15.18s	11.57s	13.01s	16.63s	**7.23s**
F02	Best	**7.591E−07**	1.268E−05 −	1.658E−06 −	3.705E−03 −	1.578E−04 −	4.571E−05 −
	Worst	**6.390E−06**	5.160E−04 −	6.548E−01 −	2.354E+00 −	1.306E+00 −	3.413E+00 −
	Mean	**2.158E−06**	1.815E−04 −	4.581E−02 −	7.902E−01 −	4.505E−01 −	9.478E−01 −
	Std.	**8.820E−07**	1.233E−04 −	3.547E−02 −	8.501E−01 −	2.658E−01 −	2.045E−01 −
	ACT	21.59s	17.34s	14.37s	14.28s	21.02s	**8.76s**
F03	Best	**−1.000E+00**	**−1.000E+00** ≈	**−1.000E+00** ≈	**−1.000E+00** ≈	**−1.000E+00** ≈	**−1.000E+00** ≈
	Worst	**−1.000E+00**	**−1.000E+00** ≈	**−1.000E+00** ≈	**−1.000E+00** ≈	**−1.000E+00** ≈	**−1.000E+00** ≈
	Mean	**−1.000E+00**	**−1.000E+00** ≈	**−1.000E+00** ≈	**−1.000E+00** ≈	**−1.000E+00** ≈	**−1.000E+00** ≈
	Std.	**0.000E+00**	**0.000E+00** ≈	**0.000E+00** ≈	**0.000E+00** ≈	**0.000E+00** ≈	**0.000E+00** ≈
	ACT	22.10s	16.76s	13.82s	13.74s	20.39s	**8.64s**
F04	Best	**0.000E+00**	**0.000E+00** ≈	**0.000E+00** ≈	**0.000E+00** ≈	3.110E−07 −	2.515E−10 −
	Worst	2.918E−06	**6.182E−10** +	1.561E+00 −	1.025E−08 +	6.158E−01 −	3.651E−06 −
	Mean	1.029E−07	**3.058E−10** +	2.548E−01 −	1.660E−09 +	1.350E−02 −	1.610E−07 −
	Std.	9.156E−08	**7.105E−11** +	2.611E−01 −	3.610E−09 +	1.977E−02 −	4.272E−07 −
	ACT	23.41s	18.70s	15.69s	15.87s	22.34s	**9.12s**
F05	Best	1.182E−08	6.219E−08 −	**1.548E−09** +	3.148E−05 −	9.124E−07 −	3.659E−06 −
	Worst	**8.916E−06**	3.174E−05 −	8.259E−04 −	2.272E−01 −	1.580E−03 −	2.025E−03 −
	Mean	**3.913E−06**	8.091E−06 −	2.251E−05 −	4.929E−02 −	6.454E−04 −	1.005E−04 −
	Std.	**2.153E−06**	5.130E−06 −	3.811E−05 −	3.457E−02 −	2.820E−04 −	2.133E−04 −
	ACT	20.21s	16.40s	12.82s	13.37s	18.94s	**7.96s**
F06	Best	1.025E−08	**2.069E−10** +	2.068E−08 −	5.474E−06 −	2.351E−07 −	1.580E−07 −
	Worst	5.620E−06	**1.863E−08** +	3.561E−05 −	2.667E−03 −	2.104E−02 −	2.269E−02 −
	Mean	1.068E−06	**4.613E−09** +	4.909E−06 −	4.511E−04 −	1.698E−03 −	9.002E−03 −
	Std.	1.269E−06	**4.752E−09** +	3.180E−06 −	3.828E−04 −	2.414E−03 −	4.460E−03 −
	ACT	22.25s	17.37s	14.58s	14.32s	20.43s	**8.73s**
F07	Best	3.547E−08	5.920E−05 −	7.881E−07 −	6.187E−06 −	2.581E−05 −	**2.681E−08** +
	Worst	**8.962E−02**	7.630E−01 −	8.581E+00 −	8.413E−01 −	3.068E+00 −	4.859E+00 −
	Mean	**6.291E−03**	1.523E−01 −	1.280E−01 −	1.058E−01 −	6.184E−01 −	7.155E−01 −
	Std.	**7.269E−03**	6.190E−02 −	4.980E−01 −	7.841E−02 −	4.552E−01 −	4.308E−01 −
	ACT	24.21	19.52s	16.47s	17.75s	21.10s	**9.86s**
F08	Best	**8.210E−10**	3.201E−07 −	2.658E−09 −	4.447E−07 −	2.184E−08 −	5.817E−06 −
	Worst	**9.320E−06**	6.390E−05 −	6.365E−05 −	2.360E−05 −	6.318E−04 −	2.658E−04 −
	Mean	**2.360E−06**	1.582E−05 −	4.030E−06 −	5.505E−06 −	5.681E−05 −	7.218E−05 −
	Std.	**6.301E−07**	7.605E−06 −	4.941E−06 −	5.898E−06 −	6.117E−05 −	8.258E−05 −
	ACT	16.40s	12.96s	11.19s	10.13s	13.36s	**6.58s**
F09	Best	8.693E−13	1.250E−12 −	2.617E−11 −	**2.658E−14** +	1.178E−07 −	4.180E−09 −
	Worst	1.052E−08	6.392E−10 +	3.647E−07 −	**2.058E−11** +	6.247E−04 −	2.618E−05 −
	Mean	2.536E−09	2.360E−10 +	5.218E−08 −	**4.855E−12** +	5.440E−05 −	7.310E−06 −
	Std.	2.069E−09	6.620E−11 +	4.151E−08 −	**5.001E−12** +	6.940E−05 −	7.097E−06 −
	ACT	18.45s	15.26s	13.35s	11.90s	15.64s	**7.63s**
F10	Best	**0.000E+00**	**0.000E+00** ≈	**0.000E+00** ≈	**0.000E+00** ≈	8.957E−12 −	**0.000E+00** ≈
	Worst	**0.000E+00**	**0.000E+00** ≈	**0.000E+00** ≈	**0.000E+00** ≈	6.187E−08 −	2.367E−12 −
	Mean	**0.000E+00**	**0.000E+00** ≈	**0.000E+00** ≈	**0.000E+00** ≈	3.617E−09 −	2.958E−13 −
	Std.	**0.000E+00**	**0.000E+00** ≈	**0.000E+00** ≈	**0.000E+00** ≈	6.581E−09 −	6.628E−13 −
	ACT	19.35s	15.08s	12.85s	12.41s	15.90s	**7.43s**
F11	Best	**8.320E−09**	5.602E−06 −	6.140E−07 −	6.170E−06 −	7.184E−05 −	2.151E−06 −
	Worst	**1.850E−05**	2.080E−04 −	7.114E−04 −	5.218E−03 −	1.471E−02 −	3.184E−04 −
	Mean	**3.206E−06**	5.890E−05 −	1.250E−04 −	9.147E−04 −	4.571E−03 −	1.251E−04 −
	Std.	**5.190E−06**	4.149E−05 −	1.517E−04 −	1.471E−03 −	7.418E−03 −	2.584E−04 −
	ACT	21.97s	16.85s	14.04s	13.55s	17.35s	**8.26s**
F12	Best	**0.000E+00**	**0.000E+00** ≈	**0.000E+00** ≈	**0.000E+00** ≈	**0.000E+00** ≈	**0.000E+00** ≈
	Worst	**0.000E+00**	**0.000E+00** ≈	**0.000E+00** ≈	5.368E−12 −	3.291E−10 −	8.152E−13 −
	Mean	**0.000E+00**	**0.000E+00** ≈	**0.000E+00** ≈	1.058E−13 −	2.681E−11 −	1.560E−14 −
	Std.	**0.000E+00**	**0.000E+00** ≈	**0.000E+00** ≈	1.581E−12 −	6.170E−11 −	7.077E−14 −
	ACT	22.87s	17.46s	15.50s	15.50s	20.85s	**8.91s**
F13	Best	**0.000E+00**	**0.000E+00** ≈	**0.000E+00** ≈	**0.000E+00** ≈	4.281E−11 −	**0.000E+00** ≈
	Worst	**0.000E+00**	**0.000E+00** ≈	**0.000E+00** ≈	4.518E−08 −	5.617E−04 −	6.187E−10 −
	Mean	**0.000E+00**	**0.000E+00** ≈	**0.000E+00** ≈	7.447E−09 −	9.714E−05 −	4.077E−11 −
	Std.	**0.000E+00**	**0.000E+00** ≈	**0.000E+00** ≈	1.248E−08 −	5.257E−05 −	7.686E−11 −
	ACT	19.18s	15.05s	13.22s	13.14s	15.36s	**7.64s**
F14	Best	**−4.500E+02**	**−4.500E+02** ≈	**−4.500E+02** ≈	**−4.500E+02** ≈	**−4.500E+02** ≈	**−4.500E+02** ≈
	Worst	**−4.500E+02**	**−4.500E+02** ≈	**−4.500E+02** ≈	−4.228E+02 −	**−4.500E+02** ≈	−4.358E+02 −
	Mean	**−4.500E+02**	**−4.500E+02** ≈	**−4.500E+02** ≈	−4.401E+02 −	**−4.500E+02** ≈	−4.415E+02 −
	Std.	**0.000E+00**	**0.000E+00** ≈	**0.000E+00** ≈	3.217E+00 −	**0.000E+00** ≈	4.027E+00 −
	ACT	18.96s	14.63s	12.23s	12.16s	18.93s	**7.28s**
F15	Best	**−4.500E+02**	**−4.500E+02** ≈	**−4.500E+02** ≈	**−4.500E+02** ≈	**−4.500E+02** ≈	**−4.500E+02** ≈
	Worst	**−4.020E+02**	−3.955E+02 −	−3.987E+02 −	−3.154E+02 −	−3.912E+02 −	−3.847E+02 −
	Mean	**−4.397E+02**	−4.276E+02 −	−4.224E+02 −	−4.001E+02 −	−4.205E+02 −	−4.150E+02 −
	Std.	**1.001E+01**	1.250E+01 −	1.455E+01 −	2.874E+01 −	1.314E+01 −	2.247E+01 −
	ACT	21.62s	18.15s	15.11s	14.57s	19.13s	**8.94s**

(*continued on next page*)

Table 4 (*continued*)

Function		IAFOA	*c-m*FOA	MSFOA	CMFOA	IFOA	IFFO
F16	Best	**6.208E−14**	5.414E−10 −	3.471E−13 −	6.227E−10 −	7.481E−06 −	3.247E−12 −
	Worst	8.620E−08	4.192E−06 −	**1.471E−09 +**	4.257E−04 −	1.478E−01 −	4.771E−08 +
	Mean	6.821E−09	7.929E−07 −	**5.810E−10 +**	6.418E−05 −	3.610E−02 −	8.952E−09 −
	Std.	2.956E−09	7.120E−07 −	**4.336E−10 +**	8.557E−05 −	2.149E−02 −	4.417E−09 −
	ACT	25.12s	20.05s	17.08s	18.00s	23.53s	**10.23s**
F17	Best	2.599E−11	5.927E−10 −	3.647E−11 +	9.471E−08 −	**2.357E−14 +**	3.217E−12 +
	Worst	6.112E−08	2.415E−06 −	1.480E−08 +	3.334E−04 −	**2.154E−10 +**	3.597E−09 +
	Mean	2.018E−08	7.139E−07 −	3.268E−09 +	6.171E−05 −	**7.551E−11 +**	9.044E−10 +
	Std.	1.592E−08	8.157E−07 −	5.007E−09 +	4.222E−05 −	**5.370E−11 +**	3.159E−10 +
	ACT	28.96s	23.59s	19.29s	18.01s	25.91s	**11.62s**
F18	Best	**8.174E−09**	7.608E−06 −	3.022E−06 −	3.215E−05 −	6.147E−04 −	3.299E−06 −
	Worst	**1.592E−05**	4.990E−02 −	3.179E−04 −	1.547E+00 −	3.476E+01 −	2.581E−01 −
	Mean	**5.180E−06**	6.491E−03 −	6.475E−05 −	2.541E−01 −	2.557E+00 −	3.522E−02 −
	Std.	**4.109E−06**	5.889E−03 −	1.112E−04 −	5.828E−01 −	6.698E+00 −	5.149E−02 −
	ACT	39.82s	31.60s	26.07s	25.91s	32.39s	**15.80s**
F19	Best	7.981E−09	1.250E−09 +	**1.562E−10 +**	3.225E−05 −	3.214E−06 −	3.141E−03 −
	Worst	4.912E−05	6.811E−06 +	**2.936E−07 +**	4.449E−02 −	2.358E−03 −	1.115E−01 −
	Mean	5.156E−06	2.447E−06 +	**5.229E−08 +**	6.417E−03 −	1.227E−04 −	4.997E−02 −
	Std.	7.108E−06	6.491E−07 +	**4.513E−08 +**	7.949E−03 −	2.979E−04 −	2.461E−02 −
	ACT	44.32s	34.69s	28.73s	27.33s	36.62s	**17.52s**
F20	Best	**0.000E+00**	**0.000E+00 ≈**	3.284E−05 −	**0.000E+00 ≈**	1.187E−06 −	6.332E−03 −
	Worst	5.827E−06	6.840E−04 −	4.291E+00 −	**3.177E−07 +**	6.307E−01 −	2.611E−02 −
	Mean	5.690E−07	7.401E−05 −	9.272E−01 −	**6.217E−08 +**	7.192E−02 −	1.041E−02 −
	Std.	6.147E−07	7.355E−05 −	8.040E−01 −	**5.228E−08 +**	6.318E−02 −	4.114E−03 −
	ACT	30.54s	23.34s	19.61s	18.65s	28.15s	**12.03s**
F21	Best	**0.000E+00**	**0.000E+00 ≈**	3.212E−08 −	**0.000E+00 ≈**	6.331E−10 −	2.367E−12 −
	Worst	**0.000E+00**	9.481E−09 −	5.326E−05 −	8.368E−03 −	9.147E−05 −	3.652E−06 −
	Mean	**0.000E+00**	3.551E−10 −	4.662E−06 −	2.774E−04 −	2.110E−05 −	1.480E−07 −
	Std.	**0.000E+00**	4.502E−10 −	5.080E−06 −	2.14E0−03 −	3.008E−05 −	3.429E−07 −
	ACT	33.89s	27.16s	22.53s	21.73s	30.08s	**13.25s**
F22	Best	**−2.900E+01**	**−2.900E+01 ≈**	**−2.900E+01 ≈**	**−2.900E+01 ≈**	**−2.900E+01 ≈**	**−2.900E+01 ≈**
	Worst	−2.562E+01	−2.212E+01 −	−2.358E+01 −	**−2.567E+01 ≈**	−2.219E+01 −	−1.784E+01 −
	Mean	−2.778E+01	−2.658E+01 −	−2.667E+01 −	−2.612E+01 −	**−2.780E+01 ≈**	−2.100E+01 −
	Std.	**1.158E+00**	1.359E+00 −	1.746E+00 −	2.410E+00 −	1.585E+00 −	1.795E+00 −
	ACT	42.29s	33.13s	26.63s	24.04s	35.89s	**16.24s**
F23	Best	**−4.890E+03**	−4.520E+03 −	−4.802E+03 −	−4.059E+03 −	−3.935E+03 −	−3.621E+03 −
	Worst	**−4.215E+03**	−3.845E+03 −	−3.946E+03 −	−2.625E+03 −	−2.236E+03 −	−1.845E+03 −
	Mean	−4.387E+03	−4.050E+03 −	**−4.467E+03 +**	−3.593E+03 −	−3.481E+03 −	−2.082E+03 −
	Std.	**1.015E+02**	1.362E+02 −	2.030E+02 −	2.570E+02 −	2.947E+02 −	2.210E+02 −
	ACT	35.96s	27.46s	24.10s	23.82s	30.26s	**14.01s**
F24	Best	6.110E−09	7.118E−07 −	3.258E−06 −	3.548E−03 −	**6.348E−11 +**	6.347E−04 −
	Worst	2.041E−05	2.894E−04 −	5.114E−03 −	5.948E+00 −	**1.284E−05 +**	3.641E+00 −
	Mean	8.449E−06	2005E−05 −	3.601E−04 −	1.154E+00 −	**3.149E−06 +**	2.770E−01 −
	Std.	5.011E−06	3.647E−05 −	6.674E−04 −	2.092E+00 −	**3.440E−06 +**	4.182E−01 −
	ACT	33.98s	27.54s	23.25s	22.70s	30.17s	**13.84s**
F25	Best	**5.290E−08**	7.181E−07 −	8.525E−07 −	8.617E−08 −	6.248E−04 −	7.550E−04 −
	Worst	**1.952E−06**	2.525E−04 −	3.397E−03 −	4.669E−05 −	3.221E−01 −	2.358E−02 −
	Mean	**2.265E−07**	4.105E−05 −	3.683E−04 −	3.479E−06 −	2.553E−02 −	4.788E−03 −
	Std.	**5.608E−07**	6.232E−05 −	5.226E−04 −	5.801E−06 −	5.402E−02 −	7.245E−03 −
	ACT	25.50s	20.97s	17.29s	16.57s	21.07s	**10.23s**
F26	Best	**1.158E−11**	6.471E−08 −	3.657E−05 −	6.810E−07 −	9.241E−05 −	3.269E−10 −
	Worst	**3.058E−08**	1.952E−06 −	3.327E−02 −	3.118E−03 −	6.377E−01 −	4.297E−08 −
	Mean	**2.187E−09**	4.571E−07 −	5.330E−03 −	2.942E−04 −	8.411E−02 −	3.572E−09 −
	Std.	**4.158E−09**	2.118E−07 −	4.991E−03 −	7.778E−04 −	7.163E−02 −	8.470E−09 −
	ACT	29.56s	23.12s	18.96s	18.38s	23.70s	**11.56s**
F27	Best	1.258E−08	**3.229E−13 +**	9.512E−07 −	4.556E−04 −	3.467E−05 −	3.660E−04 −
	Worst	1.952E−05	**2.247E−10 +**	1.116E−03 −	3.615E−02 −	2.990E−01 −	2.650E−02 −
	Mean	3.140E−06	**4.265E−11 +**	3.681E−04 −	7.712E−03 −	2.682E−02 −	2.333E−03 −
	Std.	4.155E−06	**5.166E−11 +**	2.560E−04 −	4.959E−03 −	1.030E−01 −	3.878E−03 −
	ACT	40.00s	30.41s	26.42s	25.19s	35.33s	**15.36s**
F28	Best	4.925E−12	4.584E−08 −	**3.229E−13 +**	3.212E−09 −	1.548E−07 −	3.265E−06 −
	Worst	1.054E−08	1.479E−05 −	**2.247E−10 +**	4.319E−05 −	3.258E−03 −	2.165E−03 −
	Mean	1.889E−09	3.206E−06 −	**4.265E−11 +**	7.912E−06 −	2.309E−04 −	1.088E−04 −
	Std.	2.697E−09	2.545E−06 −	**5.166E−11 +**	6.585E−06 −	4.221E−04 −	4.126E−04 −
	ACT	39.53s	30.18s	26.18s	25.72s	32.65s	**15.40s**
F29	Best	**8.696E−06**	3.617E−05 −	5.147E−03 −	3.616E−03 −	6.335E−04 −	2.225E−05 −
	Worst	**4.559E−03**	4.447E−02 −	1.566E+00 −	2.221E−01 −	3.155E+00 −	3.214E−02 −
	Mean	**6.995E−04**	5.694E−03 −	1.159E−01 −	6.606E−02 −	1.945E−01 −	2.352E−03 −
	Std.	**8.441E−04**	7.889E−03 −	5.440E−01 −	5.693E−02 −	9.322E−01 −	3.065E−03 −
	ACT	42.32s	33.23s	27.17s	27.83s	36.18s	**16.37s**
Best	+	/	3	4	1	2	2
	−	/	15	16	17	21	19
	≈	/	11	9	11	6	8
Worst	+	/	5	4	3	2	2
	−	/	18	20	22	25	26
	≈	/	6	5	4	2	1
Mean	+	/	5	5	3	2	1
	−	/	18	19	23	24	27
	≈	/	6	5	3	3	1
Std.	+	/	5	4	3	2	1
	−	/	18	20	23	25	27
	≈	/	6	5	3	2	1
Average of ACT		27.70s	21.83s	18.28s	17.92s	24.11s	10.91s

Table 5
Comparative results of IAFOA with advanced intelligent algorithms on the 29 benchmark functions ($n = 50$).

Function	IAFOA		IASFA		CLPSO		DLHS		ALO		GWO	
	Mean	Std.	Mean t−test	Std. t−test	Mean t−test	Std. t−test	Mean t−test	Std. t−test	Mean t−test	Std. t−test	Mean t−test	Std. t−test
F01	8.611E−08	4.581E−07	1.845E+03 −	1.426E+02 −	4.650E−03 **−**	1.126E−03 −	4.162E−08 +	1.465E−07 +	5.261E−02 −	8.162E−02 −	**0.000E+00 +**	**0.000E+00 +**
F02	**1.541E−04**	**3.177E−03**	7.621E+03 −	1.024E+03 −	1.878E+01 −	1.994E+00 −	2.144E+00 −	1.947E+00 −	1.112E+00 −	1.241E+00 −	5.268E−01 −	1.004E−02 −
F03	**−1.000E+00**	**0.000E+00**	**−1.000E+00 ≈**	**0.000E+00 ≈**	**−1.000E+00 ≈**	**0.000E+00 ≈**	**−1.000E+00 ≈**	**0.000E+00 ≈**	**−1.000E+00 ≈**	**0.000E+00 ≈**	**−1.000E+00 ≈**	**0.000E+00 ≈**
F04	6.422E−04	3.880E−03	2.486E+02 −	1.947E+02 −	2.144E+00 −	8.732E+00 −	1.002E+03 −	4.562E+03 −	2.315E+02 −	6.347E+02 −	**0.000E+00 +**	**0.000E+00 +**
F05	6.809E−03	1.513E−02	3.257E+01 −	4.587E+00 −	4.558E−03 +	1.005E−03 +	2.155E−03 +	4.587E−03 +	5.687E−02 −	2.657E−02 −	**2.547E−04 +**	**1.147E−04 +**
F06	**2.014E−01**	**3.396E−01**	5.162E+03 −	1.119E+03 −	1.258E+02 −	8.591E+01 −	2.158E+00 −	5.216E+00 −	1.847E+02 −	1.895E+02 −	4.226E+01 −	8.597E+00 −
F07	1.024E−01	2.963E−01	1.447E−04 +	2.667E−04 +	7.116E−02 +	2.584E−02 +	1.119E+02 −	4.116E+02 −	1.026E+00 −	1.558E+00 −	**0.000E+00 +**	**0.000E+00 +**
F08	2.361E−03	4.550E−03	3.005E+01 −	1.110E+01 −	2.228E+01 −	1.557E+00 −	1.258E−02 −	1.663E−02 −	1.211E+01 −	2.008E+00 −	**0.000E+00 +**	**0.000E+00 +**
F09	6.034E−08	7.008E−08	2.569E+01 −	5.621E+01 −	1.125E−01 −	2.113E−01 −	4.239E−04 −	6.320E−03 −	1.634E+02 −	1.634E+02 −	**0.000E+00 +**	**0.000E+00 +**
F10	**0.000E+00**	**0.000E+00**	6.217E+00 −	2.317E+00 −	5.627E−02 −	2.007E−02 −	2.957E−06 −	3.258E−05 −	2.617E−07 −	1.005E−07 −	**0.000E+00 ≈**	**0.000E+00 ≈**
F11	4.195E−04	7.891E−04	3.269E+02 −	3.247E+02 −	6.448E−02 −	1.635E−02 −	2.657E+00 −	4.697E+00 −	**3.697E−07 +**	**2.035E−07 +**	1.125E+00 −	3.227E−01 −
F12	**0.000E+00**	**0.000E+00**	1.151E−02 −	1.090E−02 −	**0.000E+00 ≈**	**0.000E+00 ≈**	1.652E−14 −	3.216E−13 −	2.014E−08 −	1.647E−08 −	**0.000E+00 ≈**	**0.000E+00 ≈**
F13	**0.000E+00**	**0.000E+00**	6.325E−01 −	5.209E−01 −	1.562E−04 −	2.364E−04 −	2.697E−07 −	3.647E−06 −	4.897E−02 −	5.110E−02 −	**0.000E+00 ≈**	**0.000E+00 ≈**
F14	**−4.450E+02**	**0.000E+00**	−4.362E+02 −	2.569E+00 −	−4.236E+02 −	1.860E+00 −	**−4.500E+02** ≈	**0.000E+00** ≈	**−4.500E+02** ≈	**0.000E+00** ≈	−4.430E+02 −	4.392E+00 −
F15	**−4.210E+02**	**8.520E+00**	−4.101E+02 −	1.060E+01 −	−3.800E+02 −	4.669E+01 −	−4.022E+02 −	3.392E+01 −	−3.900E+02 −	3.260E+01 −	−3.869E+02 −	4.962E+01 −
F16	6.208E−07	3.281E−07	1.569E+01 −	3.264E+00 −	2.561E−02 −	2.974E−02 −	1.647E−07 +	5.269E−07 −	3.647E−06 −	6.317E−06 −	**1.261E−12 +**	**1.358E−12 +**
F17	**4.632E−06**	**1.028E−06**	3.157E+01 −	1.214E+00 −	5.361E−02 −	1.526E−02 −	5.314E−04 −	2.317E−03 −	5.147E−01 −	2.361E−01 −	2.369E−05 −	1.118E−04 −
F18	**3.780E−04**	**4.485E−04**	2.358E+02 −	1.162E+01 −	4.765E+01 −	3.255E+00 −	5.660E+01 −	1.118E+01 −	1.225E+02 −	1.684E+01 −	2.548E−03 −	2.331E−03 −
F19	**2.841E−02**	**3.627E−02**	1.115E+01 −	1.884E+00 −	6.548E+00 −	6.118E−01 −	2.229E+00 −	8.621E−01 −	1.558E+00 −	1.192E+00 −	6.258E+00 −	1.864E+00 −
F20	**6.774E−04**	**2.418E−04**	4.054E+01 −	1.820E+00 −	1.584E−03 −	4.845E−04 −	2.228E−01 −	3.117E−01 −	6.489E+00 −	1.894E+00 −	5.668E−02 −	1.998E−02 −
F21	**3.940E−05**	**4.521E−05**	1.187E+01 −	3.022E+00 −	5.326E−03 −	1.286E−03 −	6.328E−03 −	1.326E−02 −	5.229E−03 −	6.185E−03 −	3.156E−03 −	1.228E−03 −
F22	**−4.610E+01**	**1.922E+00**	−2.369E+01 −	9.330E+00 −	−3.620E+01 −	1.320E+01 −	−3.949E+01 −	6.455E+00 −	−2.036E+01 −	1.609E+01 −	−4.442E+01 −	3.008E+00 −
F23	**−2.014E+04**	**5.142E+02**	−9.605E+03 −	2.601E+03 −	−1.086E+04 −	2.629E+03 −	−1.320E+04 −	8.110E+02 −	−1.123E+04 −	7.116E+02 −	−1.269E+04 −	7.668E+02 −
F24	**6.841E−04**	**7.469E−04**	5.620E+00 −	1.623E−01 −	5.116E+00 −	1.994E−01 −	1.562E+00 −	2.331E−01 −	5.323E−03 −	3.361E−03 −	5.260E+00 −	3.229E−01 −
F25	5.207E−03	5.217E−03	2.318E+01 −	2.331E+01 −	2.008E+01 −	2.451E+00 −	4.951E+00 −	3.119E+00 −	4.996E−02 −	2.119E−02 −	**0.000E+00 +**	**0.000E+00 +**
F26	**1.596E−02**	**1.433E−02**	1.336E+01 −	2.336E+00 −	2.117E+01 −	1.860E+00 −	6.479E+00 −	4.631E+00 −	2.394E+01 −	3.116E+01 −	1.010E+00 −	1.663E+00 −
F27	**2.941E−02**	**2.344E−02**	1.892E+01 −	3.215E+00 −	1.136E+00 −	1.001E−01 −	3.621E+00 −	4.256E−01 −	8.113E−01 −	1.015E−01 −	1.360E−01 −	6.258E−02 −
F28	7.468E−05	9.631E−06	6.325E+00 −	4.261E−01 −	3.618E−01 −	1.257E−01 −	4.116E−01 −	4.691E−01 −	2.118E−06 +	3.197E−06 +	**5.319E−10 +**	**2.351E−10 +**
F29	**2.141E+00**	**2.841E+00**	6.178E+05 −	6.332E+05 −	3.112E+00 −	3.260E+00 −	4.005E+01 −	5.339E+01 −	1.116E+03 −	1.008E+02 −	1.102E+03 −	1.087E+02 −
+			1	1	2	2	3	2	2	2	9	9
−			27	27	25	25	24	25	25	25	16	16
≈			1	1	2	2	2	2	2	2	4	4

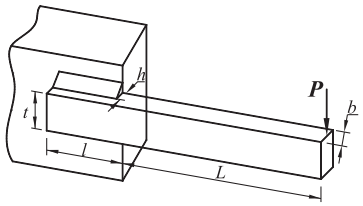

Fig. 11. Schematic diagram of welded beam design [46].

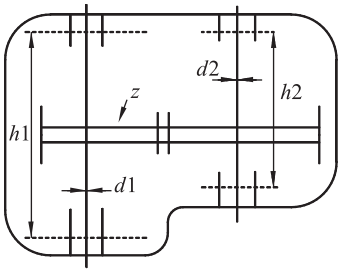

Fig. 12. Schematic diagram of speed reducer design [45].

The problem can be described mathematically as follows:

$$\text{Min } f(X) = x_1^2 x_2 (x_3 + 2) \tag{12}$$

subjected to $g_1(X) = 1 - \frac{x_2^3 x_3}{71782 x_1^4} \le 0$; $g_2(X) = \frac{4x_2^2 - x_1 x_2}{12566(x_1^3 x_2 - x_1^4)} + \frac{1}{5108 x_1^2} - 1 \le 0$; $g_3(X) = 1 - \frac{140.45 x_1}{x_2^2 x_3} \le 0$; $g_4(X) = \frac{x_1 + x_2}{1.5} - 1 \le 0$, where $0.05 \le x_1 \le 1$, $0.25 \le x_2 \le 1.3$, $2 \le x_3 \le 15$.

4.1.2. Welded beam design

The objective of this well-known problem is to design a welded beam with minimum total cost of fabricating. As shown in Fig. 11, the welded beam, designed to support a load and welded to a rigid support, is subjected to four design variables, which are the thickness of the weld h (x_1), the length of the welded joint l (x_2), the height of the beam t (x_3) and the thickness of the beam b (x_4), and the five constraints, which are the shear stress τ, the bending stress in the beam σ, the buckling load on the bar P_c, the deflection of the beam δ and side constraints [46].

The problem can be described mathematically as follows:

$$\text{Min } f(X) = (1 + C_1) x_1^2 x_2 + C_2 x_3 x_4 (L + x_2) \tag{13}$$

subjected to $g_1(X) = \tau(X) - \tau_{max} \le 0$; $g_2(X) = \sigma(X) - \sigma_{max} \le 0$; $g_3(X) = x_1 - x_4 \le 0$; $g_4(X) = \delta(X) - \delta_{max} \le 0$; $g_5(X) = P - P_c(X) \le 0$, where $\tau(X) = \sqrt{(\tau')^2 + 2\tau'\tau''\frac{x_2}{2R} + (\tau'')^2}$, $\tau' = \frac{P}{\sqrt{2}x_1 x_2}$, $\tau'' = \frac{MR}{J}$, $M = P(L + 0.5x_2)$, $R = \sqrt{0.25x_2^2 + 0.25(x_1 + x_3)^2}$, $J = 2\{\sqrt{2}x_1 x_2[\frac{x_2^2}{12} + 0.25(x_1 + x_3)^2]\}$, $\sigma(X) = \frac{6PL}{x_3^2 x_4}$, $P_c(X) = \frac{4.013E\sqrt{\frac{x_3^2 x_4^6}{36}}}{L^2}(1 - \frac{x_3}{2L}\sqrt{\frac{E}{4G}})$, $C_1 = 0.10471$ \$/in^3, $C_2 = 0.04811$ \$/in^3, $P = 6000$ lb, $L = 14$ in, $\delta_{max} = 0.25$ in, $E = 3.0 \times 10^7$ psi, $G = 1.2 \times 10^7$, $\tau_{max} = 1.36 \times 10^4$ psi, $\sigma_{max} = 3.0 \times 10^4$ psi, $0.125 \le x_1 \le 5$, $0.1 \le x_2 \le 10$, $0.1 \le x_3 \le 10$, $0.1 \le x_4 \le 5$.

4.1.3. Speed reducer design

The objective of this problem is to find a minimum total weight of the speed reducer. As shown in Fig. 12, the speed reducer is subjected to seven design variables, which are the face width b (x_1), the module of teeth m (x_2), the number of teeth on pinion z (x_3), the length of first shaft between bearings l_1(x_4), the length of second shaft between bearings l_2(x_5), the diameter of first shaft d_1(x_6) and the diameter of second shaft d_2(x_7), and four constrains, which are the limits on the bending stress of the gear teeth, the surface stress, the transverse deflections of shafts 1 and 2 due to transmitted force, and stresses in shafts 1 and 2 [45].

The problem can be described mathematically as follows:

$$\begin{aligned}\text{Min } f(X) = {} & 0.7854 x_1 x_2^2 (3.3333 x_3^2 + 14.9334 x_3 - 43.0934) \\ & - 1.508 x_1 (x_6^2 + x_7^2) + 7.4777 (x_6^3 + x_7^3) \\ & + 0.7854 (x_4 x_6^2 + x_5 x_7^2)\end{aligned} \tag{14}$$

subjected to $g_1(X) = \frac{27}{x_1 x_2^2 x_3} - 1 \le 0$; $g_2(X) = \frac{397.5}{x_1 x_2^2 x_3} - 1 \le 0$; $g_3(X) = \frac{1.93 x_4^3}{x_2 x_6^4 x_3} - 1 \le 0$; $g_4(X) = \frac{1.93 x_5^3}{x_2 x_7^4 x_3} - 1 \le 0$; $g_5(X) = \frac{1}{110 x_6^3}\sqrt{(\frac{745 x_4}{x_2 x_3})^2 + 16.9 \times 10^6} - 1 \le 0$, $g_6(X) = \frac{1}{85 x_7^3}\sqrt{(\frac{745 x_5}{x_2 x_3})^2 + 157.5 \times 10^6} - 1 \le 0$; $g_7(X) = \frac{x_2 x_3}{40} - 1 \le 0$; $g_8(X) = \frac{5x_2}{x_1} - 1 \le 0$; $g_9(X) = \frac{x_1}{12x_2} - 1 \le 0$; $g_{10}(X) = \frac{1.5x_6 + 1.9}{x_4} - 1 \le 0$, $g_{11}(X) = \frac{1.1x_7 + 1.9}{x_5} - 1 \le 0$, $2.6 \le x_1 \le 3.6$, $0.7 \le x_2 \le 0.8$, $17 \le x_3 \le 28$, $7.3 \le x_4 \le 8.3$, $7.3 \le x_5 \le 8.3$, $2.9 \le x_6 \le 3.9$, $5 \le x_7 \le 5.5$.

4.2. Results and analyses

IAFOA is compared with several intelligent algorithms mentioned in Sections 3.3.1 and 3.3.2 to solve the three engineering optimization problems. The parameters of these algorithms are the same as the previous. The statistical results obtained by IAFOA and other algorithms are shown in Table 6.

From Table 6, it can be clearly seen that IAFOA has a significant better performance than other comparative algorithms when solving the three engineering optimization problems. In detail, for engineering optimization problem coil compression spring design, although all the algorithms can achieve the same "Best", IAFOA can produce significantly better "Worst", "Mean" and "Std." than other algorithms. The best solution achieved by the proposed IAFOA is $x^* = (0.05165669, 0.35593916, 11.33499876)$ with $f(x^*) = 0.0126654861$. Meanwhile, the constraint values are $g(x^*) = (-2.04490856E{-}05, -5.73786773E{-}07, -4.05213877, -0.72826943)$, which illustrate that the solution is feasible.

Besides, for the second engineering optimization problem welded beam design, IAFOA is superior to all the comparative algorithms in terms of "Best", "Worst", "Mean" and "Std.". The best solution achieved by the proposed IAFOA is $x^* = (0.20572614, 3.47056150, 9.03663019, 0.20572961)$ with $f(x^*) = 1.7248564741$. What's more, the constraint values are $g(x^*) = (-6.21550498E{-}07, -3.74049439E{-}02, -3.46690000E{-}06, -0.23554035, -1.38686835E{-}04)$. However, for the third engineering optimization problem speed reducer design, IAFOA generates the best "Best" and "Mean", while GWO produces the best "Worst" and "Std.". The best solution achieved by IAFOA is $x^* = (3.5, 0.69999961, 17.00000600, 7.29477842, 7.79998996, 3.35021544, 5.28675679)$ with $f(x^*) = 2996.3478982241$.

Overall, IAFOA can efficiently solve the three engineering optimization problems and has the best performance compared with other ten algorithms.

5. Conclusion

In this paper, a new improved fruit fly optimization algorithm named IAFOA is designed to solve the high-dimensional function optimization problems and engineering optimization problems. In detail, we proposed four mechanisms including the adaptive selection mechanism for the search direction, the adaptive adjustment mechanism for the iteration step value, the adaptive crossover and mutation mechanism, and the multi-sub-swarm mechanism.

Table 6
Results of different algorithms on the three engineering optimization problems.

Algorithm	Coil compression spring design				Welded beam design				Speed reducer design			
	Best	Worst	Mean	Std.	Best	Worst	Mean	Std.	Best	Worst	Mean	Std.
*c-m*FOA	**0.012665**	0.012763	0.012710	2.145E−04	1.724880	1.725219	1.725114	2.963E−03	2996.356940	2999.745821	2997.236411	1.352E+00
MSFOS	**0.012665**	0.012784	0.012721	3.738E−04	1.724911	1.726155	1.725348	6.215E−03	2996.415711	3002.756491	2998.062837	6.354E+00
CMFOA	**0.012665**	0.013420	0.012931	1.454E−03	1.724866	1.724962	1.724922	7.883E−06	3000.981058	3009.261801	3007.242156	4.963E+00
IFOA	**0.012665**	0.012868	0.012765	9.287E−05	1.725224	1.727564	1.726517	2.405E−04	2996.348165	2996.348225	2996.348195	5.950E−05
IFFO	0.012666	0.013012	0.012851	1.185E−04	1.725717	1.727373	1.726393	3.497E−03	2996.348072	3094.556809	3016.492651	2.420E+01
IASFA	**0.012665**	0.012719	0.012687	2.452E−05	1.724866	1.759479	1.739654	8.064E−03	3014.605030	3076.723522	3057.704453	1.107E+00
CLPSO	**0.012665**	0.012698	0.012688	5.737E−06	1.724866	1.746361	1.736811	7.152E−03	2996.974415	3007.301217	3000.278329	3.804E+00
DLHS	**0.012665**	0.012900	0.012713	6.301E−05	1.724870	1.726047	1.725518	6.159E−04	2996.947261	3005.836268	3000.005428	2.356E+00
ALO	**0.012665**	0.012766	0.012702	2.612E−05	1.724911	1.725416	1.725120	1.562E−04	2996.348236	2996.348565	2996.348405	5.621E−05
GWO	**0.012665**	0.012721	0.012691	2.314E−05	1.724860	1.724871	1.724866	9.481E−06	2996.348164	**2996.348212**	2996.348185	**5.135E−06**
IAFOA	**0.012665**	**0.012688**	**0.012673**	**3.015E−06**	**1.724856**	**1.724856**	**1.724856**	**8.991E−07**	**2996.347898**	2996.348356	**2996.348069**	3.519E−05

After combining the four mechanisms with FOA, IAFOA is presented. Then the computational complexity of IAFOA is analyzed to estimate the computing time, and convergence analysis prove that IAFOA can converge to the global extreme with a probability of 100% theoretically. In the experimental section, five state-of-the-art variants of FOA and five advanced intelligent optimization algorithms are used to test the advantages of IAFOA based on the 29 benchmark functions. Compared with the several variants of FOA, which are *c-m*FOA, MSFOA, CMFOA, IFOA and IFFO, IAFOA generates 15/16/17/21/19 significantly better "Best", 18/20/22/25/26 significantly better "Worse", 18/19/23/24/27 significantly better "Mean", 18/20/23/25/27 significantly better "Std.", respectively. Besides, experiment results show that IAFOA can produce 27/25/24/25/16 significantly better, 1/2/3/2/9 significantly worse, 1/2/2/2/4 equal "Mean", and 27/25/25/25/16 significantly better, 1/2/2/2/9 significantly worse, 1/2/2/2/4 equal "Std." compared with IASFA/CLPSO/DLHA/ALO/GWO. Therefore, we can conclude that IAFOA has better performance than the five variants of FOA and five advanced intelligent optimization algorithms. Finally, IAFOA is used to solve three engineering optimization problems. Experiment results show that IAFOA can achieve the best "Best", "Worst", "Mean" and "Std." compared with the other 10 algorithms when solving coil compression spring design and welded beam design. Meanwhile, IAFOA can produce the best "Best" and "Std." for engineering optimization problem speed reducer design. The results prove that IAFOA has strong practicability for engineering optimization problems. In the future, we plan to apply IAFOA in solving other engineering optimization problems, such as layout problems which are widely used in the modern industry [47].

Acknowledgments

This work was supported in part by the project foundation of china Ministry of industry and information technology "Research of gordian technique of deep-water semi-submersible platforms", and Project of scientific and technological achievements of Jiangsu province "Research and industrialization of the key techniques of drilling string used in marine deep water".

References

[1] M. Akbari, H. Rashidi, S.H. Alizadeh, An enhanced genetic algorithm with new operators for task scheduling in heterogeneous computing systems, Eng. Appl. Artif. Intell. 61 (2017) 35–46.
[2] W.T. Pan, A new fruit fly optimization algorithm: taking the financial distress model as an example, Knowl. Based Syst. 26 (2012) 69–74.
[3] L. Wang, R. Liu, S. Liu, An effective and efficient fruit fly optimization algorithm with level probability policy and its applications, Knowl. Based Syst. 97 (2016) 158–174.
[4] Q.K. Pan, H.Y. Sang, J.H. Duan, L. Gao, An improved fruit fly optimization algorithm for continuous function optimization problems, Knowl. Based Syst. 62 (2014) 69–83.
[5] R.M. Rizk-Allah, R.A. El-Sehiemy, S. Deb, G.G. Wang, A novel fruit fly framework for multi-objective shape design of tubular linear synchronous motor, J. Supercomput. 73 (3) (2017) 1235–1256.
[6] J.Q. Li, Q.K. Pan, K. Mao, A hybrid fruit fly optimization algorithm for the realistic hybrid flowshop rescheduling problem in steelmaking systems, IEEE Trans. Autom. Sci. Eng. 13 (2) (2016) 932–949.
[7] Y.Y. Han, D.W. Gong, J.Q. Li, Y. Zhang, Solving the blocking flow shop scheduling problem with makespan using a modified fruit fly optimisation algorithm, Int. J. Produc. Res. 54 (22) (2016) 6782–6797.
[8] S.M. Lin, Analysis of service satisfaction in web auction logistics service using a combination of fruit fly optimization algorithm and general regression neural network, Neural Comput. Appl. 33 (2013) 783–791.
[9] L. Shen, H.L. Chen, Z. Yu, W.C. Kang, B.Y. Zhang, H.Z. Li, B. Yang, D.Y. Liu, Evolving support vector machines using fruit fly optimization for medical data classification, Knowl. Based Syst. 96 (2016) 61–75.
[10] H.Z. Li, S. Guo, C.J. Li, J.Q. Sun, A hybrid annual power load forecasting model based on generalized regression neural network with fruit fly optimization algorithm, Knowl. Based Syst. 37 (2013) 378–387.
[11] P.W. Chen, W.Y. Lin, T.H. Huang, W.T. Pan, Using fruit fly optimization algorithm optimized grey model neural network to perform satisfaction analysis for e-business service, Appl. Math. Inf. Sci. 7 (2) (2013) 459–465.

[12] Y. Yu, Y.C. Li, J.C. Li, X.Y. Gu, Self-adaptive step fruit fly algorithm optimized support vector regression model for dynamic response prediction of magnetorheological elastomer base isolator, Neurocomputing 211 (2016) 41–52.
[13] L. Wang, Y.L. Shi, S. Liu, An improved fruit fly optimization algorithm and its application to joint replenishment problems, Expert Syst. Appl. 42 (9) (2015) 4310–4323.
[14] X. Liu, Y. Shi, J. Xu, Parameters tuning approach for proportion integration differentiation controller of magnetorheological fluids brake based on improved fruit fly optimization algorithm, Symmetry 9 (2017) 109.
[15] C.L. Zuo, L.H. Wu, Z.F. Zeng, H.L. Wei, Stochastic fractal based multiobjective fruit fly optimization, Int. J. Appl. Math. Comput. Sci. 27 (2) (2017) 417–433.
[16] J. Xu, Z.B. Wang, C. Tan, L. Si, X.H. Liu, A novel denoising method for an acoustic-based system through empirical mode decomposition and an improved fruit fly optimization algorithm, Appl. Sci. 7 (3) (2017) 215.
[17] R. Hu, S.P. Wen, Z.G. Zeng, T.W. Huang, A short-term power load forecasting model based on the generalized regression neural network with decreasing step fruit fly optimization algorithm, Neurocomputing 221 (2017) 24–31.
[18] X.F. Yuan, X.S. Dai, J.Y. Zhao, Q. He, On a novel multi-swarm fruit fly optimization algorithm and its application, Appl. Math. Comput. 233 (2014) 260–271.
[19] L.H. Wu, C.L. Zuo, H.Q. Zhang, Z.H. Liu, Bimodal fruit fly optimization algorithm based on cloud model learning, Soft Comput. 21 (7) (2017) 1877–1893.
[20] T.F. Li, L. Gao, P.G. Li, Q.K. Pan, An ensemble fruit fly optimization algorithm for solving range image registration to improve quality inspection of free-form surface parts, Inf. Sci. 367 (2016) 953–974.
[21] D.X. Niu, T.N. Ma, B.Y. Liu, Power load forecasting by wavelet least squares support vector machine with improved fruit fly optimization algorithm, J. Combin. Optim. 33 (3) (2017) 1122–1143.
[22] Y. Zhang, G. Cui, J. Wu, W.T Pan, Q. He, A novel multi-scale cooperative mutation fruit fly optimization algorithm, Knowl. Based Syst. 114 (2016) 24–35.
[23] W.C. Wang, X.G. Liu, Melt index prediction by least squares support vector machines with an adaptive mutation fruit fly optimization algorithm, Chemom. Intell. Lab. Syst. 141 (2015) 79–87.
[24] F. Ye, X.Y. Lou, L.F. Sun, An improved chaotic fruit fly optimization based on a mutation strategy for simultaneous feature selection and parameter optimization for SVM and its applications, Plos One 12 (4) (2017) e0173516.
[25] W.T. Pan, Using modified fruit fly optimization algorithm to perform the function test and case studies, Connection Sci. 25 (2013) 151–160.
[26] L. Si, Z.B. Wang, X.H. Liu, C. Tan, Z. Liu, J. Xu, Identification of shearer cutting patterns using vibration signals based on a least squares support vector machine with an improved fruit fly optimization algorithm, Sensors 16 (1) (2016) 90.
[27] L.H. Wu, C.L. Zuo, H.Q. Zhang, A cloud model based fruit fly optimization algorithm, Knowl. Based Syst. 89 (2015) 603–617.
[28] S. Kanarachos, J. Griffin, M.E. Fitzpatrick, Efficient truss optimization using the contrast-based fruit fly optimization algorithm, Comput. Struct. 182 (2017) 137–148.
[29] J.W. Niu, W.M. Zhong, Y. Liang, N. Luo, F. Qian, Fruit fly optimization algorithm based on differential evolution and its application on gasification process operation optimization, Knowl. Based Syst. 88 (2015) 253–263.
[30] X.F. Yuan, Y.M. Liu, Y.Z. Xiang, X.G. Yan, Parameter identification of BIPT system using chaotic-enhanced fruit fly optimization algorithm, Appl. Math. Comput. 268 (2015) 1267–1281.
[31] X.L. Zheng, L. Wang, A two-stage adaptive fruit fly optimization algorithm for unrelated parallel machine scheduling problem with additional resource constraints, Expert Syst. Appl. 65 (2016) 28–39.
[32] X.J. Lei, Y.L. Ding, H. Fujita, A.D. Zhang, Identification of dynamic protein complexes based on fruit fly optimization algorithm, Knowl. Based Syst. 105 (2016) 270–277.
[33] T. Meng, Q.K Pan, An improved fruit fly optimization algorithm for solving the multidimensional knapsack problem, Appl. Soft Comput. 50 (2017) 79–93.
[34] L. Wu, W.S. Xiao, J.L. Wang, H.Q. Zhou, X. Tian, A new adaptive genetic algorithm and its application in the layout problem, Int. J. Comput. Intell. Syst. 8 (6) (2015) 1044–1052.
[35] L.M. Schmitt, Theory of Genetic Algorithms II: models for genetic operators over the string-tensorrepresentation of populations and convergence to global optima for arbitrary fitness function under scaling, Theor. Comput. Sci. 310 (2004) 181–231.
[36] S. Mirjalili, S.M. Mirjalili, A. Hatamlou, Multi-verse optimizer: a nature-inspired algorithm for global optimization, Neural Comput. Appl. 27 (2) (2016) 495–513.
[37] C.S. Li, N. Zhang, X.J. Lai, J.Z. Zhou, Y.H. Xu, Design of a fractional-order PID controller for a pumped storage unit using a gravitational search algorithm based on the Cauchy and Gaussian mutation, Inf. Sci. 396 (2017) 162–181.
[38] M. Jiang, D. Yuan, Y. Cheng, Improved artificial fish swarm algorithm, in: IEEE Fifth International Conference on Natural Computation (ICNC'09), 4, 2009, pp. 281–285.
[39] J.J. Liang, A.K. Qin, P.N. Suganthan, S. Baskar, Comprehensive learning particle swarm optimizer for global optimization of multimodal functions, IEEE Trans. Evol. Comput. 10 (3) (2006) 281–295.
[40] Q.K. Pan, P.N. Suganthan, J.J. Liang, M.F. Tasgetiren, A local-best harmony search algorithm with dynamic subpopulations, Eng. Optim. 42 (2) (2010) 101–117.
[41] S. Mirjalili, The ant lion optimizer, Adv. Eng. Software 83 (2015) 80–98.
[42] M.J. Wang, H.L. Chen, H.Z. Li, Z.N. Cai, X.H. Zhao, C.F. Tong, J. Li, X. Xu, Grey wolf optimization evolving kernel extreme learning machine: application to bankruptcy prediction, Eng. Appl. Artif. Intell. 63 (2017) 54–68.
[43] S. Mirjalili, S.M. Mirjalili, A. Lewis, Grey wolf optimizer, Adv. Eng. Software 69 (2014) 46–61.
[44] L. Wu, X. Tian, J.X. Zhang, Q. Liu, W.S. Xiao, Y.W. Yang, An improved heuristic algorithm for 2D rectangle packing area minimization problems with central rectangles, Eng. Appl. Artif. Intell. 66 (2017) 1–16.
[45] X. Meng, X.Z. Gao, Y. Liu, H. Zhang, A novel bat algorithm with habitat selection and Doppler effect in echoes for optimization, Expert Syst. Appl. 42 (2015) 6350–6364.
[46] A.H. Gandomi, X.S. Yang, A.H. Alavi, S. Talatahari, Bat algorithm for constrained optimization tasks, Neural Comput. & Applic. 22 (6) (2013) 1239–1255.
[47] L. Wu, L. Zhang, W.-S. Xiao, Q. Liu, C. Mu, Y.W. Yang, A novel heuristic algorithm for two-dimensional rectangle packing area minimization problem with central rectangle, Comput. Ind. Eng. 102 (2016) 208–218.