

## A new swarm intelligent optimization algorithm: Pigeon Colony Algorithm (PCA)

Ting-Hua Yi<sup>\*1</sup>, Kai-Fang Wen<sup>1a</sup> and Hong-Nan Li<sup>1,2b</sup>

<sup>1</sup>*School of Civil Engineering, Dalian University of Technology, Dalian 116023, China*

<sup>2</sup>*School of Civil Engineering, Shenyang Jianzhu University, Shenyang 110168, China*

*(Received October 15, 2015, Revised January 17, 2016, Accepted March 5, 2016)*

**Abstract.** In this paper, a new Pigeon Colony Algorithm (PCA) based on the features of a pigeon colony flying is proposed for solving global numerical optimization problems. The algorithm mainly consists of the take-off process, flying process and homing process, in which the take-off process is employed to homogenize the initial values and look for the direction of the optimal solution; the flying process is designed to search for the local and global optimum and improve the global worst solution; and the homing process aims to avoid having the algorithm fall into a local optimum. The impact of parameters on the PCA solution quality is investigated in detail. There are low-dimensional functions, high-dimensional functions and systems of nonlinear equations that are used to test the global optimization ability of the PCA. Finally, comparative experiments between the PCA, standard genetic algorithm and particle swarm optimization were performed. The results showed that PCA has the best global convergence, smallest cycle indexes, and strongest stability when solving high-dimensional, multi-peak and complicated problems.

**Keywords:** optimization algorithm; Pigeon Colony Algorithm; low-dimensional function; high-dimensional function; nonlinear equation

### 1. Introduction

In conjunction with the rapid development of production, optimization problems have emerged widely in industry, communication, transportation, civil and hydraulic engineering (Li *et al.* 2012, 2015), and in many other fields. For example, Genetic Algorithms (GAs) have been used in vehicle scheduling (Zuo *et al.* 2015), and the Monkey Algorithm (MA) has been adopted for sensor placements in a structural health monitoring field (Yi *et al.* 2012). Traditional optimization algorithms such as Newton's method (Qi and Sun 1993), Simplex algorithm (Spielman and Teng 2004), and the Branch-and-bound method (Zuo *et al.* 2015) have been used in many fields. However, these approaches demonstrate some limitations in nonlinear, complex situations that have multiple peak functions (Lei *et al.* 2012, 2013).

In recent years, swarm intelligent optimization algorithms have become popular because they

---

\*Corresponding author, Professor, E-mail: [yth@dlut.edu.cn](mailto:yth@dlut.edu.cn)

<sup>a</sup> MSc Student, Email: [wenkaifang@mail.dlut.edu.cn](mailto:wenkaifang@mail.dlut.edu.cn)

<sup>b</sup> Professor, E-mail: [hnli@dlut.edu.cn](mailto:hnli@dlut.edu.cn)

can solve combinatorial optimization problems well, and with them, it is not easy to fall into locally optimal solutions. The first swarm intelligent optimization algorithm was the GA (Holland 1975), which is based on natural selection and the biological evolution of a genetics mechanism; this approach is independent from the concrete aspects of the fields of application and is broadly used in many fields, such as combination optimization, mathematical problems and signal processing. Dorigo *et al.* (1991) simulated the process of ants foraging and proposed Ant Colony Optimization, which can well solve optimal path selection problems, such as the traveling salesman problem. Kennedy and Eberhart (1995, 1998) proposed Particle Swarm Optimization (PSO) through the simulation of birds flying, which has been extensively used in function optimization, neural network training, and fuzzy system control. Harmony Search (Geem *et al.* 2001) was proposed in terms of musicians composing wonderful and sweet harmonies. The algorithm was successfully applied to many combinatorial optimization domains. The Artificial Fish-School Algorithm (Li and Qian 2003) was proposed through the study of fish foraging behavior. It has been practiced in power system planning and the optimization of multi-step logistic transit shipment systems. Eusuff and Lansey (2003) proposed the theory of the Shuffled Frog Leaping Algorithm by simulating a frog foraging, and this approach has been used to solve the problem of minimizing the pipeline size in a pipeline network expansion. Zhao and Tang (2008) proposed the MA, which is based on the climb, watch and jump process of a monkey. This approach can effectively solve the optimization problem of high-dimensional, non-linear and non-differentiable functions. Yang (2009) used inspiration from the group behavior of firefly information exchange via fluorescence and proposed the Firefly Algorithm, which has good performance in peak function optimization. Yan *et al.* (2010) proposed the Wolves Algorithm, which is based on the predator and prey distribution of wolves. It can better handle complex multimodal, high-dimensional functions while avoiding the premature convergence of a general intelligence algorithm.

The intelligent optimization algorithms described above have played effective roles in their own specific fields, while most have premature convergence, large cycle indexes and slow convergence and easily fall into local optimums for high-dimensional, multi-peak and complicated functions. This paper proposes a new swarm intelligence optimization algorithm, the Pigeon Colony Algorithm (PCA), to solve global numerical optimization problems. PCA has good global convergence, small cycle indexes, and strong stability when finding optimal solutions in high-dimensional, multi-peak and complicated problems. This paper is organized as follows: Section 2 describes the concept and detailed implementation steps of PCA. Section 3 demonstrates the impact of the parameters on the PCA solution quality. In Section 4, low-dimensional functions, high-dimensional functions and systems of nonlinear equations are used to test the global optimization ability of the PCA. Section 5 shows comparative experiments between PCA, standard GA and PSO. Finally, conclusions are drawn in Section 6.

## 2. Pigeon colony algorithm

### 2.1 The PCA architecture

The pigeon is a type of common social animal that is characterized by having a strong homing ability, high sensitivity, and good memory, among other traits. This paper refines the characteristics of the pigeon colony and further proposes PCA. PCA includes three processes:

take-off, flying and homing. 1) Take-off process: to simulate the take-off characteristics of the pigeon colony, including initialize, spring up and ascend, which constitute three sub-processes, to homogenize the initial values and look for the direction of the optimal solution; 2) flying process: to simulate the flying characteristics of the pigeon colony, including the level fly, turn, and chase, which constitute three sub-processes; the level fly is used to search for a local optimum, and turning is used to search for a global optimum, while chasing is used to improve the global worst solution; 3) homing process: according to the strong homing characteristics of the pigeon colony, the homing process avoids having the algorithm fall into a local optimum. The flowchart of PCA is shown in Fig. 1.

## 2.2 Take-off process

### 2.2.1 Initialize

$N$  is defined as the size of the pigeon population; the vector  $X_i = (x_{i1}, x_{i2}, \dots, x_{ij}, \dots, x_{in})$ ,  $i = 1, 2, \dots, N$ ,  $j = 1, 2, \dots, n$ , is each pigeon's current position; and  $n$  is the number of unknown numbers, namely, the dimension. Each pigeon's current position vector  $X_i$  corresponds to a feasible solution of one optimization problem, and it has the same dimension  $n$ ; the vector  $Y_i = (y_{i1}, y_{i2}, \dots, y_{ij}, \dots, y_{in})$  is pigeon  $i$ 's current optimal position; the vector  $P_b = (p_{b1}, p_{b2}, \dots, p_{bn})$  is the pigeon colony's current best position; and the vector  $P_w = (p_{w1}, p_{w2}, \dots, p_{wn})$  is the pigeon colony's current worst position.

Step 1: Initialize the pigeon colony's position.

For a multi-dimensional function, the variable  $x_{ij}$  has a domain of definition  $x_{ij} \in [x_{down}, x_{up}]$ . For the order that is followed when the pigeon colony takes off, within the range of the domain, each pigeon's initial domain decreases as in Eq. (1), which makes each pigeon's domain different.

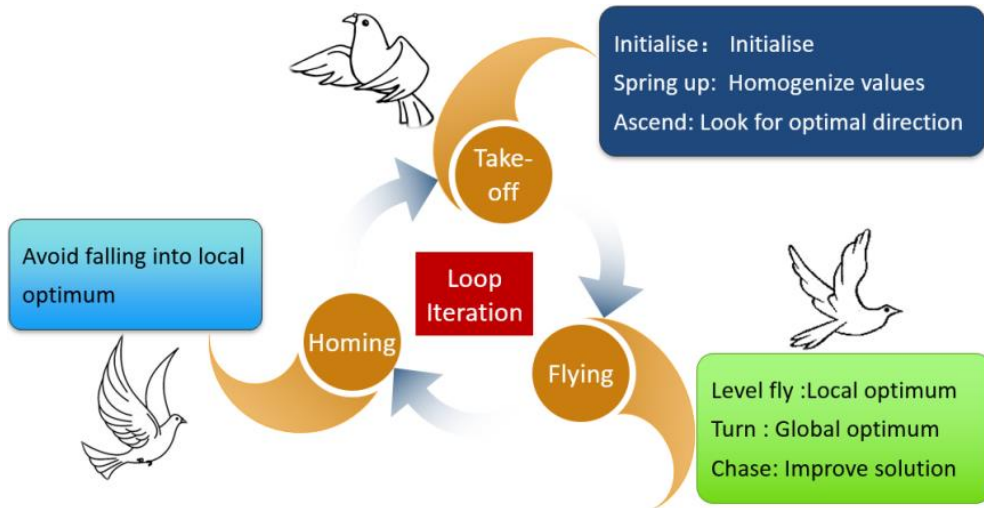


Fig. 1 Flowchart of the processes of the PCA

Define vector  $\lambda = \left( \frac{1}{N+1}, \frac{2}{N+1}, \dots, \frac{N}{N+1} \right)$  and disrupt the order of the component in  $\lambda$  to obtain the vector  $\lambda'$ . It will enrich the initial values of the solution.

$$x_{ij} = \lambda'_i (x_{up} - x_{down}) + x_{down} \quad (1)$$

Step 2: Initialize the pigeon colony's sensitivity.

Pigeons are characterized by quick acuteness and high alertness, and they are easily frightened. Each pigeon's sensitivity is different from the others. Define  $\alpha_i$  as the sensitivity coefficient of pigeon  $i$ , where  $\alpha_i$  is a random number that is generated from the range  $[0,1]$ . The larger the value of  $\alpha_i$  is, the higher the pigeon springs up, and the more disperse the initial values.

Step 3: Initialize the pigeon colony's speed.

Define the vector  $V_i = (v_{i1}, v_{i2}, \dots, v_{ij}, \dots, v_{in})$  as pigeon  $i$ 's flying speed, which means a positive direction when the  $v_{ij}$  is positive, and vice versa. Define  $[-V_{\max}, V_{\max}]$  as the speed range; let  $v_{ij}$  be a uniformly generated random number from the range of  $[-V_{\max}, V_{\max}]$ , and its expression is

$$v_{ij} = \delta V_{\max} \quad (2)$$

In Eq. (2),  $\delta$  is a uniformly generated random number in the range  $[-1,1]$ .

### 2.2.2 Spring up

When the pigeon colony takes off, the jump up height of each pigeon is different. According to this characteristic, homogenize the initial values, and define  $[down, up]$  as the pigeon colony's spring up range.

Step 1: Define  $\Delta X_i = (\Delta x_{i1}, \Delta x_{i2}, \dots, \Delta x_{ij}, \dots, \Delta x_{in})$  as pigeon  $i$ 's jump up height, where  $\Delta x_{ij}$  is a uniformly generated random number from the range  $[down, up]$ ; its expression is

$$\Delta x_{ij} = \varepsilon (up - down) + down \quad (3)$$

where  $\varepsilon$  is a random number generated in the range  $[0,1]$ .

Step 2: Update each pigeon's current position  $X_i$  using the expression

$$X_i = Y_i + \alpha_i * \Delta X_i \quad (4)$$

If  $X_i$  is better than the current optimal position  $Y_i$ , then the current position  $X_i$  is assigned to the current optimal position  $Y_i$ ; in other words,  $Y_i = X_i$ . If  $X_i$  is better than the pigeon colony's current best position  $P_b$ , then make  $P_b = X_i$ .

Remark 1: To improve the precision and speed of the PCA, the spring up range  $[down, up]$  has the same precision as the maximum item  $p_{ij}$  of  $P_b$ . For example, when the precision of the

maximum item  $p_{ij}$  of  $P_b$  is one percent, then the spring up range  $[down, up]$  maintains the same precision in the following way

$$\begin{cases} up_{new} = up * 0.01 \\ down_{new} = down * 0.01 \end{cases} \quad (5)$$

### 2.2.3 Ascend

The pigeon colony has an ascend process after the spring up, which makes it fly in a better direction. Simulating this property, using the Pseudo Gradient Method, define  $f'_{ij}(X_i)$  as the direction of the optimal solution.

Step 1: Randomly generate vector  $\Delta C_i = (\Delta c_{i1}, \Delta c_{i2}, \dots, \Delta c_{ij}, \dots, \Delta c_{in})$  by Eq. (6).

$$\Delta c_{ij} = \begin{cases} ri & \text{with probability } \frac{1}{2} \\ -ri & \text{with probability } \frac{1}{2} \end{cases} \quad (6)$$

where  $ri$  is called the up height.

Step 2: Calculate pigeon  $i$ 's up direction  $f'_{ij}(X_i)$  for each dimension  $j$  using the following expression

$$f'_{ij}(X_i) = \frac{f(X_i + \Delta C_i) - f(X_i - \Delta C_i)}{2\Delta c_{ij}}, \quad i=1,2,\dots,N, \quad j=1,2,\dots,n. \quad (7)$$

Step 3: Update each pigeon's current position  $X_i$  using the following expression

$$x_{ij} = y_{ij} + ri * \text{sign}(f'_{ij}(X_i)) \quad (8)$$

where  $\text{sign}(x)$  is the symbol function. When  $x \geq 0$ ,  $\text{sign}(x)=1$ . When  $x < 0$ ,  $\text{sign}(x)=-1$ . If  $X_i$  is better than the current optimal position  $Y_i$ , then make  $Y_i = X_i$ ; if  $X_i$  is better than the pigeon colony's current optimal position  $P_b$ , then make  $P_b = X_i$ .

Step 4: Cycle through steps 1 to 3 once more.

Remark 2: To avoid the random vector  $\Delta C_i$  having a large deviation that can affect the accuracy of the up direction  $f'_{ij}(X_i)$ , more time is needed for the ascending process in Step 4. To improve the precision and speed of the PCA, the up height  $ri$  has one more level of precision than the spring up range  $[down, up]$  when the range  $[down, up]$  is changed. For example, when the precision of the spring up range  $[down, up]$  is changed to one percent,  $ri$  is also changed:  $ri_{new} = ri * 0.001$ .

## 2.3 Flying process

After the pigeon colony's take-off, the colony has entered the phase of flying, and Zhang's (2014) research shows that when the pigeon colony flies at a flat height, the direction of each pigeon follows its neighbors. When in a turn, the direction is to obey the leadership. Using this feature, a local optimum can be found when the pigeon colony flies in a level flying region, and a global optimum will be obtained when they turn.

### 2.3.1 Level fly

Define pigeon  $i$ 's range of neighbors as  $M$ , e.g., in the range of one pigeon, there are  $M$  pigeons around as its neighbors. For each pigeon, the average position of the neighbors is defined as  $Ave_i$ . The time of the level fly is  $F_1$ .

Step 1: Calculate, for pigeon  $i$ , the average position of its neighbors  $Ave_i$  using the expression

$$Ave_i = \frac{\sum_{i=i-\lfloor M/2 \rfloor}^{M-(i-\lfloor M/2 \rfloor)+1} y_i}{M} \quad (9)$$

where  $M$  is the key parameter of the flying process, which will affect the rate of convergence of the local optimization. When  $M$  is too large, the value of  $Ave_i$  will be similar to the global optimization, which will decrease the speed of convergence. Here,  $\lfloor \bullet \rfloor$  means down to the nearest integer. When  $M$  is too small, the algorithm will have premature convergence, which will decrease the convergence accuracy.

Step 2: Use  $Ave_i$  to calculate pigeon  $i$ 's flying speed.

$$V_i = w * V_i + c_1 * (Ave_i - X_i) \quad (10)$$

where  $c_1$  is the local flying factor, and  $w$  is the flying weight coefficient, and using Eberhart's (1999) regressive method from 0.9 to 0.4, its expression is

$$w = 0.9 - \frac{0.5}{M_c - 1} * (cn - 1) \quad (11)$$

where  $M_c$  is the total number of cycle indexes, and  $cn$  is the current cycle index.

Step 3: Update each pigeon's current location using the expression

$$X_i^{r+1} = X_i^r + V_i \quad (12)$$

If  $X_i^{r+1}$  is better than the current optimal position  $Y_i$ , then make  $Y_i = X_i^{r+1}$ ; if  $X_i$  is better than the pigeon colony's current optimal position  $P_b$ , then make  $P_b = X_i^{r+1}$ .

Step 4: Repeat step 1 to step 3, until they reach the level fly cycles  $F_1$ .

### 2.3.2 Turn

A pigeon colony often turns when it is flying, to make the whole colony maintain a better direction. Define the turn times as  $F_2$ .

Step 1: Calculate pigeon  $i$ 's flying speed as  $V_i$

$$V_i = c_2 * (P_b - Y_i) \quad (13)$$

where  $c_2$  is a global flying factor.

Step 2: Update each pigeon's current location, the same as in Eq. (12).

If  $X_i^{r+1}$  is better than the current optimal position  $Y_i$ , then make  $Y_i = X_i^{r+1}$ ; if  $X_i$  is better than the pigeon colony's current optimal position  $P_b$ , then make  $P_b = X_i^{r+1}$ .

Step 3: Repeat step 1 to step 2, until the turning cycle  $F_2$  is reached.

### 2.3.3 Chase

Compared with other birds, pigeons are "monogamous" birds. After the female pigeons fly out of the nest, male pigeons will have a "chasing wife" behavior, called chase. Set the optimal location  $P_b$  of a pigeon colony to be occupied by a female and its worst position  $P_w$  to be occupied by a male, and make them match to improve the worst global solution.

Step 1: Among the  $n$ -dimensional space vectors, from  $[n/2] \sim n$  dimensions, a uniformly generated random integer bit is used as a substitute for the location points  $cp$

$$cp = [n/2] + [\phi(n/2)] \quad (14)$$

where  $\phi$  is a uniformly generated random number from the range  $[0,1]$ .

Step 2: Copy the value from the position  $cp \sim n$  dimensional of  $P_b = (p_{b1}, p_{b2}, \dots, p_{bcp}, \dots, p_{bn})$  directly to its corresponding positions of  $P_w = (p_{w1}, p_{w2}, \dots, p_{wcp}, \dots, p_{wn})$ . After the update, if a group's worst position  $P_w$  is better than the previous worst location, then keep the update; otherwise, it is not updated.

This process will improve the global worst solution, which will improve the convergence speed significantly.

### 2.4 Homing process

Pigeons possess a homing ability (Perera *et al.* 1999), and after the flying process, they will always return to their own nests. Define  $[-rg, rg]$  as the homing range; the smaller the range of homing is, the smaller the final landing area, and vice versa. The memory of each pigeon is different, and thus, the average position difference  $\Delta H_i$  is introduced to prevent having a large deviation among the individual pigeons when landing.

Step 1: For pigeon  $i$ , the homing coefficient  $r_i$  is a uniformly generated random number from the range  $[-rg, rg]$ .

Step 2: Based on the optimal position  $Y_i$  of each pigeon, calculate the gap in the average position between the individual position and that of the other pigeons.

$$\Delta H_i = r_i * \left( \left( \sum_{i=1}^N Y_i - Y_i \right) / (N-1) - Y_i \right) \quad (15)$$

$\Delta H_i$  is very important in PCA, which will avoid having the algorithm fall into a local optimal solution.

Step 3: Update pigeon  $i$ 's current position.

$$X_i = Y_i + \Delta H_i \quad (16)$$

If  $X_i$  is better than the current optimal position  $Y_i$ , then make  $Y_i = X_i$ ; if  $X_i$  is better than the pigeon colony's current optimal position  $P_b$ , then make  $P_b = X_i$ .

A complete PCA procedure contains take-off, flying, and homing, and these three processes are called one iteration. This routine is iterated until the global optimal solution is found or until the termination conditions are met. The advantages of PCA include that it has high convergence accuracy as well as a high convergence speed. The take-off process and homing process can improve the convergence accuracy by the changing of  $[down, up]$ ,  $ri$  and  $\Delta H_i$ . The flying process improves the convergence speed by  $M$  and the chase process. The pseudo code of the process of PCA is shown in Fig. 2.

### 3. Parameter analyses

Similar to many intelligent optimization algorithms, PCA also has several controllable parameters. PCA has ten controllable parameters in total, except that  $N$  is the pigeon's population ( $N$  is fixed in each comparison experiment, and  $N = 60$  is chosen in all of the later experiments); the parameters are divided into two parts, in groups of three. The first part has parameters that are in the flying process, which include  $(c_1, c_2, M)$  and  $(F_1, F_2, V_{\max})$ . The former are the parameters of the functions. The latter are the vector and iteration numbers of the flying process. The second part has parameters that are in the other process, which contains  $([down, up], ri, [-rg, rg])$ . Analyzing the variable parameters of PCA is necessary and requires having a selection range for each parameter. The Rastrigin function is chosen to test the optimization performance of the intelligent optimization algorithm.

$$F(X) = \sum_{i=1}^n [x_i^2 - 10 \cos(2\pi x_i) + 10] \quad (17)$$

This function is based on the function Sphere. It uses the cosine function to generate a large number of local minima. The global optimal solution is zero in  $(x_1, x_2, \dots, x_n) = (0, 0, \dots, 0)$ . The Rastrigin function is a typical complex multimodal function with a large number of locally optimal points; this function makes the algorithm fall into a local optimum easily, and then, it cannot obtain the global optimal solution. The 3D surface of the Rastrigin function when the dimension  $n = 2$  is shown in Fig. 3.

Each group of experiment data was run 100 times in the experiments in this paper. The control variable method was used in the experiments to keep the other parameters fixed when analyzing



one group of parameters. The average number of total cycle indexes and the success rate is calculated in 100 experiments of PCA. An iteration of PCA has three processes: take-off, flying and homing. Each process has its cycles, and thus, the total number of cycle indexes means all of the cycles of all of the iterations of PCA. The termination condition is when the change in the value  $P_b$  is less than  $10^{-6}$  for 10 iterations. Success means that the difference between the PCA minimum value  $P_b$  and the real minimum value of the Rastrigin is less than  $10^{-5}$ ; otherwise, there is a failure. The success rate means the success proportion out of a total of 100 times. PCA has three processes, and it uses cycle indexes to calculate the cost of the whole process.

```

Pigeon Colony Algorithm
Set population of PCA N
Initialize PCA
Set parameters  $[down, up]$ ,  $ri$ ,  $c_1$ ,  $c_2$ ,  $M$ ,  $V_{max}$ ,  $F_1$ ,  $F_2$  and  $[-rg, rg]$ 
Set termination condition
for s = 1 to termination condition do:
{
  Initialise  $X_i = (x_{i1}, x_{i2}, \dots, x_{ij}, \dots, x_{in})$ ,  $x_{ij} \in [x_{down}, x_{up}]$   $x_{ij} = \lambda_i'(x_{up} - x_{down}) + x_{down}$ 
  Calculate jump up height  $\Delta X_i = (\Delta x_{i1}, \Delta x_{i2}, \dots, \Delta x_{ij}, \dots, \Delta x_{in})$ ,
   $\Delta x_{ij} = \varepsilon(up - down) + down$ 
  Update  $X_i = Y_i + \alpha_i * \Delta X_i$ 
  For i=1 to 2 do:
  {
    Calculate  $\Delta C_i = (\Delta c_{i1}, \Delta c_{i2}, \dots, \Delta c_{ij}, \dots, \Delta c_{in})$ 
    Calculate  $f'_{ij}(X_i) = \frac{f(X_i + \Delta C_i) - f(X_i - \Delta C_i)}{2\Delta c_{ij}}$ 
    Update  $x_{ij} = y_{ij} + ri * sign(f'_{ij}(X_i))$ 
  }
  For i = 1 to  $F_1$  do:
  {
    Calculate  $Ave_i = \frac{\sum_{i=1}^{M-(i-[M/2])+1} y_i}{M}$ ,
    Calculate  $V_i = w * V_i + c_1 * (Ave_i - X_i)$ 
    Update  $X_i^{r+1} = X_i^r + V_i$ 
  }
  For i=1 to  $F_2$  do:
  {
    Calculate  $V_i = c_2 * (P_b - Y_i)$ , Update  $X_i^{r+1}$ 
  }
  Calculate  $cp = [n/2] + [\phi(n/2)]$  copying the value from  $p_{gcp} \sim p_{gn}$  of
   $P_b = (p_{b1}, p_{b2}, \dots, p_{bcp}, \dots, p_{bm})$  directly to its corresponding positions in
   $P_w = (p_{w1}, p_{w2}, \dots, p_{wcp}, \dots, p_{wn})$ 
  Calculate  $\Delta H_i = r_i * \left( \left( \sum_{i=1}^N Y_i - Y_i \right) / (N-1) - Y_i \right)$ 
  Update  $X_i = Y_i + \Delta H_i$ 
}
Print Optimal Solutions

```

Fig. 2 Pseudo code of the process of PCA

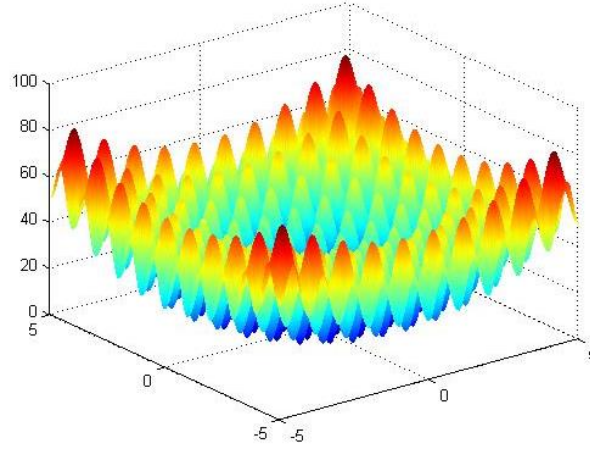


Fig. 3 The 3D surface of the Rastrigin function when the dimension

The calculation cost of PCA means the cycle times. In one iteration, the takeoff process contains two ascending cycles. The flying process contains  $F_1$  level fly cycles,  $F_2$  turn cycles and once chase cycle. It will have many iteration times when the PCA reaches the termination condition; the cycle indexes are the sum of the cycles of all of the iterations, which can reflect the cost of the algorithm. The average of the cycle indexes is calculated in 100 experiments using PCA.

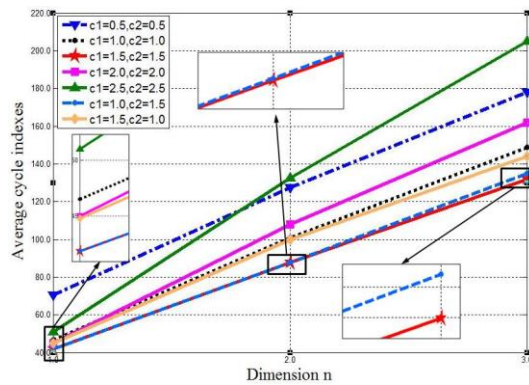
### 3.1 The influence of $c_1, c_2, M$

The average cycle indexes of PCA in a different combination of  $c_1, c_2, M$  when  $n=1,2,3$  are shown in Fig. 4. First, set the other six parameters to be fixed and  $M=5$ , when comparing the different combinations of  $c_1, c_2, M$ . Fig. 4(a) shows that the average cycle indexes of PCA are smaller when  $c_1, c_2$  is set to be from 1 to 1.5, and the average cycle indexes are smallest with  $c_1 = c_2 = 1.5$  when  $n=1,2,3$ . Here,  $c_1 = c_2 = 1.5$  has the best performance, while  $c_1 = c_2 = 2.5$  has the worst. Then, set  $c_1 = c_2 = 1.5$  to compare the different values of  $M$  (Fig. 4(b)). The results show that  $M=3$  and  $M=5$  have better performances.  $M=5$  becomes better while  $n$  increases, gradually. Last, to validate if  $c_1 = c_2 = 1.5$  and  $M=5$  will have an even better performance, set  $n=4,5,6$ . The results in Table 1 show that  $c_1 = c_2 = 1.5$  and  $M=5$  have better performance when the dimension is increasing and that  $M=3$  is also a good choice.

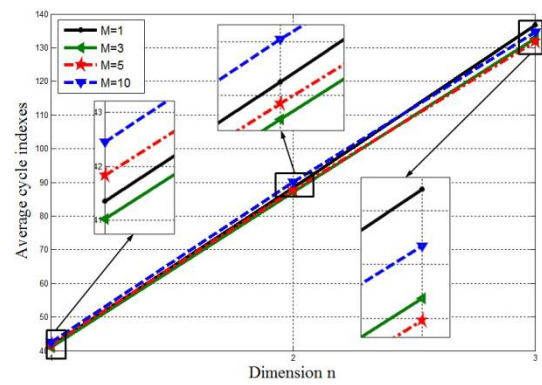
### 3.2 The influence of $F_1, F_2, V_{\max}$

First, set the other six parameters to be fixed, and set  $V_{\max} = 1$ . Fig. 5(a) shows that the average total cycles of the PCA is the smallest with  $F_1 = F_2 = 5$  when  $n=1,2,3$ . Thus,  $F_1 = F_2 = 5$  has

the best performance. The performance is also good when  $F_2$  rises to 10 and  $F_1 = 5$ . Thus,  $F_2$  can be chosen to be from 5 to 10. Then, set  $F_1 = F_2 = 5$  to compare the different values of  $V_{\max}$  (Fig. 5(b)). The results show that when  $V_{\max} = 1$ , there is better performance. Last, set  $n = 4, 5, 6$  to see if  $F_1 = F_2 = 5$  and  $V_{\max} = 1$  will still have better performance. The results in Table 2 show that  $F_1 = F_2 = 5$  and  $V_{\max} = 1$  have better performance when the dimension is increasing.

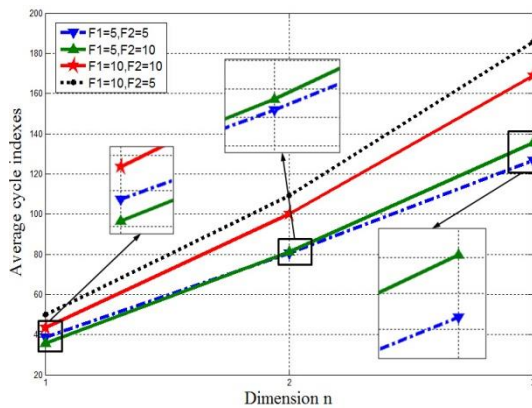


(a) The different combinations of  $c_1, c_2$  when  $M = 5$  and  $n = 1, 2, 3$

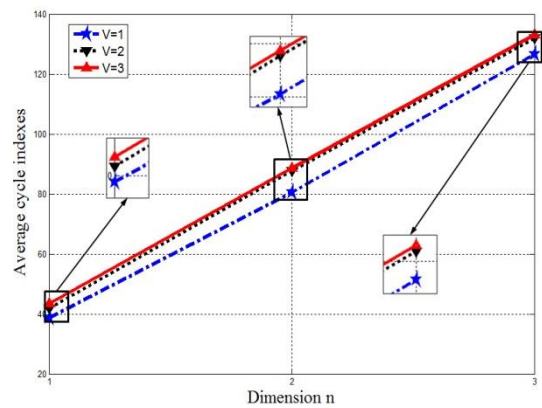


(b) The different values of  $M$  when  $c_1 = c_2 = 1.5$  and  $n = 1, 2, 3$

Fig. 4 The different combinations of  $c_1, c_2, M$  when  $n = 1, 2, 3$



(a) The different combinations of  $F_1, F_2$  when  $V_{\max} = 1$  and  $n = 1, 2, 3$



(b) The different values of  $V_{\max}$  when  $F_1 = 5, F_2 = 5$  and  $n = 1, 2, 3$

Fig. 5 The different combinations of  $F_1, F_2, V_{\max}$  when  $n = 1, 2, 3$

Table 1 Regular verification when  $n = 4, 5, 6$ 

Dimension $n$	Value of $c_1$	Value of $c_2$	Value of $M$	Average cycle indexes	Success rate
4	1.5	1.5	3	171.45	100%
4	1.5	1.5	5	170.325	100%
4	1	1.5	5	182.325	100%
5	1.5	1.5	3	204.675	100%
5	1.5	1.5	5	203.1	100%
5	1	1.5	5	219.525	100%
6	1.5	1.5	3	232.275	100%
6	1.5	1.5	5	232.125	100%
6	1	1.5	5	256.425	100%

Table 2 The different combinations of  $F_1, F_2, V_{\max}$  when  $n = 1 \sim 6$ 

Dimension $n$	Value of $F_1$	Value of $F_2$	Value of $V_{\max}$	Average iterations	Average cycle indexes	Success rate
4	5	5	1	11.14	167.1	100%
4	5	5	2	11.355	170.325	100%
5	5	5	1	13.295	199.425	100%
5	5	5	2	13.539	203.1	100%
6	5	5	1	15.164	227.475	100%
6	5	5	2	15.475	232.125	100%

### 3.3 The influence of $[down, up]$ , $ri$ and $[-rg, rg]$

First, set the other six parameters be fixed, and set  $ri = 0.1$ . Fig. 6(a) shows that when the spring up range  $[down, up]$  increases, the average total cycle indexes of PCA increase, when the homing range  $[-rg, rg]$  is fixed in each dimension. When the homing range  $[-rg, rg]$  increases, the average total cycle indexes of PCA also increase, when the spring up range  $[down, up]$  is fixed in each dimension. The best choices for the spring up range and homing range are  $[down, up] = [-1, 1]$  and  $[-rg, rg] = [-1, 1]$ . When  $[down, up] = [-1, 1]$  and the homing range

$[-rg, rg]$  rises to  $[-10, 10]$ , the performance is also good. Then, set  $[down, up] = [-1, 1]$  and  $[-rg, rg] = [-1, 1]$ , and compare the different values of  $ri$  (Fig. 6(b)). The results show that  $ri = 0.1$  and  $ri = 0.01$  both perform well. Last, set  $n = 4, 5, 6$  to see if  $[down, up] = [-1, 1]$ ,  $[-rg, rg] = [-1, 1]$  and  $ri = 0.1$  or  $ri = 0.01$  will still have better performance. The results in Table 3 show that the regulars above are correct while the dimensions are increasing.

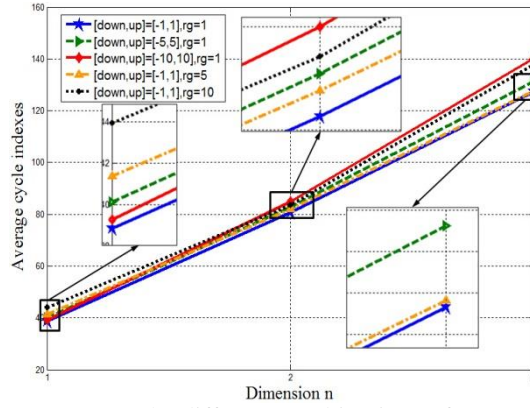
The reference selection of ten controllable parameters in PCA is shown in Table 4.

Table 3 The different combinations of  $[down, up], rise, [-range, range]$  when  $n = 1 \sim 6$

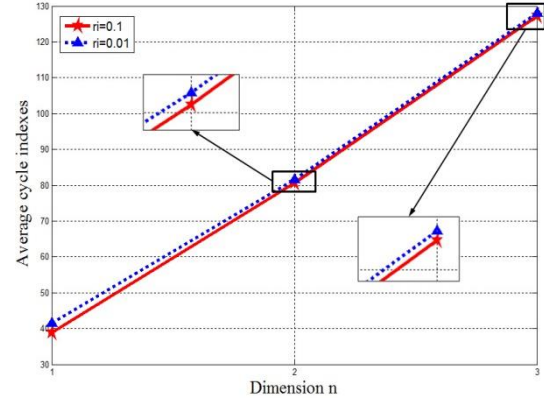
Dimension $n$	Value of $[down, up]$	Value of $ri$	Value of $[-rg, rg]$	Average iterations	Average cycle indexes	Success rate
4	1	0.1	1	11.02	165.3	100%
4	1	0.01	1	11.21	168.15	100%
5	1	0.1	1	13.402	201.03	100%
5	1	0.01	1	13.406	201.09	100%
6	1	0.1	1	15.16	227.4	100%
6	1	0.01	1	15.324	229.86	100%

Table 4 Optimization parameter selection for a low-dimensional function

Parameters	Value
population $N$	60
spring up range $[down, up]$	$[-1, 1]$
up height $ri$	$[0.01, 0.1]$
local flying factor $c_1$	$[1, 1.5]$
global flying factor $c_2$	$[1, 1.5]$
range of neighbors $M$	$[3, 5]$
max flying speed $V_{\max}$	1
level fly cycles $F_1$	5
turning cycles $F_2$	$[5, 10]$
homing range $[-rg, rg]$	$[-10, 10]$



(a) The different combinations of  $[down, up], [-rg, rg]$  when  $ri = 0.1$  and  $n = 1, 2, 3$



(b) The different values of  $ri$  when  $[down, up] = [-rg, rg] = [-1, 1]$  and  $n = 1, 2, 3$

Fig. 6 The different combinations of  $[down, up], rise, [-rg, rg]$  when  $n = 1, 2, 3$

Table 5 The optimal value of each iteration

Iteration times	Optimal solution	Iteration times	Optimal solution
1	0.2878351374228	11	0.0056533942916
2	0.2878351374228	12	0.0044230302844
3	0.2793306299891	13	3.010200505e-04
4	0.2773423612152	14	4.270724202e-05
5	0.2773423612152	15	1.013942223e-05
6	0.2623245785077	16	3.222942948e-06
7	0.2623245785077	17	4.239604173e-07
8	0.2265190088335	18	2.607510770e-07
9	0.2237455514596	19	2.167854229e-09
10	0.0458584677429	20	5.505182976e-10

### 3.4 An example

To test the parameters of Table 4, a high-dimensional function, the Griewank function, is adopted here. This function has many local minimum points, and the number is related to the dimension of the problem. When the variable  $x_i \in [-600, 600]$ , the global minimum value 0 can be

obtained in  $(x_1, x_2, \dots, x_n) = (0, 0, \dots, 0)$ . The dimension is set to  $n = 30$ . The parameters are selected as in Table 4 to observe the suitability and effectiveness of the parameters in other functions. Fig. 7 shows the convergence process of Griewank. Table 5 shows the optimal value of each iteration. It can be seen that the optimal value decreases rapidly when the number of iterations increases from 9 to 10, and the optimal value is close to 0 when the number of iterations is 12. From Table 5, it can be seen that PCA can find the optimal value very well.

#### 4. Optimization analysis of PCA

To test the optimal performance of PCA to solve the global numerical optimization, the low-dimensional functions, high-dimensional functions and systems of nonlinear equations are adopted here. The selection of PCA parameters refers to Table 4. The termination condition is when the change in the value of  $P_b$  is less than  $10^{-6}$  for 50 iterations, and each group of experimental data was run 100 times. When the difference between the PCA minimum value  $P_b$  and the real minimum value of the functions is less than  $10^{-5}$ , it is deemed to succeed; otherwise, it is not successful. PCA has three processes, and each process has its cycle indexes. Calculate the total of the indexes of the three processes, and then, calculate the average cycle indexes and the success rate over 100 run times.

##### 4.1 Low-dimensional function optimization

Labeling a function as a low-dimensional function means that the function has fixed unknown numbers. Usually, such a function does not have many dimensions. Here, three famous functions are selected.

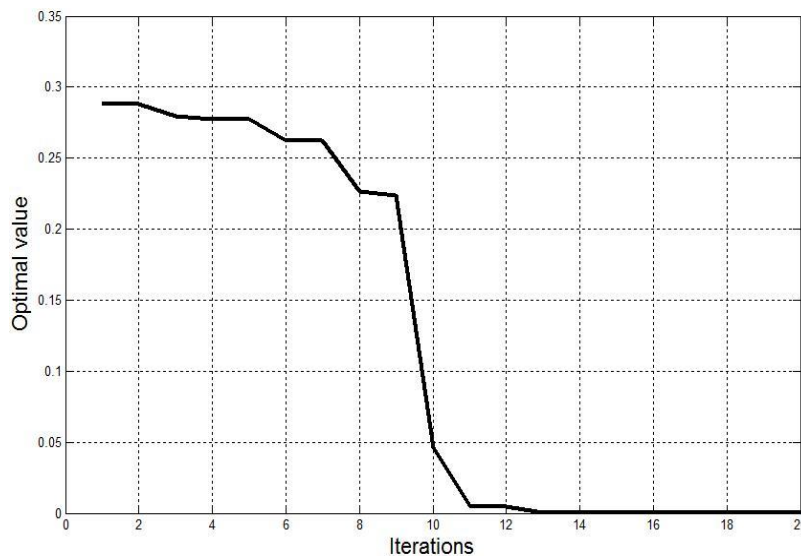


Fig. 7 The Optimal value changed with the iterations increasing

**Rosenbrock function:** This function is a typical pathological quadratic function that is difficult to minimize; when  $x_i \in [-10, 10]$ , place  $(0, 0)$  obtains the minimum value of 0. There is a narrow valley between the global optimal and local optimal that can be reached, and the steepest descent direction of a point on the surface of the valley is almost vertical to the minimum value of the best direction of the function. Because this function provides very little information for the search, the general algorithm finds it very difficult to discern the search direction; thus, there is only a slight chance of finding the global optimal point.

**Schaffer function:** This function is a multi-peak function that is difficult to optimize; the unique global optimum is surrounded by a large number of local optima, and around the function, there is a large amount of shock. When variable  $x_i \in [-10, 10]$ , in place  $(0, 0)$ , the minimum value of 0 can be obtained.

**Shuber function:** This function has multiple peaks, with several global and local optimal points, and there is a large amount of shock. When  $x_i \in [-10, 10]$ , the function is a multimodal function; there is a total of 760 local minimum points within its domain, and 18 of them are the global minimum points -189.7309.

Details on these functions are shown in Table 6. Table 7 shows that for the objective functions of low dimension, PCA can accurately find the global optimal solution, and the average cycle indexes are small, which reflects characteristics such as good global convergence, small cycle indexes and a high rate of convergence.

#### 4.2 High-dimensional function optimization

Labelling a function a high-dimensional function means that the function can have an uncertain number of unknowns. Usually, the dimensions that we used were larger than 30. In this test, four typical representative functions are selected; for the parameter selections of the PCA, refer to Table 4.

**Ackley function:** This function is a continuous experiment function that is reached by an exponential function plus moderate amplification of a cosine. When  $x_i \in [-32, 32]$ , the global optimal solution in place  $(x_1, x_2, \dots, x_n) = (0, 0, \dots, 0)$  obtains the value 0. The search of this function is extremely complex because a strict local optimization algorithm in the process of climbing will inevitably fall into the trap of a local optimum. Scanning a larger area can gradually achieve a better optimal point.

**Griewank function:** This function has many local minimum points, and the number is related to the dimension of the problem. When the variable  $x_i \in [-600, 600]$ , the global minimum value 0 can be obtained in  $(x_1, x_2, \dots, x_n) = (0, 0, \dots, 0)$ , and this function is a typical nonlinear multimodal function with a wide range of search space, which is usually considered to be a complex multimodal problem that is difficult to address by an optimization algorithm.

**Generalized Schwefel function:** This function is an inseparable multi-peak function and is somehow deceptive. When variable  $x_i \in [-500, 500]$ , the global optimal solution, which is related to the dimension  $n$ , is  $-418.983n$ . The global optimum and best local optimum are far apart, and thus, the convergence of the search algorithm is always moving in the wrong direction.

The details of these are shown in Table 8. Table 9 shows that when the dimension  $n = 30, n = 100$ , PCA can still 100% find the global optimal solution, and the average number of



iterations and average cycle indexes are small; in addition, the cycle indexes do not increase obviously with the increase in the dimension. The results embody PCA under a high dimension and have characteristics such as having a strong global convergence ability, fast convergence rate and good stability.

Table 6 Optimization results on low-dimensional functions

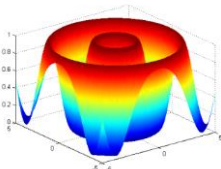
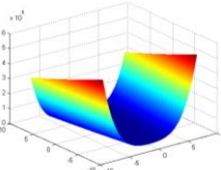
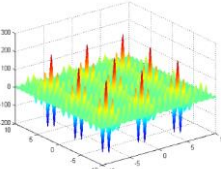
Function	Expression	Domain of definition	3D figure
Rosenbrock	$f(x, y) = (1 - x)^2 + 100(y - x^2)^2$	$x_i \in [-10, 10]$	
Schaffer	$f(x, y) = 0.5 + \frac{(\sin \sqrt{x^2 + y^2})^2 - 0.5}{(1 + 0.001(x^2 + y^2))^2}$	$x_i \in [-10, 10]$	
Shubert	$F(x) = \left[ \sum_{i=1}^5 i \cos((i+1)x_1 + i) \right] \left[ \sum_{i=1}^5 i \cos((i+1)x_2 + i) \right]$	$x_i \in [-10, 10]$	

Table 7 Optimization results on low-dimensional functions

Function	Domain of definition	Dimension $n$	Global optimal solution	Average cycle indexes	Success rate
Rosenbrock	$x_i \in [-10, 10]$	2	0	11.43	100%
Schaffer	$x_i \in [-10, 10]$	5	0	11.355	100%
Shubert	$x_i \in [-10, 10]$	3	-189.7309	12.1	100%

Table 8 Optimization results on high-dimensional functions

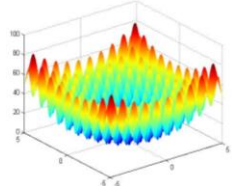
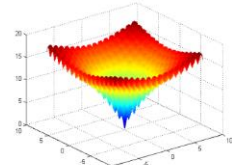
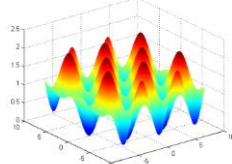
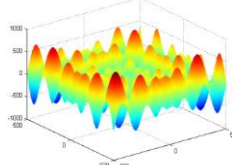
Function	Expression	Domain of definition	3D figure
Rastrigin	$F(X) = \sum_{i=1}^n \left[ x_i^2 - 10 \cos(2\pi x_i) + 10 \right]$	$x_i \in [-5.12, 5.12]$	
Ackley	$F(x) = -20e^{-0.2\sqrt{\frac{1}{n}\sum_{i=1}^n x_i^2}} - e^{\frac{1}{n}\sum_{i=1}^n \cos(2\pi x_i)} + e + 20$	$x_i \in [-32, 32]$	
Griewank	$F(x) = \sum_{i=1}^n \frac{x_i}{4000} - \prod_{i=1}^n \cos\left(\frac{x_i}{\sqrt{i}}\right) + 1$	$x_i \in [-600, 600]$	
Generalized Schwefel	$F(x) = \sum_{i=1}^n \left[ x_i \sin(\sqrt{ x_i }) \right]$	$x_i \in [-500, 500]$	

Table 9 Optimization results on high-dimensional functions

Function	Domain of definition	Dimension $n$	Global optimal solution	Average cycle indexes	Success rate
Rastrigin	$x_i \in [-5.12, 5.12]$	30	0	374.85	100%
		100	0	404.85	100%
Ackley	$x_i \in [-32, 32]$	30	0	982.5	100%
		100	0	1087.05	100%
Griewank	$x_i \in [-600, 600]$	30	0	366	100%
		100	0	388.8	100%
Generalized Schwefel	$x_i \in [-500, 500]$	30	-12569.49	3265.5	100%
		100	-41898.3	5191.55	100%

### 4.3 Nonlinear equation optimization

Nonlinear equations are represented in Eq. (19) below

$$F(x) = \begin{cases} f_1(X) = 0 \\ f_2(X) = 0 \\ \dots \\ f_m(X) = 0 \end{cases} \quad (18)$$

Vector  $(x_1, x_2, \dots, x_n)$  can be regarded as  $X$ , where  $m$  is the number of equations, and  $n$  is the number of unknown variables. The domain of the variable is  $a_i \leq x_i \leq b_i$  ( $i=1, \dots, n$ ), and  $a_i$ ,  $b_i$  are the lower limit and upper limit of  $X$ 's component  $x_i$ .

Solving a system of equations is equivalent to solving an optimization problem, such as Eq. (19)

$$\min f(X) = \sum_{i=1}^m |f_i(X)| \quad (19)$$

Table 10 Optimization results for the nonlinear equations

Equation set	Domain of definition	Optimal solution	Average cycle indexes	Success rate
$F(X) = \begin{cases} f_1(x) = (x_1 - 5x_2)^2 + 40\sin^2(10x_3) = 0 \\ f_2(x) = (x_2 - 2x_3)^2 + 40\sin^2(10x_1) = 0 \\ f_3(x) = (3x_1 + x_3)^2 + 40\sin^2(10x_2) = 0 \end{cases}$	$x_1, x_2, x_3 \in [-1, 1]$	$X = (0, 0, 0)$	3168	100%
$F(X) = \begin{cases} f_1(x) = x_1^2 - x_2 + 1 = 0 \\ f_2(x) = x_1 - \cos(0.5\pi x_2) = 0 \end{cases}$	$x_1, x_2 \in [-2, 2]$	$X = (0, 1)$ $(-1, 2)$ $(-1/\sqrt{2}, 1.5)$	1014.9	100%
$F(X) = \begin{cases} f_1(x) = (x_1 + 99.7091)^2 + x_2^2 - 10000 = 0 \\ f_2(x) = \sin(5x_1) + \cos(5x_2) - 1.9932 = 0 \end{cases}$	$x_1, x_2 \in [-2, 2]$	$X = (0.2909, 0)$	749.25	100%

When  $f(X)$  reaches the minimum value (the general equation is 0), the corresponding  $X$  is the solution of the equations; in other words, when  $f(X)$  is found, the global optimal solution is 0.

To test the three nonlinear equations in Table 10 (Tan 2011), the parameter selection for the PCA is the same as in Table 4.

Note that in the second equation,  $ri = 0.0001$ . In the third equation, set  $N = 100$ ,  $ri = 0.0001$ , and  $V_{\max} = 0.0001$ .

PCA can accurately find the optimal solution for the nonlinear equations in Table 10, rather than an approximate solution, and the numbers for the average cycle indexes are small, which embodies the PCA's advantages of fast global optimization, strong optimization capability and good stability.

The tests on the low-dimensional function, high-dimensional function and nonlinear equation show that the selected parameters in Table 4 are suitable and effective under most conditions.

## 5. Comparisons of PCA, GA and PSO

To confirm the superior performance of PCA, the famous GA and PSO are selected to compare. Here, three multimodal and complex functions, Rastrigin, Ackley and Griewank, are used to determine which can find the global optimal solution of the test functions fast and exactly. The three functions all have the same minimum value of zero. In the following tests, each group of experiment data is run 100 times to calculate the average of each algorithm's cycle indexes and success rates.

The parameter selection for the PCA is shown in Table 4. The termination condition is when the change in the value of  $P_b$  is less than  $10^{-15}$  for 10 iterations. The Matlab Toolbox's (MathWorks, Natick, MA, USA) Standard GA and PSO code was used to test the functions. To be consistent with the PCA, for the PSO, the population quantity was also set to 60. For the maximum flying speed  $V = 1$  to be consistent with the PCA, the learning factor was set to  $c_1 = c_2 = 2$ , which is better for the PSO (Eberhart 1998); the weighting coefficient  $w$  was set to decrease from 0.9 to 0.4 (Eberhart 1999), and the convergence condition was the same as in the PCA. For the Standard GA, the population quantity was also set to be 60 for consistency, and the convergence condition of the GA is related to the PSO, which can compare the optimal values at the same cost level.

The results obviously show that when the dimension increases, PCA can still find the global optimal value very well from Tables 11 to 13. The Standard GA and PSO cannot find the global optimal value exactly when the dimension increases, and the optimal value of the Standard GA becomes worse when the function is multimodal and complex. The accuracy of the optimal value of the PSO decreases rapidly when the dimension increases. In contrast, PCA can still find the optimal values more exactly when the dimension increases, and the average cycle indexes are less than those of the PSO most of the time. The comparison shows that PCA has strong global convergence, fewer algorithm cycle indexes and a fast convergence speed.

Table 11 The average cycle indexes and success rate of PCA, Standard GA and PSO with dimensions  $n = 1 \sim 10$  for Rastrigin

Dimension $n$	Optimal value	Optimal value of PCA	Average cycle indexes	Optimal value of Standard GA	Average cycle indexes of GA	Optimal value of PSO	Average cycle indexes
1	0	0	38.85	0.00482	1000	0	717
2	0	0	80.6449	0.24196	1000	0	1086
3	0	0	127.32	1.66975	1000	0	1300
4	0	0	165.3	1.28719	1000	0	1498
5	0	0	201.03	1.62723	2000	0	1622.92
6	0	0	227.4	4.36164	3000	0.9949590	2067.57
7	0	0	277.05	4.66171	3000	1.9899181	2254.78
8	0	0	288.15	4.98130	3000	2.9848771	1891.67
9	0	0	451.05	6.52652	3000	3.9798362	1800
10	0	0	310.5	7.44965	3000	6.9798362	2068
30	0	0	374.85	120.3924	3000	38.435065	2405.45
100	0	0	404.85	249.30225	3000	208.77619	3856.23

Table 12 The average cycle indexes and success rate of the PCA, Standard GA and PSO with dimensions  $n = 1 \sim 6$  for Ackley

Dimension $n$	Optimal value	Optimal value of PCA	Average cycle indexes of PCA	Optimal value of Standard GA	Average cycle indexes of GA	Optimal value of PSO	Average cycle indexes of PSO
1	0	0	72.9	0.13118	1000	0	100
2	0	0	155.1	0.33627	2000	0	1270
3	0	0	257.55	2.51639	2000	0	1516
4	0	0	395.1	2.78169	2000	3.552e-15	1655.17
5	0	0	534.45	3.34631	2000	3.552e-15	1866.66
6	0	0	872.4	4.99771	2000	3.552e-15	1900
30	0	0	982.5	11.1285	4000	2.131e-14	3321.3
100	0	0	1087.05	18.43115	8000	3.836e-13	7454.67

Table 13 The average cycle indexes and success rate of the PCA, Standard GA and PSO with dimensions  $n = 1 \sim 4$  for Griewank

Dimension $n$	Optimal value	Optimal value of PCA	Average cycle indexes of PCA	Optimal value of Standard GA	Average cycle indexes of GA	Optimal value of PSO	Average cycle indexes of PSO
1	0	0	119.4	0.04043	1000	0	205
2	0	0	193.08	0.17581	1000	0	918.75
3	0	0	6744.9	0.46214	10000	0.00739	1342.11
4	0	0	14154.75	0.53315	20000	0.03696	1422.54
30	0	0	366	13.7982	20000	0.02701	1734.7
100	0	0	388.8	450.4097	20000	9.103e-15	5233.21

## 6. Conclusions

To solve the traditional optimization algorithms' shortcomings, such as having a slow speed under the condition of nonlinearity, complexity and multiple peak calculations, poor convergence performance, and easily falling into a local optimal solution, PCA is proposed here, which is a new type of swarm intelligence optimization algorithm. This paper makes the following contributions:

(1) A new swarm intelligent optimization algorithm, PCA, has been originally proposed. The algorithm includes take-off, flying, and homing in three processes, and the architecture, steps and expressions have been given in detail.

(2) Parametric analysis of PCA has been conducted. The parameters are divided into three groups and use the control variable method to select the parameter values; parameter selection references have been given in Table 4. The parameters in Table 4 are suitable and effective in a low-dimensional function, high-dimensional function and nonlinear equation.

(3) PCA was used to conduct optimization tests on low-dimensional and high-dimensional functions and on nonlinear equations. The results show that PCA has the following features: 1) because this algorithm only needs to make a comparison between the objective function values, and the nature of the objective function is not constrained, it can be a function expression and can also be a functional form of representation; 2) good global convergence, small algorithm cycle indexes and a fast convergence rate; 3) the algorithm still has good global convergence, a high rate of convergence and strong stability for high-dimensional, multi-peak and complicated problems.

(4) PCA was compared with standard GA and PSO; the results prove the superiority of PCA.

## Acknowledgements

This research work was jointly supported by the 973 Program (2015CB060000), the National Natural Science Foundation of China (51421064, 51478081), the Science Fund for Distinguished Young Scholars of Dalian (2015J12JH209), and the Fundamental Research Funds for the Central Universities (DUT16LAB07).

## References

- Colorni, A., Dorigo, M. and Maniezzo, V. (1991), "A distributed optimization by ant colonies", *Proceedings of the 1st European Conference on Artificial Life*, Paris, December.
- Eberhart, R.C. and Kennedy, J. (1995), "A new optimizer using particle swarm theory", *Proceedings of the 6th International Symposium On Micro Machine and Human Science*, NJ.
- Elbeltagi, E., Hegazy, T. and Grierson, D. (2005), "Comparison among five evolutionary-based optimization algorithms", *Adv. Eng. Inform.*, **19**(1), 43-53.
- Eusuff, M., Lansey, K. and Pasha, F. (2006), "Shuffled frog-leaping algorithm: a memetic meta-heuristic for discrete optimization", *Eng. Optimiz.*, **38**(2), 129-154.
- Eusuff, M.M. and Lansey, K.E. (2003), "Optimization of water distribution network design using the shuffled frog leaping algorithm", *J. Water Res. Pl.-ASCE*, **129**(3), 210-225.
- Geem, Z.W., Kim, J.H. and Loganathan, G.V. (2001), "A new heuristic optimization algorithm: harmony search", *Simulat.*, **76**(2), 60-68.
- Holland, J.H. (1975), *Adaptation in natural and artificial systems*, University of Michigan Press, Ann Arbor, Michigan, USA.
- Holland, J.H. (1988), "Genetic algorithms and machine learning", *Mach. Learn.*, **3**(2), 95-99.
- Kennedy, J. (2010), *Encyclopedia of machine learning*, Springer, Berlin, Germany.
- Lawler, E.L. and Wood, D.E. (1966), "Branch-and-bound methods: A survey", *Oper. Res.*, **14**(4), 699-719.
- Lei, Y., Liu, C., Jiang Y.Q., and Mao, Y.K. (2013), "Substructure based structural damage detection with limited input and output measurements", *Smart. Struct. Syst.*, **12**(6), 619-640.
- Lei, Y., Wang, H.F. and Shen, W.A. (2012), "Update the finite element model of Canton Tower based on direct matrix updating with incomplete modal data", *Smart. Struct. Syst.*, **10**(4-5), 471-483.
- Li, J. and Law, S.S. (2012), "Damage identification of a target substructure with moving load excitation", *Mech. Syst. Signal Pr.*, **30**(7), 78-90.
- Li, J., Hao, H. and Lo, J.V. (2015), "Structural damage identification with power spectral density transmissibility: numerical and experimental studies", *Smart. Struct. Syst.*, **15**(1), 15-40.
- Li, X.L. and Qian, J.X. (2003), "Studies on artificial fish swarm optimization algorithm based on decomposition and coordination techniques", *Circ. Syst.*, **1**, 1-6.
- Li, X.L., Lu, F., Tian, G.H. and Qian, J.X. (2004), "Applications of artificial fish school algorithm in combinatorial optimization problems", *J. Shandong Univ. (Eng. Sci.)*, **5**, 015.
- MATLAB, *The MathWorks*, Inc. Natwick, MA (USA), <http://www.mathworks.com>.
- Møller, M.F. (1993), "A scaled conjugate gradient algorithm for fast supervised learning", *Neur. Net.*, **6**(4), 525-533.
- Nagy, M., Ákos, Z., Biro, D. and Vicsek, T. (2010), "Hierarchical group dynamics in pigeon flocks", *Nature*, **464**(7290), 890-893.
- Perera, T.B.D. and Guilford, T. (1999), "The orientational consequences of flocking behavior in homing pigeons, *Columba livia*", *Ethology*, **105**(1), 13-23.
- Qi, L. and Sun, J.A. (1993), "A nonsmooth version of Newton's method", *Math. Program.*, **58**(1-3), 353-367.
- Shi, Y. and Eberhart, R. (1998), "A modified particle swarm optimizer", *Proceedings of the IEEE World*

- Congress on Computational Intelligence*, Anchorage, May.
- Shi, Y. and Eberhart, R.C. (1999), "Empirical study of particle swarm optimization", *Proceedings of the 1999 Congress on Evolutionary Computation*, Washington, July, **3**.
- Spielman, D.A. and Teng, S.H. (2004), "Smoothed analysis of algorithms: why the simplex algorithm usually takes polynomial time", *ACM (JACM)*, **51**(3), 385-463.
- Tamm, S. (1980), "Bird orientation: single homing pigeons compared with small flocks", *Behav. Ecol. Sociobiol.*, **7**(4), 319-322.
- Tan, S.S. (2011), "A new swarm intelligent optimization algorithm: cell membrane optimization and its applications", Master. Dissertation, South China University of Technology, Guangzhou, China.
- Yan, X.H. (2010), "Research on path planning for mobile robot based on the biological intelligence", Ms.D. Dissertation, North China Electric University (Baoding), China.
- Yang, X.S. (2009), "Stochastic algorithms: foundations and applications", Springer, Berlin, Germany.
- Yi, T.H., Li, H.N. and Zhang, X.D. (2012a), "A modified monkey algorithm for optimal sensor placement in structural health monitoring", *Smart. Mater. Struct.*, **21**(10), 105033.
- Zhang, H.T., Chen, Z., Vicsek, T., Feng, G., Sun, L., Su, R. and Zhou, T. (2014), "Route-dependent switch between hierarchical and egalitarian strategies in pigeon flocks", *Sci. Rep.*, **4**, 1-7.
- Zhao, R.Q. and Tang, W.S. (2008), "Monkey algorithm for global numerical optimization", *J. Uncertain Syst.*, **2**(3), 165-176.
- Zuo, X., Chen, C., Tan, W. and Zhou, M. (2015), "Vehicle scheduling of an urban bus line via an improved multiobjective genetic algorithm", *J. Intell. Transport. S.*, **16**(2), 1030-1041.