A rotation learning-based colliding bodies optimization algorithm

Hui Liu¹, Bing Liu¹, Yong Li¹, Shiqi Cui¹

¹School of Computer Science &. Engineering, Changchun University of Technology, Changchun 130012, Jilin, China

Corresponding author and e-mail: Yong Li, LiYong@ccut.edu.cn

Abstract. In this paper, we propose an enhancement algorithm of colliding bodies optimization based on rotation learning, named by RBL-CBO. Firstly, we use the rotation-based learning to search for any point in the rotating space by adjusting the rotation angle, thereby improving the ability of the proposed algorithm jump out of the local optimum. Next, we leverage the sinusoid-based nonlinear adjustment strategy to modify the control parameters to improve the calculation accuracy of the proposed algorithm. Finally, we process the cross-boundary object by the mirroring strategy. We conduct extensive experiments to test the performance of the proposed algorithm. In simulation-based experiments, 23 benchmark functions are used to compare RBL-CBO algorithm with the CBO, DE, BBO, PSO and GSA algorithm. The experimental results demonstrate that the proposed RBL-CBO algorithm is superior to the other comparison algorithms, while the RBL-CBO algorithm is at least 20% higher than the CBO algorithm in terms of the accuracy of solving function optimization problem.

1. Introduction

The intelligent optimization algorithm starts from any solution and explores the optimal solution in the whole solution space according to a certain mechanism. The intelligent optimization algorithms can be divided into two types, one is to use the mathematical algorithm, and the other is to use the heuristic algorithms of natural phenomena such as biology and evolution [1]. In some optimization problems, mathematical algorithms require a large amount of calculation and time. Therefore, their application in practical problems is hindered. Later it was found that many heuristic algorithms have unique advantages in solving multivariable and discontinuous problems, which causes many scholars and experts to study and explore in this field [2]. There are some representative meta-heuristic algorithms such as: Differential Evolution (DE) [1], Biogeography Based Optimization (BBO) [3], Gravitational Search Algorithm (GSA) [4] and Particle Swarm Optimization (PSO) [5].

Colliding bodies optimization (CBO) algorithm is an excellent optimization algorithm, proposed by Kaveh A et al. in 2014[6]. Because the CBO algorithm is simple and the easy to accomplish, many scientists apply this algorithm to solve the practical engineering problem. For example, Bouchekara et al. used the improved CBO algorithm to solve the OPF problem [2]; Partha S.Pal et al. applied the algorithm for stable and unstable nonlinear systems [7]. However the CBO algorithm is still limited by its search accuracy and easy fall into the local optimal solution in a later iteration, so we propose an enhancement algorithm to improve the above disadvantages. First, a rotation-based learning mechanism is applied, that is, by rotating different angles counterclockwise on a specific circle in a...
two-dimensional space, any point in the search space can be found. In addition, the sinusoid-based nonlinear strategy and mirroring strategy are also used to improve the search accuracy of the algorithm. To verify the performance, we choose 23 benchmark test functions in this paper. The experimental results prove that the RBL-CBO algorithm effectively balances the exploration and exploitation capabilities and enhances the global search ability.

The rest of the paper is shown below. In Section II, this paper describes the research overview of the CBO algorithm. In Section III, we describe the proposed RBL-CBO algorithm in details. In Section IV, we provide comparative experiments with 23 benchmark test functions. Finally, conclusions are drawn in the last section.

2. Colliding bodies optimization algorithm (CBO)

In CBO, each candidate solution can be regarded as a colliding body (CB). CB is divided into two groups, namely, the stationary group and moving group, a moving object moves towards a stationary object and collides with it, so that the two objects move to a better position. The main steps of the CBO algorithm can be described as:

Step 1: Initialize the position of each CB by formula (1):

\[ x_i^0 = x_{\min} + rand(x_{\max} - x_{\min}), i = 1,2,\ldots,n \]  

Step 2: Calculate the mass of each CB.

\[ m_i = \frac{1}{\sum_{j=1}^{n} \frac{1}{fit(j)}}, k = 1,2,\ldots,n \]  

Where, \( fit(i) \) is the objective function value of the \( i \)th CB, and \( n \) is the number of population.

Step 3: CB is sorted in ascending order according to objective function values and divided into two equal groups, CB with smaller objective function values belongs to the static group, and those with the larger objective function values belong to the moving group. The speed of static group before collision is initialized to 0. The speed of the moving group before collision is initialized to formula (4).

\[ v_i = 0, i = 1,\ldots, \frac{n}{2} \]  

\[ v_i = x_i - x_{\frac{n}{2}}, i = \frac{n}{2} + 1,\ldots,n \]  

Where, \( x_i \) and \( v_i \) are the position vector and speed vector of the \( i \)th CB respectively.

Step 4: After collision, the speed of moving group and static group change. The speed of moving group CB is:

\[ v_i' = \frac{(m_i - \varepsilon m_i \frac{n}{2})v_i}{m_i + m_i - \frac{n}{2}}, i = \frac{n}{2} + 1,\ldots,n \]  

And, the speed of static group CB is:

\[ v_i' = \frac{\frac{n}{2} + \varepsilon m_i \frac{n}{2})v_i + \frac{n}{2}}{m_i + m_i - \frac{n}{2}}, i = 1,\ldots, \frac{n}{2} \]  

Where \( v_i \) and \( v_i' \) are the speed of the \( i \)th moving object before and after collision. The Coefficient of Restitution (COR) \( \varepsilon \) is define as:
Where, \( \text{iter} \) is the number of current iteration and \( \text{iter}_{\text{max}} \) is the maximum number of iterations.

Step 5: According to the speed after collision, the position of objects in each group after collision can be calculated. Then the new position of moving CB is

\[
x_{i}^{\text{new}} = x_{i} + \frac{n}{2} + \text{rand} \times v'_{i}, i = \frac{n}{2} + 1, \ldots, n
\]  

The new position of each stationary CB is

\[
x_{i}^{\text{new}} = x_{i} + \frac{n}{2} + \text{rand} \times v'_{i}, i = \frac{n}{2} + 1, \ldots, n
\]  

Where \( x_{i}^{\text{new}} \) is the new position after collision, rand is a random number distributed at (-1,1).

Step 6: If the number of current iterations equals the maximum number of iterations, the algorithm is terminated, and otherwise, return to step 2.

3. CBO algorithm based on rotation learning (RBL-CBO)

In this section, we briefly introduce the basic principles of rotation-based learning strategy, sinusoid-based nonlinear strategy and mirroring strategy. According to these strategies, we improve the CBO algorithm, the detailed steps are shown in section 3.4.

3.1. Rotation based learning strategy

Based on Euclidean distance, Rahnamayan, et al. proved that the candidate solution and its reverse solution were closer to the global optimum than the randomly selected solution. Rotating learning is proposed based on opposition learning strategy, it expands the opposition learning to two-dimensional space, and maps a candidate solution to a point on a specific circle in the two-dimensional space. Then the point is rotated to the specified angle counterclockwise to form a new point. The RBL is embedded in CBO, which enlarges the scope of global search and increases the diversity of the population [8].

As shown in Figure 1, A and B are the minimum and maximum of the search space respectively, the point C is the midpoint of the interval \([a, b] \) and its abscissa is \((a+b)/2\). Draw a circle of the radius \((b-a)/2\) centred at the point C. Rotate the point L by \( \beta \) degree(s) along the counter-clockwise direction of the circle to a new point N and make the projection point F of N on the x-axis, the angle of \( \angle NCB \) is \( \alpha + \beta \), the value of \( z^* \) is the rotation number of \( z \), and \( u^* \) can represent the value of \( CF \).

\[
u = |\overrightarrow{LZ}| = \sqrt{r^2 - u^2} = \sqrt{(z-a)(b-z)}
\]  

\[
u^* = r \times (\cos(\alpha + \beta)) = u \times \cos \beta - v \times \sin \beta
\]  

Figure 1. Search space of Rotation-based learning.
the rotation number of \( z^* \) can be expressed as:

\[
z^* = \frac{(a + b)}{2} + u^*
\]  

(13)

The concept of rotation number can be extended to high-dimensional space.

3.2. Nonlinear adjustment parameter strategy based on sinusoid
In CBO, the parameter \( \varepsilon \) plays an important role in balancing the global exploration ability and local exploitation ability of the algorithm. In order to improve the efficiency of the algorithm of the parameter \( \varepsilon \) at different stages, the linear strategy changed to suitable nonlinear adjustment strategy. By using the characteristic of the nonlinear reduction of the sinusoid in \([0, \pi/2]\), the control parameter \( \varepsilon \) adjustment can be expressed as formula(10) \[9\]

\[
\varepsilon = a_{\text{start}} - (a_{\text{start}} - a_{\text{end}}) \cdot \sin\left(\frac{\pi \cdot t}{t_{\text{max}}}\right)
\]  

(14)

Where \( a_{\text{start}} \) and \( a_{\text{end}} \) are the initial and final values of the control parameters. Since the value of \( \varepsilon \) is decremented by \([0, 1]\), \( a_{\text{start}} \) is set to 1 and \( a_{\text{end}} \) is set to 0 in this paper.

3.3. Cross-boundary object processing strategy
During the collision of objects, the update of the position will cause some objects to exceed the specified range. In this paper, we used a mirroring strategy to deal with out-of-bounds objects. The basic idea is to perform an image (boundary) process on the boundary, that is, map the object to a specified range \([11]\). Then \( x_i \)'s update range is:

\[
x_i = \begin{cases} 
2a_i - x_i, & x_i \leq a_i \\
2b_i - x_i, & x_i > b_i 
\end{cases}
\]

(15)

3.4. The main steps of the RBL-CBO algorithm
In RBL-CBO, rotation angle \( \beta \) needs to be given in advance. The Fixed or dynamic values can be obtained in the interval of \([0^\circ, 360^\circ]\) (random values are used in the paper). Then the main steps of the RBL-CBO algorithm are:

Step 1: Randomly initialize the value of \( \beta \) and the population \( P = \{X_1, X_2, ..., X_n\} \)
Step 2: Calculate center point \( c_i \), \( r_i \) and rotation individual \( X_i^* \);
Step 3: Add rotation individual \( X_i^* \) into rotation population \( RP \), and choose \( n \) individuals from \( \{P, RP\} \) as initial population;
Step 4: Calculate the mass of CBs according to formula(2);
Step 5: Calculate the speed of stationary and moving group before collision;
Step 6: Calculate the value of \( \varepsilon \) according to the formula(14);
Step 7: Calculate the speed of stationary and moving group, and update the position after collision.
If the position exceeds the boundary, it is processed according to the formula(15);
Step 8: Determine whether the maximum number of iterations has been reached. If the maximum is reached, terminate the algorithm; otherwise, return to step 2.

4. Experiments and results
In order to verify the performance of the RBL-CBO algorithm, we compare RBL-CBO with CBO, DE[1], BBO[3], GSA[4] and PSO[5] algorithms and select 23 functions in literature[10], where F1-F7 represent the unimodal function, F8-F13 represent the multimodal function, and F14-F23 represent the fixed dimensional multimodal function. The parameter settings of the algorithms are shown in Table 1. Table 2 shows the experimental results of RBL-CBO and other comparison algorithms. Figure 2
shows convergence curves of six selected algorithms. To observe the experimental effect of the algorithms more intuitively, we set the ordinates of 6 functions to logarithmic form.

Figure 2. The experimental results on six selected functions of 23 test functions.

Table 1. Parameters setting of several algorithms.

| algorithm | parameter name | value       |
|-----------|----------------|-------------|
| BBO       | keepRate       | 0.2         |
|           | alpha          | 0.9         |
|           | pMutation      | 0.1         |
| PSO       | vmax,vmin      | 6,-6        |
|           | wmax,wmin      | 0.9,0.2     |
|           | c1,c2          | 1.496,1.496 |
| GSA       | Elitistcheck   | 1           |
Table 2. Comparative results between RBL-CBO and other algorithms.

| Function | Algorithm | RBL-CBO | CBO | BBO | DE | GSA | PSO |
|----------|-----------|---------|-----|-----|----|-----|-----|
| F1       |           | 1.24E-16 | 5.06E-08 | 1.18E-12 | 2.32E-12 | 7.91E-16 | 2.09E-05 |
| F2       |           | 5.26E-10 | 1.12E-08 | 5.47E-03 | 5.74E-05 | 1.77E-05 | 4.65E-01 |
| F3       |           | 2.63E-02 | 1.68E0   | 7.17E0   | 5.31E+1  | 8.33E+1  | 2.84E+1  |
| F4       |           | 1.70E-09 | 2.76E-04 | 3.94E-02 | 4.35E-02 | 1.90E-04 | 8.90E-04 |
| F5       |           | 4.83E+3  | 1.29E+3  | 2.54E0   | 7.41E+1  | 1.23E+1  | 1.76E+1  |
| F6       |           | 3.28E-12 | 6.81E-7  | 9.10E-12 | 9.22E-05 | 9.46E-06 | 2.28E-03 |
| F7       |           | 3.87E-4  | 3.77E-3  | 9.80E-1  | 2.24E-3  | 1.15E-02 | 3.41E-02 |
| F8       |           | -4.32E+3 | -2.47E+3 | -3.47E+3 | 2.18E+2  | -2.64E+03 | -6.59E+3 |
| F9       |           | 4.79E0   | 3.97E0   | 1.71E0   | 1.69E0   | 2.28E+1  | 1.18E+1  |
| F10      |           | 3.40E-12 | 2.57E-09 | 2.30E-03 | 7.01E-07 | 1.04E-05 | 3.94E-03 |
| F11      |           | 0        | 1.18E-02 | 5.72E-02 | 7.39E-03 | 2.02E-05 | 1.11E-01 |
| F12      |           | 3.49E-21 | 5.23E-01 | 5.07E-06 | 2.84E-08 | 2.67E-09 | 4.74E-01 |
| F13      |           | 2.38E-24 | 1.22E-10 | 2.44E-06 | 9.31E-08 | 3.66E-04 | 3.01E-01 |
| F14      |           | 9.80E0   | 3.48E0   | 2.97E0   | 6.90E0   | 7.04E0   | 9.98E-01 |
| F15      |           | 2.20E-8  | 2.03E-4  | 7.22E-2  | 6.53E-3  | 9.20E-03 | 2.03E-02 |
| F16      |           | -1.03E-0 | -1.03E0  | -1.03E0  | -1.03E0  | -1.03E0  | -1.03E0  |
| F17      |           | 3.98E-1  | 3.98E-1  | 3.98E-1  | 3.98E-1  | 3.98E-1  | 1.71E+1  |
| F18      |           | 3.00E0   | 3.00E0   | 3.00E0   | 3.00E0   | 3.16E0   | 3.00E0   |
| F19      |           | -3.86E0  | -3.86E0  | -3.86E0  | -3.86E0  | -3.86E0  | -3.86E0  |
| F20      |           | -3.32E0  | -3.22E0  | -3.29E0  | -3.32E0  | -1.59E0  | -3.27E0  |
| F21      |           | -1.02E+1 | -5.24E0  | -6.68E0  | -7.87E0  | -5.06E0  | -5.15E0  |
| F22      |           | -1.04E+1 | -9.72E0  | -7.20E0  | -7.59E0  | -7.59E0  | -6.49E0  |
| F23      |           | -1.05E+1 | -9.45E0  | -6.62E0  | -6.36E0  | -9.24E0  | -8.03E0  |
In order to ensure the fairness of the experiment, all algorithms are carried out under the same conditions, where the number of collisions (i.e., the number of agents) \( n=30 \), the maximum number of executions \( t_{\text{max}}=500 \), each algorithm is run 30 times separately. From Table 2 and Figure 1, we can see that compared with other algorithms, RBL-CBO algorithm has achieved better convergence effects. It has better results on functions F15-F23, and poor results on F8 and F9, indicating that the performance of RBL-CBO still needs to be strengthened when solving multimodal function optimization problems. In summary, the performance of RBL-CBO is better than the other five comparison algorithms, and it can better handle most of the function optimization problems.

5. Conclusions
An improved CBO algorithm based on rotation learning proposed in this paper, not only improves the accuracy but also coordinates the exploration and exploitation of the algorithm. The experimental results of 23 benchmark functions show that RBL-CBO algorithm convergence and more stable than CBO, BBO, GSA, DE and PSO algorithm. So next, we will apply RBL-CBO to actual engineering problems to test its effectiveness furthermore.

Acknowledgement
Foundation items: Jilin Province Education Department Science and Technology Research and Planning Project, Clinical Big Data Analysis and Application for Diabetes, No. JJKH20181046KJ
Jilin Province Education Department Science and Technology Research and Planning Project, Energy-efficient and efficient cloud data center resource scheduling strategy research, number: JJKH20191310KJ.

References
[1] A.A. Abou El Ela, M.A. Abido, S.R. Spea, Optimal power flow using differential evolution algorithm, Electr. Power Syst. Res, 2017, vol 80, pp 878 – 885.
[2] Bouchekara H. Optimal power flow using an Improved Colliding Bodies Optimization algorithm, Applied Soft Computing, 2016, vol 42, pp 119-131.
[3] P.K. Roy, S.P. Ghoshal, S.S. Thakur, Biogeography based optimization for multi-constraint optimal power flow with emission and non-smooth cost function, Expert Syst. Appl, 2010, vol 37, pp 8221-8228.
[4] A. Bhattacharya, P.K. Roy, Solution of multi-objective optimal power flow using gravitational search algorithm, IET Gener. Transm. Distrib, 2012, pp 751-763.
[5] Kenndy J, Eberhart R C. Particle swarm optimization[C]. Proc of IEEE Int Conf on Neural Networks. Perth: IEEE, 1995, pp 1942–1948.
[6] Kaveh A, Mahdavai VR. Colliding Bodies Optimization method for optimum design of truss structures with continuous variables, Adv Eng Soft, 2014 vol 70, pp 1-12.
[7] Partha S. Pal and R. Kar. An efficient identification approach for stable and unstable nonlinear systems using Colliding Bodies Optimization algorithm, ISA Transactions, 2015, vol 59, pp 85–104.
[8] Huichao Liu, Zhijian Wu, Huanzhe Li, Rotation-Based Learning:A Novel Extension of Opposition-Based Learning, 2014, pp 511-522.
[9] Zhou M, Li T Y. Nonlinear adjustment strategy of inertia weight in particle swarm optimization algorithm[J]. Computer Engineering, 2011, 37(5):204-207.
[10] Guo W Y, Wang Y, Alternating sine cosine algorithm based on elite chaotic search strategy, Control and Decision, 2019, vol 34, pp 1654-1661. (In Chinese).