ORIGINAL ARTICLES

Two New Beetle Antennae Search (BAS) Algorithms and Their Comparative Investigation

Zongyuan Lin[1], Sile Ma*[2], Xiaojing Ma[2], Xiangyuan Jiang[1], Shuai Li[3]

[1] *College of Information and Control Engineering, China University of Petroleum (East China), Qingdao, China*
[2] *The Institute of Marine Science and Technology (IMST), Department of Control Science and Engineering, Shandong University, Qingdao, China*
[3] *Department of Computing, The Hong Kong Polytechnic University, Hung Hom, Kowloon, Hong Kong, China*

Received: October 23, 2018 **Accepted:** November 19, 2018 **Online Published:** December 11, 2018
DOI: 10.5430/ijrc.v2n1p9 **URL:** https://doi.org/10.5430/ijrc.v2n1p9

ABSTRACT

The Beetle Antennae Search (BAS) algorithm is a meta-heuristic search algorithm, which has efficient search capabilities. This paper presents two different variant algorithms based on the BAS algorithm, which are the BAS with fitness value (BASF) algorithm and BAS with local fast search (BASL) algorithm. The test results of 23 benchmark functions will be used to verify the reliability and accuracy of these algorithm. These benchmark functions include unimodal and multimodal high-dimensional functions, as well as fixed-dimensional multimodal functions. The test results show that the improved algorithm can search for the optimal solution globally and accurately with its own search strategy without environment parameters. Stability and accuracy are significantly improved while the calculation time does not change much.

Key Words: Beetle Antennae Search, Optimization, Benchmark function

1. INTRODUCTION

From the initial PID algorithm[1] has been presented, the control algorithm has been developed more than one hundred years old, and so many algorithms emerged during the period. At the same time the development history of control theory has also experienced three stages: classical control theory,[2] modern control theory[3] and intelligent control theory.[4] Nowadays, intelligent algorithms based on bionics theory have many branches, among many of them provide a lots effective solution to complex optimization problems, and the recent rise of artificial intelligence[5] have pushed the application of control theory to a new level.

Intelligent computing is also known as "soft computing",[6] which is inspired by the laws of nature, then imitate the algorithm for solving the problem according to the principle of it. Inspired by nature, imitating its structure for invention and creation, this is bionics. This is one aspect of our learning from nature. On the other hand, we can also use the bionic principle to design operations, which is the idea of intelligent computing. The initial algorithm in this paper belongs to the meta-heuristic search algorithm in the bionic domain. Like Genetic Algorithm (GA)[7] and Genetic Programming[8] based on evolutionary algorithms, Black Hole algorithm[9] and Simulated Annealing algorithm[10] based on physics-based algorithm, and swarm-based algorithm such as Particle Swarm Optimization (PSO) algorithm[11] and Ant Colony Optimization algorithm,[12] they are all based on the theory of bionics, and most of them are guided by two basic behaviors: exploration and exploitation.

*__Correspondence:__ Sile Ma; Email: masile@sdu.edu.cn; Address: The Institute of Marine Science and Technology, Shandong University, Qingdao, China.

The BAS algorithm[13, 14] is also derived from the behavior of mimicking the predation behavior of the beetle. It is similar to the single-particle PSO algorithm,[15] more like a gradient-free Gradient Descent algorithm[16] combined with a Simulated Annealing algorithm. Thanks to the core algorithm of BAS is very concise, new algorithms based on bas algorithm or combined with BAS algorithm emerge in an endless stream. The Beetle Swarm Optimization[17] algorithm proposed by Wang is combined with the PSO algorithm. The Beetle Swarm Antennae Search algorithm[18] proposed by Wang is to apply its application to group search.

The main work of this paper is proposed two improved algorithms based on BAS, which are named as BAS with fitness value (BASF) and BAS with local fast serach (BASL) algorithm. In this paper, 23 benchmark functions are tested for each of the three algorithms, and the comparison of test results is given. Among them, the BASF algorithm introduces the difference of the fitness value in the iterative process into the iterative change of $\boldsymbol{x}$, which weakens the influence of the step size. The BASL algorithm adds a local fast search when the global optimal value is updated. This local fast search uses small step size and large probability direction guidance to ensure that a potential local optimal solution can be searched. This algorithm greatly enhances the global optimization ability of BAS and does not significantly increase the calculation time.

The next organization of this article are as follows. In Section 2, this paper describes the flow and structure of BAS and two improved algorithms in detail. In Section 3, twenty-three benchmark functions are used to test the global optimization ability and accuracy of the three algorithms, and analyze their robustness by standard deviation. In Section 4, a conclusion is drawn.

2. Improved beetle antennae search algorithm design

2.1 Beetle Antennae Search Algorithm

As a meta-heuristic algorithm, BAS algorithm is mainly inspired by the beetle's behavior of prey, it treats the surrounding natural environment as the current search area and guides the beetles forward with the odor concentration in the air. The beetle takes two long antennas as signal receivers, and when the odor signal on one side is at a higher concentration than the other, then this beetle will move to a higher odor level ground by crawling or flying. When it reach the intended location, antennas head are full of randomness. So in this situation, the two long antennas must again compare the difference odor concentration, until the prey in the area is found. By changing the distance moved, beetle can effectively avoid falling into the local higher concentration of odor. So after the appropriate number of moves, it will have a great probability to find the right result.

The structure of BAS algorithm is described in pseudo code 1. First the article assume that the beatles position $\boldsymbol{x}$ at time tth is represented as $\boldsymbol{x}^t$, t denotes the number of iterations. Establish the benchmark function $f(\boldsymbol{x}^t)$, where variable $\boldsymbol{x}^t$=[$\boldsymbol{x}^1$,$\cdots$,$\boldsymbol{x}^k$], and k denotes the dimensions of $\boldsymbol{x}$. For the beetle exploration, the article set the two main parameters to update the new location information, they are direction $\overrightarrow{\boldsymbol{b}}$ and step length δ.

Algorithm 1: BAS algorithm

```
Input: Establish an objective function $f(\boldsymbol{x}^t)$, where variable $\boldsymbol{x}^t = [x_1, \cdots, x_i]^{\mathrm{T}}$ ,
       initialize the parameters $\boldsymbol{x}^0, d^0, \delta^0$.
Output: $\boldsymbol{x}_{\mathrm{bst}}$, $f_{\mathrm{bst}}$.
while (t < $T_{max}$) or (stop criterion) do
    Generate the direction vector unit $\overrightarrow{\boldsymbol{b}}$ according to (1);
    Search in variable space with two kinds of antennae according to (2);
    Update the state variable $\boldsymbol{x}^t$ according to (3);
    if $f(\boldsymbol{x}^t) > f_{bst}$ then
        $f_{\mathrm{bst}} = f(\boldsymbol{x}^t)$, $\boldsymbol{x}_{\mathrm{bst}} = \boldsymbol{x}^t$.
    Update sensing diameter $d$ and step size $\delta$ with decreasing functions (4) and (5)
    respectively, which could be further studied by the designers.
return $\boldsymbol{x}_{bst}$, $f_{bst}$.
```

So firstly, in a simple BAS algorithm model, the article set a random direction as follow,

$$\overrightarrow{\boldsymbol{b}} = \frac{rand(k,1)}{||rand(k,1)||}, \tag{1}$$

where $rand(.)$ denotes a random function. The article denotes the coordinates of the two antennae,

$$\begin{aligned} \boldsymbol{x}_r &= \boldsymbol{x}^{t-1} + d^t \overrightarrow{\boldsymbol{b}}, \\ \boldsymbol{x}_l &= \boldsymbol{x}^{t-1} - d^t \overrightarrow{\boldsymbol{b}}, \end{aligned} \tag{2}$$

where $\boldsymbol{x}_r$ denotes the beetle's right antenna coordinates and $\boldsymbol{x}_l$ denotes the left, and the article sets $\boldsymbol{x}$ represents the centroid of the beetle head model, d represents the variable related to search step δ.

Subsequently, as long as determined the movement of the centroid, it will be able to solve the optimization problem of the entire area. So in the following iterative manners,

$$\boldsymbol{x}^t = \boldsymbol{x}^{t-1} - \delta^t \overrightarrow{\boldsymbol{b}} \operatorname{sign}(f(\boldsymbol{x}_r) - f(\boldsymbol{x}_l)), \tag{3}$$

where δ is the step size of searching which accounts for the convergence speed following a decreasing function of t instead of an increasing function or a constant. The initialization of δ should be equivalent to the searching area. sign(.) represents a sign function.

$$\begin{aligned} d^t &= 0.95d^{t-1} + 0.01, & (4)\\ \delta_1^t &= 0.95\delta^{t-1}. & (5) \end{aligned}$$

where d^t denotes the distance between two antennae at tth times, and δ^t denotes the step size. It can be seen that both d^t and δ^t decrease with the times of iterations, this ensures that the algorithm can eventually converge to a certain value.

2.2 Fitness Function Value Iteration

To give a less appropriate example, in deep learning image recognition, the more features, the more reference information is given and the accuracy is improved. In the original BAS algorithm, the main influencing factors of $\boldsymbol{x}$ iteration are direction $\overrightarrow{\boldsymbol{b}}$ and step length δ, so we want to explore whether the fitness function value obtained by the search can also be used to guide the iteration of $\boldsymbol{x}$?

Based on the original BAS algorithm, let μ denotes $f(\boldsymbol{x}_r) - f(\boldsymbol{x}_l)$, as follows,

$$\begin{aligned} \boldsymbol{x}^t &= (1-w)\boldsymbol{x}^{t-1} + w\boldsymbol{x}_g^{t-1} - \delta^t \overrightarrow{\boldsymbol{b}} min(\mu, \alpha), (\mu \geq 0),\\ \boldsymbol{x}^t &= (1-w)\boldsymbol{x}^{t-1} + w\boldsymbol{x}_g^{t-1} - \delta^t \overrightarrow{\boldsymbol{b}} max(\mu, -\alpha), (\mu < 0), \end{aligned} \tag{6}$$

Where $\boldsymbol{x}_g$ denotes the current global optimal solution, and $w \in [0, 1]$ denotes the proportion of the best solution in this iteration. α is a positive number which set in advance in order to prevent excessive step size. In this way, $\boldsymbol{x}$ can keep a reasonable search range even when the changes are drastic, and converge effectively when approaching the value of the best fitness function value.

Since the random direction is the same dimension as the iteration $\boldsymbol{x}$, the global optimal solution is able to become the influencing factor of the new iteration direction. Therefore, in this paper, the optimal solution is unit vectorized and added to the randomly generated direction vector.

$$\begin{aligned} \overrightarrow{\boldsymbol{b_g}} &= \frac{\boldsymbol{x}_g}{||\boldsymbol{x}_g||}, & (7)\\ \overrightarrow{\boldsymbol{b}}^t &= c\overrightarrow{\boldsymbol{b}}^{t-1} + (1-c)\overrightarrow{\boldsymbol{b_g}}, & (8) \end{aligned}$$

Where $\overrightarrow{\boldsymbol{b_g}}$ denotes the unit vector of the global optimal solution $\boldsymbol{x}$, constant $c \in [0, 1]$ denotes the proportion of the direction vector. Here, in order to increase the probability that the beetle approaches the optimal solution. The direction vector unitization is not performed again.

2.3 Local Fast Search Iteration

During the test of BAS algorithm, we found that the search step of the beetle has a strong guiding effect, so that in some cases, the optimal solution will be missed. Therefore, this paper adds a local fast search ability to the original BAS algorithm. Just like a beetle in the predation process, then it is discovering a place with a high concentration of odor, but it does not want to stop, so it split itself, this split beetle is very small and can quickly search in the area and feed the final search results back to the original beetle.

One thing to note here is that local fast search is limited.

(1) Local fast searches require lag. Performing a local fast search from the beginning not only takes up a lot of computing resources, but also has poor convergence. So it is best to start after a certain number of iterations.

(2) Local fast searches need to be done after getting better results. A better fitness function values mean potentially more optimal solutions.

This paper establish the linear relationship between the distance of two antennas and the step size,

$$d^t = \frac{\delta^t}{c}, \tag{9}$$

In order to speed up the local search, this paper introduces two simpler centroid $\boldsymbol{x}$ iterative models,

$$\begin{aligned} \boldsymbol{x}_{bst}^m &= \boldsymbol{x}_l, (f_r > f_l), & \\ \boldsymbol{x}_{bst}^m &= \boldsymbol{x}_r, (f_r < f_l), & (10)\\ \boldsymbol{x}_{bst}^m &= \boldsymbol{x}_{bst}^{m-1} - \delta^{m-1}\overrightarrow{\boldsymbol{b}} \operatorname{sign}(f(\boldsymbol{x}_r) - f(\boldsymbol{x}_l)), & (11) \end{aligned}$$

Where $\boldsymbol{x}_{bst}$ denotes the optimal value in the local search, m is a pre-set constant, and the step size δ^m is also independent of the global search step δ. Here m determines the number of local fast searches, and the step attenuation is based on (4).

The algorithm structure is shown as pseudo code 2. The following is the search step attenuation mode for local fast search,

```
Algorithm 2: BASL algorithm
Input: Establish an objective function $f(\boldsymbol{x}^t)$, where variable $\boldsymbol{x}^t = [x_1, \cdots, x_i]^{\mathrm{T}}$ ,
       initialize the parameters $\boldsymbol{x}^0, d^0, \delta^0, M$.
Output: $\boldsymbol{x}_{\mathrm{bst}}$, $f_{\mathrm{bst}}$.
while $(t < T_{max})$ or (stop criterion) do
    Generate the direction vector unit $\overrightarrow{b}$ according to (1);
    Search in variable space with two kinds of antennae according to (2);
    Update the state variable $\boldsymbol{x}^t$ according to (3);
    if $f(\boldsymbol{x}^t) < f_{bst}$ then
        $f_{bst} = f(\boldsymbol{x}^t)$, $\boldsymbol{x}_{\mathrm{bst}} = \boldsymbol{x}^t$.
    if $f(\boldsymbol{x}^t) = f_{bst}$ then
        while $(m < M_{max})$ do
            $\delta^m = \delta^t$,$\boldsymbol{x}^m = \boldsymbol{x}^t$,$f^m_{bst} = f(\boldsymbol{x}^t)$;
            Generate the direction vector unit $\overrightarrow{b}$ according to (1);
            Search in variable space with two kinds of antennae according to (2);
            Update the state variable $\boldsymbol{x}^m$ according to (11) or (10);
            if $f(\boldsymbol{x}^m) < f^m_{bst}$ then
                $f^m_{\mathrm{bst}} = f(\boldsymbol{x}^m)$;
            Update step size $\delta^m$.
        $f^n_{gbst} = f^m_{bst}$.
    Update sensing diameter $d$ and step size $\delta$ with decreasing functions (9) and
    (12) or (13) respectively, which could be further studied by the designers.
    $f_{bst}$=min($f_{bst}, f^1_{gbst}, \cdots, f^n_{gbst}$).
return $\boldsymbol{x}_{bst}$, $f_{bst}$.
```

$$\delta_2^t = \lambda^{b/(a+\gamma i/n)} \cdot \delta^{t-1}, \tag{12}$$

Where $\lambda \in [0, 1]$ denotes a constant that determines how fast the step size decays. a, b and γ are constants. The following search step attenuation model is similar to it,

$$\delta_3^t = \delta_{min} + \delta^{t-1} cos(\frac{\pi}{2} \cdot \frac{i}{n}), \tag{13}$$

Where δ_{min} is the minimum value of the search step determined in advance.

The original step attenuation mechanism and the two step attenuation mechanisms proposed by us are shown in the Figure 1.

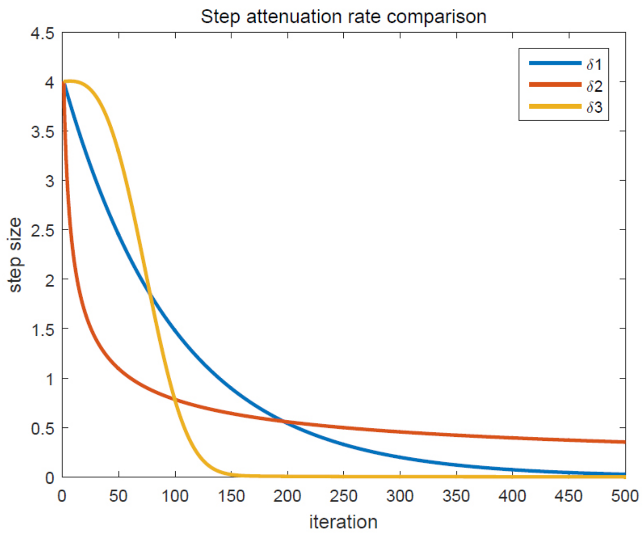

Figure 1. Step attenuation rate comparison. The blue line is the attenuation figure of the original step size δ_1 with the number of iterations. The red line in the figure indicates that the step size δ_2 decays with the number of iterations, where λ is 0.99, a is 0.1, b is 1 and γ is 10. The orange line represents the attenuation figure of step size δ_3, where δ_{min} is 0.001. Their initial step size is 100.

3. Benchmark validation

In order to verify the effectiveness of the bas algorithm and the proposed two improved algorithms, this paper selects 23 classical benchmark functions for testing research. These benchmark functions are reported in Table 1. Among them, functions $f_1 \sim f_7$ are unimodal high-dimensional benchmark functions, functions $f_8 \sim f_{13}$ are multimodal high-dimensional benchmark functions, and functions $f_{14} \sim f_{23}$ are fixed-dimension multimodal benchmark functions.

There we set the number of variables is 10, which is the dimension of the benchmark function. The *Range* in the table represents the optimal interval for each variable. Moreover, the theoretical optimal values of the benchmark functions in the table are reported in the last column of the table.

Figure 2 is a typical shape of three types of reference functions. As shown, Figure 2(a) presents a shape that converges around the center. Figure 2(b) shows that the function consists of many local peaks, but only one optimal value, which

Table 1. Description of unimodal high-dimensional, multimodal high-dimensional and fixed-dimension multimodal benchmark functions

$Function$	$Range$	f_{min}
$f_1(\boldsymbol{x}) = \sum_{i=1}^{n}(x_i^2)$	$[-100, 100]^{10}$	0
$f_2(\boldsymbol{x}) = \sum_{i=1}^{n}(\lvert\boldsymbol{x}_i\rvert) + \prod_{i=1}^{n}(\lvert\boldsymbol{x}_i\rvert)$	$[-10, 10]^{10}$	0
$f_3(\boldsymbol{x}) = \sum_{i=1}^{n}(\sum_{j=1}^{i}(\boldsymbol{x}_j))^2$	$[-100, 100]^{10}$	0
$f_4(\boldsymbol{x}) = \underset{i}{max}\lvert\boldsymbol{x}_i\rvert, \{1 \leq i \leq n\}$	$[-100, 100]^{10}$	0
$f_5(\boldsymbol{x}) = \sum_{i=1}^{n-1}([100(\boldsymbol{x}_{i+1} - (\boldsymbol{x}_i - 1)^2)])$	$[-30, 30]^{10}$	0
$f_6(\boldsymbol{x}) = \sum_{i=1}^{n}((\boldsymbol{x}_i + 0.5)^2)$	$[-100, 100]^{10}$	0
$f_7(\boldsymbol{x}) = \sum_{i=1}^{n}(i\boldsymbol{x}^4 + random[0, 1))$	$[-1.28, 1.28]^{10}$	0
$f_8(\boldsymbol{x}) = \sum_{i=1}^{n} -\boldsymbol{x}_i sin(\sqrt{\lvert\boldsymbol{x}_i\rvert})$	$[-600, 600]^{10}$	-4189.8
$f_9(\boldsymbol{x}) = \sum_{i=1}^{n}([\boldsymbol{x}_i^2 - 10cos(2\pi\boldsymbol{x}_i) + 10])$	$[-5.12, 5.12]^{10}$	0
$f_{10}(\boldsymbol{x}) = -20\exp(-0.2\sqrt{\frac{1}{n}\sum_{i=1}^{n}(\boldsymbol{x}_i^2)})$ $-\exp(\frac{1}{n}\sum_{i=1}^{n}(cos(2\pi\boldsymbol{x}_i))) + 20 + e$	$[-32, 32]^{10}$	0
$f_{11}(\boldsymbol{x}) = \frac{1}{4000}\sum_{i=1}^{n}\boldsymbol{x}_i^2 - \prod_{i=1}^{n}[\frac{\boldsymbol{x}_i}{\sqrt{i}}] + 1$	$[-600, 600]^{10}$	0
$f_{12}(\boldsymbol{x}) = \frac{\pi}{n}\{10sin(\pi y_1) + \sum_{i=1}^{n-1}((y_i - 1)^2[1 + 10sin^2(\pi y_{i+1})]$ $+(u_n - 1)^2)\} + \sum_{i=1}^{n}(u(\boldsymbol{x}_i, 10, 100, 4))$	$[-50, 50]^{10}$	0
$y_i = 1 + \frac{\boldsymbol{x}_i + 1}{4}, u(\boldsymbol{x}_i, a, k, m) = \begin{cases} k(\boldsymbol{x}_i - a)^m & \boldsymbol{x}_i > a \\ 0 & -a < \boldsymbol{x}_i < a \\ k(-\boldsymbol{x}_i - a)^m & \boldsymbol{x}_i < -a \end{cases}$		
$f_{13}(\boldsymbol{x}) = sin^2(3\pi\boldsymbol{x}_1)/10 + \sum_{i=1}^{n}(u(\boldsymbol{x}_i, 5, 100, 4))$ $+\sum_{i=1}^{n}((\boldsymbol{x}_i - 1)^2[1 + sin^2(3\pi\boldsymbol{x}_i + 1)] + (\boldsymbol{x}_n - 1)^2[1 + sin^2(2\pi\boldsymbol{x}_n)])$	$[-50, 50]^{10}$	0
$f_{14}(\boldsymbol{x}) = [\frac{1}{500} + \sum_{j=1}^{25} 1/\sum_{i=1}^{2}(x_2 - a_{ij})^6]^{-1}$	$[-65, 65]^2$	0.9980
$f_{15}(\boldsymbol{x}) = \sum_{i=1}^{n}[a_i - \frac{x_i(b_i^2 + b_i x_2)}{b_i^2 + b_i x_3 + x_4}]^2$	$[-5, 5]^4$	0.00030
$f_{16}(\boldsymbol{x}) = 4x_1^2 - 2.1x_1^4 + \frac{1}{3}x_1^6 + x_1 x_2 - 4x_2^2 + 4x_2^4$	$[-5, 5]^2$	-1.0316
$f_{17}(\boldsymbol{x}) = (x_2 - \frac{5.1}{4\pi^2}x_1^2 - 6)^2 + 10(1 - \frac{1}{8\pi})cos x_1 + 10$	$[-5, 5]^2$	0.398
$f_{18}(\boldsymbol{x}) = [1 + (\boldsymbol{x}_1 + \boldsymbol{x}_2 + 1)^2(19 - 14\boldsymbol{x}_1 + 3\boldsymbol{x}_1^2 - 14\boldsymbol{x}_2 + 6\boldsymbol{x}_1\boldsymbol{x}_2 + 3\boldsymbol{x}_2^2)]$ $\times[30 + (2\boldsymbol{x}_1 - 3\boldsymbol{x}_2)^2 \times (18 - 32\boldsymbol{x}_1 + 12\boldsymbol{x}_1^2 + 48\boldsymbol{x}_2 - 36\boldsymbol{x}_1\boldsymbol{x}_2 + 27\boldsymbol{x}_2^2)]$	$[-2, 2]^2$	3
$f_{19}(\boldsymbol{x}) = -\sum_{i=1}^{4}(c_i \exp(-\sum_{j=1}^{3}((\boldsymbol{x}_j - p_{ij})^2))$	$[1, 3]^3$	-3.36
$f_{20}(\boldsymbol{x}) = -\sum_{i=1}^{4}(c_i \exp(-\sum_{j=1}^{6}((\boldsymbol{x}_j - p_{ij})^2))$	$[0, 1]^6$	-3.32
$f_{21}(\boldsymbol{x}) = -\sum_{i=1}^{5}([(X - a_i)(X - a_i^T + c_i)]^{-1})$	$[0, 10]^4$	-10.1532
$f_{22}(\boldsymbol{x}) = -\sum_{i=1}^{7}([(X - a_i)(X - a_i^T + c_i)]^{-1})$	$[0, 10]^4$	-10.4028
$f_{23}(\boldsymbol{x}) = -\sum_{i=1}^{10}([(X - a_i)(X - a_i^T + c_i)]^{-1})$	$[0, 10]^4$	-10.5363

Table 2. Comparison of optimization results of fixed-dimension multimodal benchmark functions

F	BAS			PSO			GA		
	ave	std	$t(s)$	ave	std	$t(s)$	ave	std	$t(s)$
F_{14}	0.9980	1.62E-16	0.503	0.9980	0	3.110	0.9980	0	3.820
F_{15}	0.0021	5.93E-19	0.306	0.0042	0.0117	0.499	0.0039	0.0071	1.515
F_{16}	-1.0316	1.21E-15	0.297	-1.0316	0	0.427	-1.0316	0	1.244
F_{17}	0.3980	2.02E-17	0.307	0.3979	0	0.576	0.3979	0	1.217
F_{18}	3.0235	1.13E-15	0.299	3.0000	0	0.403	3.9000	4.9295	1.214
F_{19}	-3.8627	9.72E-15	0.278	-3.6913	0.124	0.6400	-3.8627	0	1.592
F_{20}	-3.3167	5.51E-15	0.318	-2.1198	0.556	0.6541	-3.2625	0.0605	1.894
F_{21}	-6.3933	6.64E-15	0.363	-1.0902	0.832	0.9072	-5.9724	3.3730	1.934
F_{22}	-6.3739	3.48E-15	0.359	-1.0196	0.406	1.0713	-7.3119	3.4237	2.129
F_{23}	-7.0295	2.95E-15	0.380	-1.2161	0.627	1.3545	-5.7112	3.5424	2.421

requires the algorithm to be able to search globally. Figure 2(c) shows a fixed 2-dimensional function with four local peaks, but only one optimal value.

In order to illustrate the effectiveness of the algorithm, this paper compares the bas algorithm with PSO and GA using the test data of the fixed-dimension multimodal benchmark function, and the comparison results are reported in Table 2. Moreover, 50 search agents were used, the maximum number of iterations was set to 1000, and each test function was run 30 times to generate statistical results. Meanwhile, the maximum number of iterations we set for the BAS algorithm is 500. Test results is performed using the average, standard deviation, and average single performance time of three performance indicators.

From the comparison of the results of BAS algorithm with PSO and GA algorithm in Table 2, it can be seen that BAS algorithm has great search ability. Compared with the PSO algorithm, it has better global search ability and higher search accuracy. It is also faster than the GA algorithm, and its standard deviation is smaller, which meaning that the value of the convergence function is more concentrated.

In order to verify the validity of the two variant algorithms proposed by us, this paper compares the test results of the two variants with the original BAS algorithm, and this paper sets the maximum number of iterations is 500, each test function is run 30 times to generate statistical results. The average, standard deviation and average single performance time obtained from the test are reported in Table 3.

The BASF and BASL test results with the original BAS algorithm are shown in Table 3, from the table we can see that the three algorithms are almost the same in each run time, but the BASF and BASL perform more accurately in the results. Especially for function f_2 f_5 f_9 and f_{12}, the accuracy of the results calculated by the BASL algorithm is more better than other results. This means that when dealing with some optimization issues, the BASL algorithm can solve the problem that the search of BAS algorithm is not comprehensive enough. But at the same time, we can also see that in the optimization problem of function f_{21} f_{22} and f_{23}, all three algorithms are not performing well. This also reflects the shortcomings of the single-particle search algorithm compared to the swarm intelligence algorithm. In the future, the population structure can be introduced into the BAS algorithm.

Figures 3 and 4 show the behavior of 2D search trajectories and function values for three typical models. From the a, d and g in Figure 3, we can intuitively see that the BAS algorithm is less efficient when it converges, and the search trajectory seems a bit messy, especially when the f_8 benchmark function test is performed, it misses the global optimal value. In contrast, the searching trajectories of BASL algorithm is more efficient, and when performing the f_8 benchmark function test, it explores a larger range and can accurately find the global optimal value. From c and f in Figure 4, we can also see that the BASL algorithm converges from a higher function value to the optimal value.

Table 3. Comparision of optimization results obtained for the unimodal, multimodal, and fixed-dimension multimodal functions

F	*BAS*			*BASF*			*BASL*		
	ave	*std*	$t(s)$	*ave*	*std*	$t(s)$	*ave*	*std*	$t(s)$
F_1	2.82E-9	1.43E-21	0.289	0.0021	1.09E-15	0.289	1.07E-4	3.95E-20	0.308
F_2	41.0627	1.88E-14	0.328	31.6048	5.89E-28	0.301	6.7123	1.52E-24	0.317
F_3	3.59E-9	1.96E-17	0.279	4.70E-9	5.89E-28	0.285	6.40E-9	1.52E-24	0.293
F_4	0.7266	7.09E-17	0.279	0.0953	6.08E-17	0.293	0.0284	2.91E-17	0.326
F_5	55.7363	4.15E-14	0.313	33.2297	1.68E-15	0.297	3.4581	8.42E-16	0.358
F_6	0.0010	4.55E-14	0.287	4.12E-4	3.36E-19	0.307	3.37E-7	2.07E-22	0.386
F_7	0.1328	3.04E-17	0.301	0.1799	4.46E-18	0.320	0.0373	3.80E-18	0.339
F_8	-2667.6	3.07E-12	0.304	-2985.8	1.32E-12	0.304	-3001.3	5.14E-12	0.351
F_9	44.6961	5.08E-15	0.330	24.4517	1.29E-15	0.295	2.9556	1.16E-14	0.321
F_{10}	1.5403	6.08E-16	0.321	0.0320	1.01E-17	0.326	0.0394	2.15E-17	0.320
F_{11}	0.4930	1.82E-16	0.320	0.0300	5.05E-16	0.304	0.0514	2.02E-17	0.352
F_{12}	0.5067	1.21E-16	0.332	0.2744	1.31E-16	0.314	1.20E-5	1.85E-21	0.362
F_{13}	0.0094	2.85E-18	0.326	5.52E-5	2.47E-20	0.319	2.88E-4	7.91E-21	0.332
F_{14}	0.9980	1.62E-16	0.503	0.9980	1.79E-15	0.583	0.9980	3.63E-16	0.434
F_{15}	0.0021	5.93E-19	0.306	0.0066	9.97E-18	0.290	0.0012	5.54E-19	0.342
F_{16}	-1.0314	1.21E-15	0.297	-1.0316	5.87E-16	0.286	-1.0316	1.90E-15	0.294
F_{17}	0.3980	2.02E-17	0.307	0.3979	9.72E-16	0.288	0.3979	8.10E-17	0.289
F_{18}	3.0235	1.13E-15	0.299	3.0306	9.72E-16	0.258	3.0038	1.21E-15	0.297
F_{19}	-3.8627	9.72E-15	0.278	-3.8477	1.37E-15	0.299	-3.8607	1.78E-15	0.319
F_{20}	-3.3167	5.51E-15	0.318	-3.3220	7.29E-16	0.324	-3.3207	4.86E-16	0.304
F_{21}	-6.3933	6.64E-15	0.363	-6.9808	2.59E-15	0.341	-7.3138	3.24E-15	0.365
F_{22}	-6.3739	3.48E-15	0.359	-8.0610	1.03E-14	0.361	-8.0519	3.24E-16	0.382
F_{23}	-7.0295	2.95E-15	0.380	-7.6920	1.31E-14	0.375	-8.1145	3.24E-15	0.502

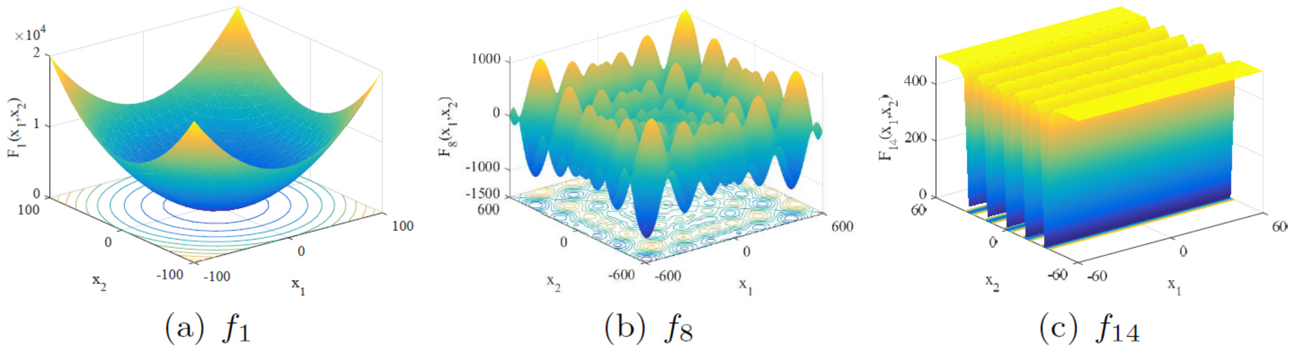

Figure 2. Benchmark function 3D space model. The three reference functions selected are f_1, f_8 and f_{14}, where their x and y axes denote the two-dimensional argument $\boldsymbol{x}$, and the z-axis denotes the function value. 3D version of unimodal function (a), multimodal function (b) and fixed-dimension multimodal function (c).

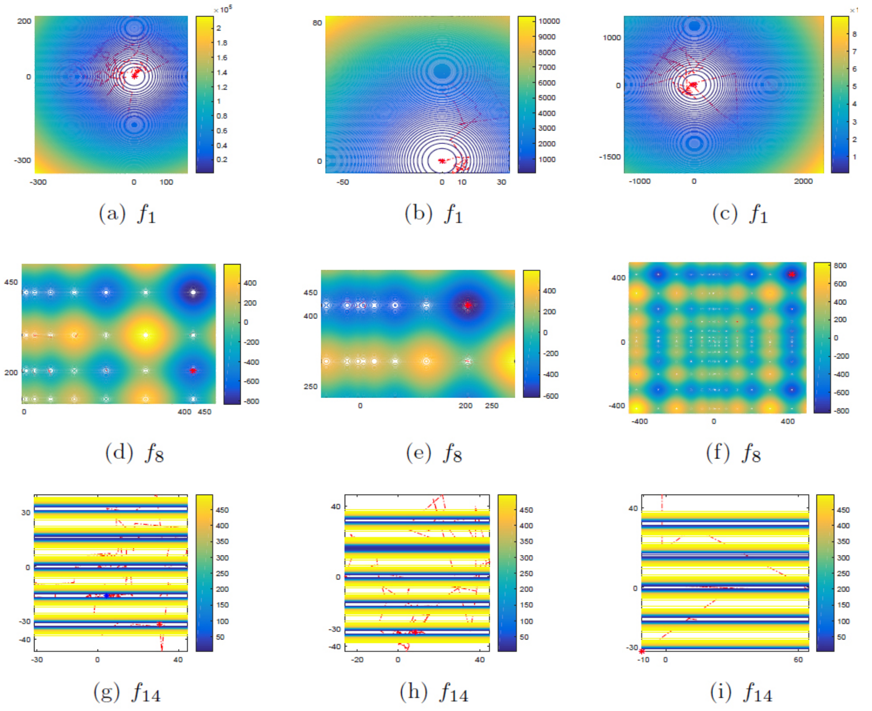

Figure 3. Behavior of BAS and its variant algorithms on the 2D benchmark problems. The 2D optimization trajectory of the BAS algorithm a, d and j. The 2D optimization trajectory of the BASF algorithm b, e and h. The 2D optimization trajectory of the BASL algorithm c, f and i.

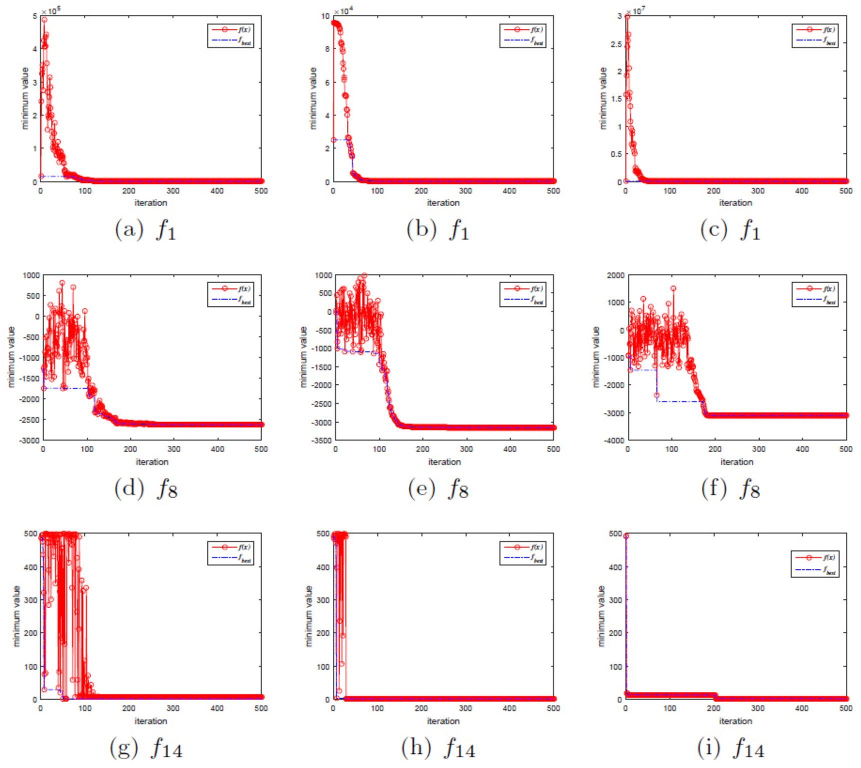

Figure 4. Behavior of BAS and its variant algorithms on the function values. The function value behavior j,m and p of BSA algorithm. The function value behavior b,e and h of BASF algorithm. The function value behavior l,o and r of BASL algorithm. The red line indicates the current optimal fitness f_{bst}, the blue line indicates the optimal value of each iteration $f(\boldsymbol{x})$ of the algorithms.

4. Conclusion

In this paper, two variant algorithms based on BAS algorithm are proposed, and the mathematical models of two variant algorithms are established. This paper completed the optimization problem of the BAS algorithm in 23 classical benchmark functions, and given three attenuation mechanisms for the search step size. Meanwhile, the performance and effectiveness of the bas algorithm are verified based on the comparison with PSO and GA algorithms. Finally, it is verified that the two variant algorithms are still improved by comparing the original BAS algorithm. Experiments show that the bas algorithm has strong robustness and stability. Considering that BAS algorithm has very fast optimization ability, we will continue to study the application of BAS and its improved algorithms in 3D path planning.

References

[1] Ang KH, Chong G, Li Y. PID control system analysis, design, and technology. IEEE Transactions on Control Systems Technology. 2005; 13(4): 559-576. https://doi.org/10.1109/TCST.2005.847331

[2] Skogestad S, Postlethwaite I. Multivariable feedback control: analysis and design. 2007; 2: 359-368. New York: Wiley.

[3] Andrei N. Modern control theory. Studies in Informatics and Control. 2006; 15(1): 51.

[4] Lin C. Intelligent control theory in guidance and control system design: An overview. Proceedings of the National Science Council, ROC. 2000; 24: 15-30.

[5] Bench-Capon TJ, Dunne PE. Argumentation in artificial intelligence. Artificial Intelligence. 2007; 171(10-15): 619-641.

[6] Jang JSR, Sun CT, Mizutani E. Neuro-fuzzy and soft computing: a computational approach to learning and machine intelligence. 1997.

[7] Grefenstette JJ. Optimization of control parameters for genetic algorithms. IEEE Transactions on Systems, Man, and Cybernetics. 1986; 16(1): 122-128. https://doi.org/10.1109/TSMC.1986.289288

[8] Koza JR. Genetic programming as a means for programming computers by natural selection. Statistics and Computing. 1994; 4(2): 87-112. https://doi.org/10.1007/BF00175355

[9] Hatamlou A. Black hole: A new heuristic optimization approach for data clustering. Information Sciences. 2013; 222: 175-184. https://doi.org/10.1016/j.ins.2012.08.023

[10] Van Laarhoven PJ, Aarts EH. Simulated annealing. In Simulated annealing: Theory and applications (pp. 7-15). Springer, Dordrecht; 1987.

[11] Kennedy J. Particle swarm optimization. In Encyclopedia of machine learning (pp. 760-766). Springer, Boston, MA; 2011.

[12] Dorigo M, Birattari M, Blum C, Clerc M, Sttzle T, Winfield A (Eds.). Ant Colony Optimization and Swarm Intelligence: 6th International Conference, ANTS 2008, Brussels, Belgium, September 22-24, 2008, Proceedings (Vol. 5217). Springer; 2008.

[13] Jiang X, Li S. BAS: beetle antennae search algorithm for optimization problems. arXiv preprint arXiv:1710.10724. 2017.

[14] Jiang X, Li S. Beetle Antennae Search without Parameter Tuning (BAS-WPT) for Multi-objective Optimization. arXiv preprint arXiv:1711.02395. 2017.

[15] Ji Z, Zhou JR, Liao HL, Wu QH. A novel intelligent single particle optimizer. Chinese Journal of Computers. 2010; 33(3): 556-561. https://doi.org/10.3724/SP.J.1016.2010.00556

[16] Madgwick SO, Harrison AJ, Vaidyanathan R. (2011, June). Estimation of IMU and MARG orientation using a gradient descent algorithm. In Rehabilitation Robotics (ICORR), 2011 IEEE International Conference on (pp. 1-7). IEEE.

[17] Wang T, Yang L, Liu Q. Beetle Swarm Optimization Algorithm: Theory and Application. arXiv preprint arXiv:1808.00206. 2018.

[18] Wang J, Chen H. BSAS: Beetle Swarm Antennae Search Algorithm for Optimization Problems. arXiv preprint arXiv:1807.10470. 2018.