Hindawi
Journal of Electrical and Computer Engineering
Volume 2018, Article ID 3847951, 9 pages
https://doi.org/10.1155/2018/3847951

Research Article

Optimized Parameter Settings of Binary Bat Algorithm for Solving Function Optimization Problems

Xiao-Xu Ma[1] and Jie-Sheng Wang[1,2]

[1]*School of Electronic and Information Engineering, University of Science & Technology Liaoning, Anshan 114044, China*
[2]*School of International Finance and Banking, University of Science & Technology Liaoning, Anshan 114044, China*

Correspondence should be addressed to Jie-Sheng Wang; wang_jiesheng@126.com

Received 7 October 2017; Accepted 11 April 2018; Published 14 May 2018

Academic Editor: Jit S. Mandeep

The bat algorithm (BA) is a new bionic intelligent optimization algorithm to simulate the foraging behavior and the echolocation principle of the bats. The parameter initialization of the discussed binary bat algorithm (BBA) has important influence on the convergence speed, convergence precision, and good global searching ability of the BBA. The convergence speed and algorithm searching precision are determined by the pulse of loudness and pulse rate. The simulation experiments are carried out by using the six typical test functions to discuss this influence. The simulation results show that the convergence speed of the BBA is relatively sensitive to the setting of the algorithm parameters. The convergence precision reduces when increasing the rate of bat transmitted pulse alone and the convergence speed increases the launch loudness alone. The proper combination of BBA parameters (the rate of bat transmitted pulse and the launch loudness) can flexibly improve the algorithm's convergence velocity and improve the accuracy of the searched solutions.

1. Introduction

Optimization is the selection of a best element from a set of some available alternatives with regard to some criterion. Optimization algorithm is a basic principle of nature, which shows many different advantages and disadvantages in computational efficiency and global search probability and has a vast variety of applications in research and industry [1]. The function optimization presents a formalized framework for modelling and solving some certain problems. Given an objective function, it takes a number of parameters as its inputs, whose goal is to find the combination of parameters and return the best value. This framework is abstract enough that a wide variety of different problems can be interpreted as function optimization problems [2].

However, the traditional function optimization algorithm is used to solve the typical problem with small dimension, often not applicable in practice. So people focus on the nature. Nature provides rich models to solve these problems (such as fireflies, bats, and ants). People discovered the swarm intelligence optimization algorithm by simulating natural biological systems. These models could stimulate computer scientists using household nontraditional tools to solve the application problems [3]. Now a lot of swarm intelligence optimization algorithms are proposed, such as particle swarm optimization (PSO) [4], ant colony algorithm (ACO) [5], bat algorithm (BA) [6], social learning optimization (SLO) algorithm [7], and chicken swarm optimization (CSO) algorithm [8]. They can be used in the dictionary learning remote sensing data, automotive safety integrity level positioning, economic dispatch, composition, and examples of the Cloud Service Composition of QOS awareness. Obviously, the study of swarm intelligence optimization has become an important research direction. The bat algorithm (BA) was proposed by professor Yang based on the swarm intelligence heuristic search algorithm in 2010 and is a kind of effective method to search the global optimal solution [9]. BA has attracted more and more attention because of its simple, less parameters, strong robustness, and the advantage of easy implementation. Firstly, BA was proposed to solve the problem with the continuous real search space. However, there are many optimization problems having the discrete

binary search space. So the binary bat optimization algorithm (BBA) was put forward to solve this kind of problems [10]. At present, many scholars have carried out many researches and proved that this algorithm has a certain competitive performance compared with other algorithms. BA has been applied in the multiobjective function optimization with the artificial neural network model [11], economic operation [6], and economic load dispatch in wind power generation system [12]. BBA has been applied in the simulation test point selection [13], low speed rolling bearing fault [14], and the optimization of the echo state network [15].

The parameter initialization of the discussed binary bat algorithm (BBA) has important influence on the convergence speed, convergence precision, and good global searching ability of the BBA. The convergence speed and algorithm searching precision are mainly determined by the rate of bat transmitted pulse and the launch loudness. In this paper, the function optimization problem is solved based on the binary bat algorithm (BBA). Then the parameter performance comparison and analysis are carried out through the simulation experiments in order to verify the binary bat algorithm's superiority. The paper is organized as follows. In Sections 2 and 3, the bat algorithm and binary bat algorithm are introduced. The simulation experiments and results analysis are introduced in detail in Section 4. Finally, the conclusion illustrates the last part.

2. Bat Algorithm

The bat algorithm is an intelligence optimization algorithm inspired by the echolocation behavior of bats [9]. Echolocation works as a type of sonar: bats, mainly microbats, emit a loud and short sound pulse. When they hit an object, after a fraction of time, the echo will return back to their ears. The bat receives and detects the location of the prey in this way. In addition, this amazing orientation mechanism makes bats able to distinguish the difference between an obstacle and a prey and allows them to hunt even in complete darkness. In order to simulate the foraging behavior of the bats, the biological mechanism of the bat algorithm to simulate the bats' foraging behavior obeys the following idealized assumptions:

(1) All bats use echolocation to sense distance, and they also know the difference between food/prey and the background barriers in some magical way.

(2) A bat flies randomly with velocity V_i at position X_i with a fixed frequency $f_{\min}$, varying wavelength λ, and loudness A_0 to search for prey. They can automatically adjust the wavelength (or frequency) of their emitted pulses and adjust the rate of pulse emission $r \in (0, 1)$ depending on the proximity of their targets.

(3) Although the loudness can vary in many ways, Yang assumes that the loudness varies from a large positive A_0 to a minimum constant value $A_{\min}$.

On the basis of three idealized assumptions, the algorithm generates a set of solutions randomly and then searches the optimal solution by cycle and strengthens the local search in the process of searching. By producing the local solution near the optimal solution by random flight, BA finally finds the global optimal solution.

The foraging space of bats is the d dimension. At time $t-1$, the location and the flight velocity of the ith bat are X_i^{t-1} and V_i^{t-1}, respectively, and X_* is the current global optimal location. At time t, the velocity and position of the ith bat are updated by using the following equations:

$$f_i = f_{\min} + \left(f_{\max} - f_{\min}\right)\beta \tag{1}$$

$$V_i^t = V_i^{t-1} + \left(X_i^{t-1} - X_*\right) f_i \tag{2}$$

$$X_i^t = X_i^{t-1} + V_i^t, \tag{3}$$

where $f_{\min}$ and $f_{\max}$ are the minimum and maximum frequency of the sound waves emitted by bats, respectively. β is a random number obeying the uniform distribution in [0, 1]. When setting up the initial values, the frequency of launch sound waves of each bat obeys the uniform distribution in $[f_{\min}, f_{\max}]$. First of all, according to (1), obtain their frequencies, and then, according to (2) and (3), update the velocities and positions.

For the local search, each bat carries out the random walk based on the optimal solution. The following equation is used to produce a new solution:

$$X_{\text{new}} = X_{\text{old}} + \varepsilon \overline{A}(t), \tag{4}$$

where ε is a random number obeying the uniform distribution in $[-1, 1]$, X_{old} is a solution randomly selected from the current optimal solution, and $\overline{A}(t)$ shows the average loudness for all bats at the tth iteration.

The update rules of the loudness of pulse emission of the bat A_i and the velocity r_i are described as follows: if a bat finds prey, it will reduce its impulse response and increase the velocity of its pulse emission. In BA, the loudness of pulse emission of the bat A_i and velocity r_i is adjusted by the following equations:

$$A_i^{t+1} = \alpha A_i^t \tag{5}$$

$$r_i^{t+1} = r_i^t \left[1 + \exp yt\right], \tag{6}$$

where r_i^0 is the initial velocity and A_i^0 is the initial loudness, which are selected randomly. α and y are constant ($0 < \alpha < 1$; $y > 0$).

3. Binary Bat Algorithm

A binary search space can be considered as a hypercube. The search agents (particles) of a binary optimization algorithm can only shift to the nearer and farther corners of this hypercube by flipping various numbers of binary bits. Hence, when designing the binary version of BA, some basic concepts of the velocity and position updating rules must be modified. In the continuous version of BA, the artificial bats can move around the search space by utilizing position and velocity vectors (or updated position vectors) within the continuous real domain. In discrete binary space, the position

updating means switching between "0" and "1." In order to achieve this change, this switching should be done based on the velocities of agents. In other words, a transfer function defines a transformation probability from 0 to 1 for a position vector element and vice versa. The transfer functions force the particles to move in a binary space. The transfer function of the discussed binary BA is defined in (7) and Figure 1 [10].

$$S\left(v_i^k(t)\right) = \frac{1}{1+e^{-v_i^k(t)}}, \tag{7}$$

where $v_i^k(t)$ is the velocity of particle i in k dimension at iteration t.

After calculating the probabilities using transfer functions, a new position updating equation is necessary to update particles' position as follows:

$$x_i^k(t+1) = \begin{cases} 0 & \text{If rand} < S\left(v_i^k(t+1)\right) \\ 1 & \text{If rand} \geq S\left(v_i^k(t+1)\right), \end{cases} \tag{8}$$

where $x_i^k(t)$ and $v_i^k(t)$ indicate the position and velocity of ith particle at tth iteration in kth dimension.

This method has a drawback as the particles are forced to take values of 0 or 1. So the particles remain unchanged in their positions when their velocity values increase. However, according to the concepts mentioned above for designing a transfer function, a better way is to oblige the particles with high velocity to switch their positions. A v-shaped transfer function and position updating rule are proposed in order to do this as in (9) and Figure 2.

$$\begin{aligned} V\left(v_i^k(t)\right) &= \left|\frac{2}{\pi}\arctan\left(\frac{\pi}{2}v_i^k(t)\right)\right| \\ x_i^k(t+1) &= \begin{cases} \left(x_i^k(t)\right)^{-1} & \text{If rand} < V\left(v_i^k(t+1)\right) \\ x_i^k(t) & \text{rand} \geq V\left(v_i^k(t+1)\right), \end{cases} \end{aligned} \tag{9}$$

where $x_i^k(t)$ and $v_i^k(t)$ indicate the position and velocity of ith particle at tth iteration in kth dimension and $(x_i^k(t))^{-1}$ is the complement of $x_i^k(t)$.

4. Simulation Experiments and Results Analysis

4.1. Test Functions. In the simulation experiments, six typical functions are adopted to verify the performance of BBA. The simulation environment adopts *Windows* 10 operating system, *INTEL* processor 2.40 GHz, and 3 G memory for *MATLAB*2014*b* simulation software. The testing functions are shown in Table 1, where $f_1 \sim f_3$ are unimodal functions and $f_4 \sim f_6$ are the multimodal functions.

4.2. Simulation Experiments and Results Analysis

4.2.1. Change of a Single Variable A_i. The initialization parameters of BBA are set as follows: the population size (noP) is 30, the number of iterations (Max_iteration) is 500,

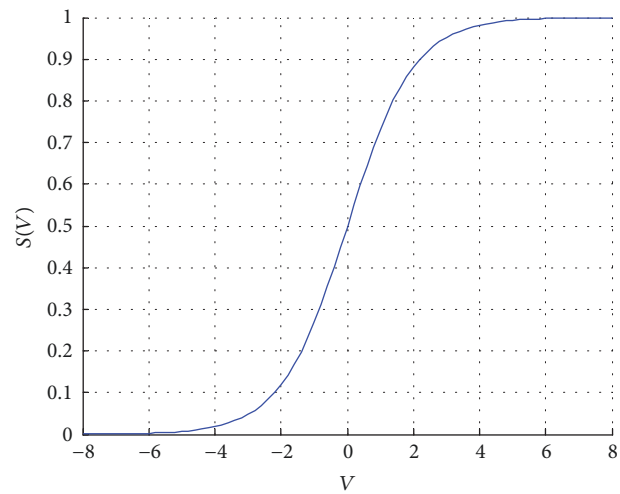


Figure 1: Sigmoid transfer function.

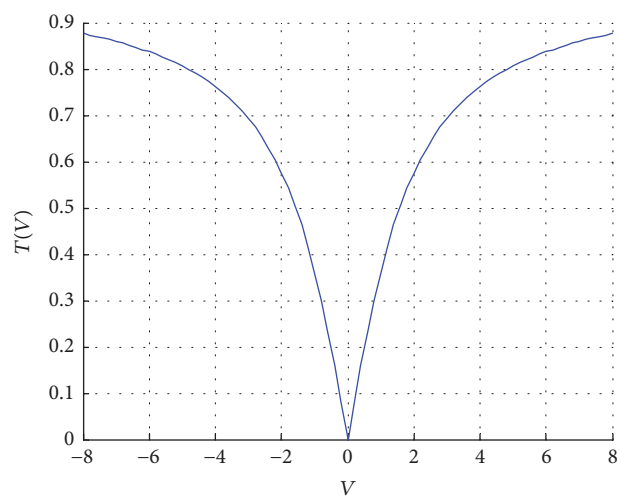


Figure 2: v-type transfer function.

$r = 0.1$, and the algorithm dimension (noV) is 100. In order to reduce the influence of random disturbance, the independent operating for each test function is carried out 10 times. The optimal value and average values of BBA in different launch loudness A_i are shown in Table 2. The simulation results of the six test functions are shown in Figure 3.

It can be seen from the convergence curves and the numerical results of six functions after 500 iterations and 10 times running independently that the optimization ability increases gradually when A_i changes from 0.25 to 0.9. Functions f_3 and f_4 get the optimal value when $A_i = 0.6$. Function f_1, f_5, and f_6 get optimal value when $A_i = 0.9$. The most obvious convergence curve is f_3 and the worst convergence curve function is f_1.

Compared with other convergence curves, function f_2 is the most volatile. The function searching capability affected by A_i is smaller and the searching running time is related to the function complexity. It can be seen from all convergence trends that the convergence rate did not increase or decrease regularly with the increase of A_i. Meanwhile, it has a certain relationship with the solution space. It is different from

Table 1: Simulation testing functions.

Function	Expression	Range	Min. value
f_1	$f(x) = \sum_{i=1}^{d=101} x$	$[-100, 100]$	0
f_2	$f(x) = \sum_{i=1}^{d=101} ([x+0.5])^2$	$[0, \pi]$	0
f_3	$f(x) = -20\exp\left(-0.2\sqrt{\frac{1}{n}\sum_{i=1}^{d=101} X_i^2}\right) - \exp\left(\frac{1}{n}\sum_{i=1}^{d=101}\cos(2\pi x_i)\right) + 20 + e$	$[-5.12, 5.12]$	0
f_4	$f(x) = \sum_{i=1}^{d=101} -x\sin\left(\sqrt{\lvert x\rvert}\right)$	$[-500, 500]$	-418.9829×5
f_5	$f(x) = \sum_{i=1}^{d=101} \frac{x_i^2}{4000} - \prod_{i=1}^{d=101}\cos\left(\frac{x_i}{\sqrt{i}}\right) + 1$	$[-32, 32]$	0
f_6	$f(x) = \sum_{i=1}^{d=101} \left[x_i^2 - 10\cos(2\pi x_i) + 10\right]$	$[-5.12, 5.12]$	0

Table 2: Performance comparison of BBA under different A_i.

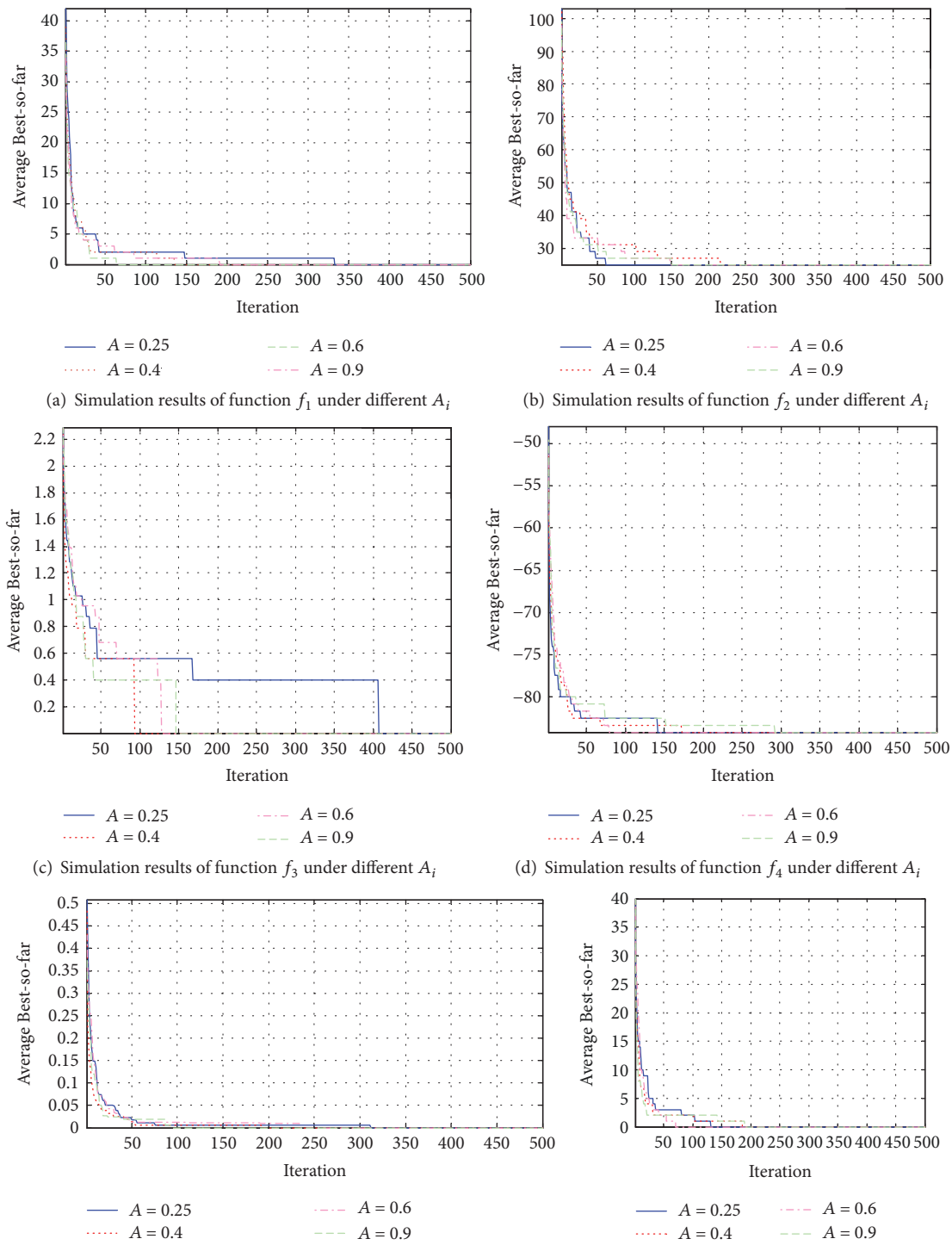
Function	Result	Simulation results of BBA under different A_i				Minimum time (s)
		0.25	0.4	0.6	0.9	
	Average	1.6	1.05	0.656	1.074	
f_1	Std.	3.6423	3.3974	3.0026	2.913	10.065 s
	Optimum	2	2	2	0	
	Average	26.58	28.176	27.068	26.984	
f_2	Std.	6.4989	7.8753	6.0687	6.5658	11.618 s
	Optimum	25	25	29	27	
	Average	0.42459	0.13147	0.55776	0.16727	
f_3	Std.	0.30212	0.30663	0.39227	0.33251	11.112 s
	Optimum	0.39603	0.55776	0.20291	0.39603	
	Average	−83.269	−83.287	−83.282	−82.781	
f_4	Std.	2.5872	2.7421	3.194	2.9798	12.209 s
	Optimum	−83.306	−82.464	−84.147	−83.306	
	Average	0.040308	0.0072643	0.012536	0.0078837	
f_5	Std.	0.040308	0.028619	0.037926	0.031519	11.426 s
	Optimum	0.0061266	0.011729	0.0068217	0.0053477	
	Average	1.172	1.008	0.85	0.986	
f_6	Std.	3.4621	2.9782	3.608	2.7468	11.01 s
	Optimum	1	3	1	1	

the function optimization performance impacted by the maximum or minimum values of parameter r_i. Hence, each function is corresponding to the optimal value of A_i. When A_i is 0.4, the optimization effect of function f_2 is the best. When A_i is 0.6, the optimization effects of functions f_3 and f_4 are the best. When A_i is 0.9, the optimization effects of functions f_1, f_5, and f_6 are the best.

4.2.2. Change of a Single Variable r_i. In view of r_i (the rate of bat transmitted pulse), the simulation experiments are carried out according to (6) and the simulation results are shown in Table 3 and Figure 4.

When $r = 0.2$, the test functions f_1, f_2, f_4, f_5, and f_6 obtain the optimal solution.The function f_3 gets the optimal solution when $r = 0.5$. With the increase of r, the optimal solution of functions f_1, f_3, and f_5 reduces gradually. The optimal solution of functions f_1, f_3, and f_5 also presents the tendency of decrease but $r = 10$ is better than $r = 2$. The operation time of the functions is almost the same time. From the convergence curves, function optimization precision is worse and worse, and convergence speed is slower and slower reduced in gradient with the increase of r. In a word, for different functions, the smaller r, the better the performance of convergence and the optimization ability is stronger and stronger.

4.2.3. Change of Variable A_i and r_i. In view of A_i (the loudness of bat transmitted pulse) and r_i (the rate of bat transmitted pulse), the simulation experiments are carried



(a) Simulation results of function f_1 under different A_i

(b) Simulation results of function f_2 under different A_i

(c) Simulation results of function f_3 under different A_i

(d) Simulation results of function f_4 under different A_i

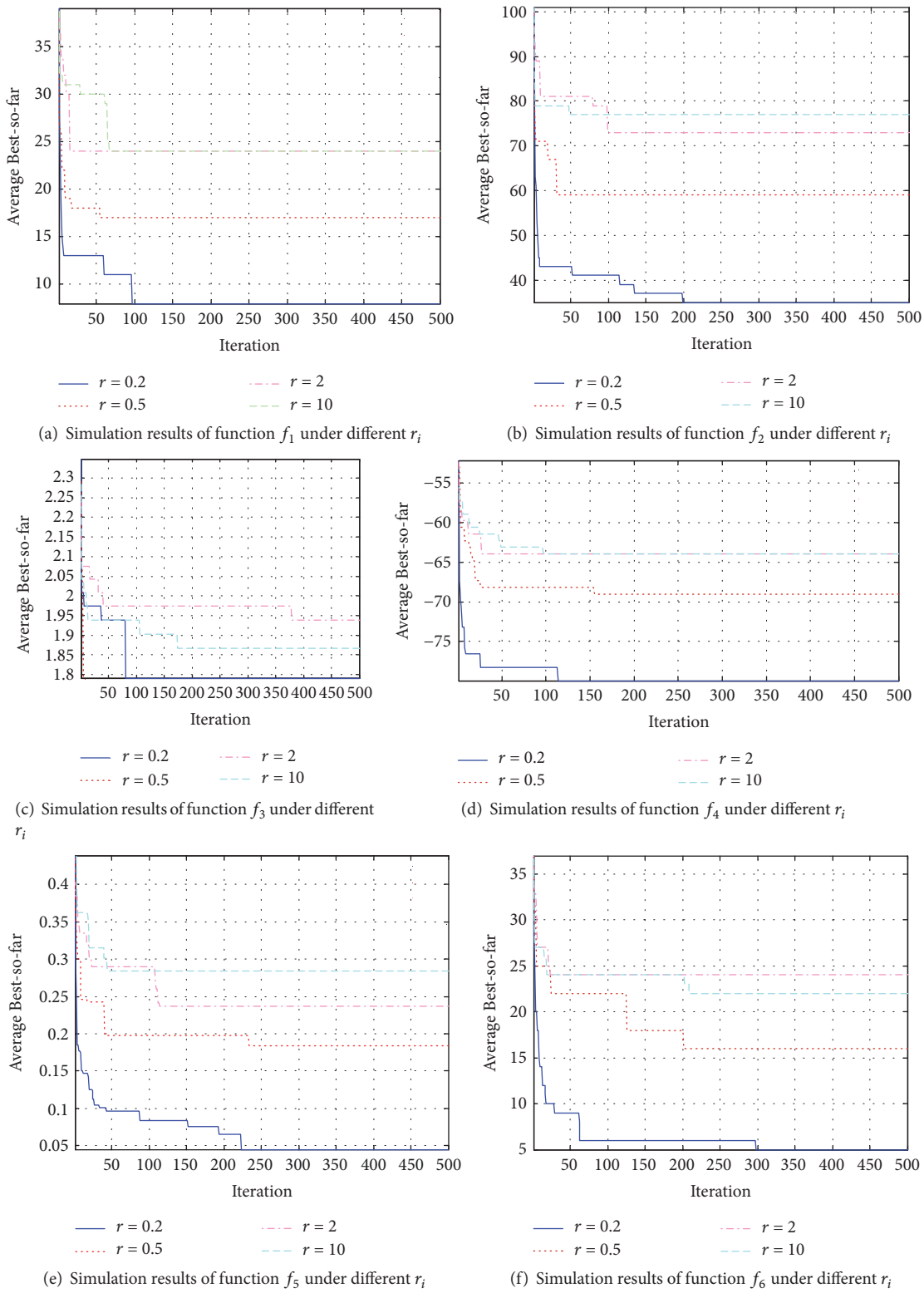
(e) Simulation results of function f_5 under different A_i

(f) Simulation results of function f_6 under different A_i

Figure 3: Simulation results of BBA under different A_i.

out and the simulation results are shown in Table 4 and Figure 5.

When A_i is 0.9 and r_i is 0.1, the optimization of function f_1 is best for the loudness of bat transmitted pulse A_i and the rate of bat transmitted pulse r_i. When A_i is 0.9 and r_i is 0.1, the optimization of function f_2 is best. When A_i is 0.6 and r_i is 0.1, the optimization of function f_3 is best. When A_i is 0.4 and r_i is 0.1, the optimization of function f_4 is best. When A_i is 0.9 and r_i is 0.1, the optimization of function f_5 is best. When A_i is 0.6 and r_i is 0.1, the optimization of function



(a) Simulation results of function f_1 under different r_i

(b) Simulation results of function f_2 under different r_i

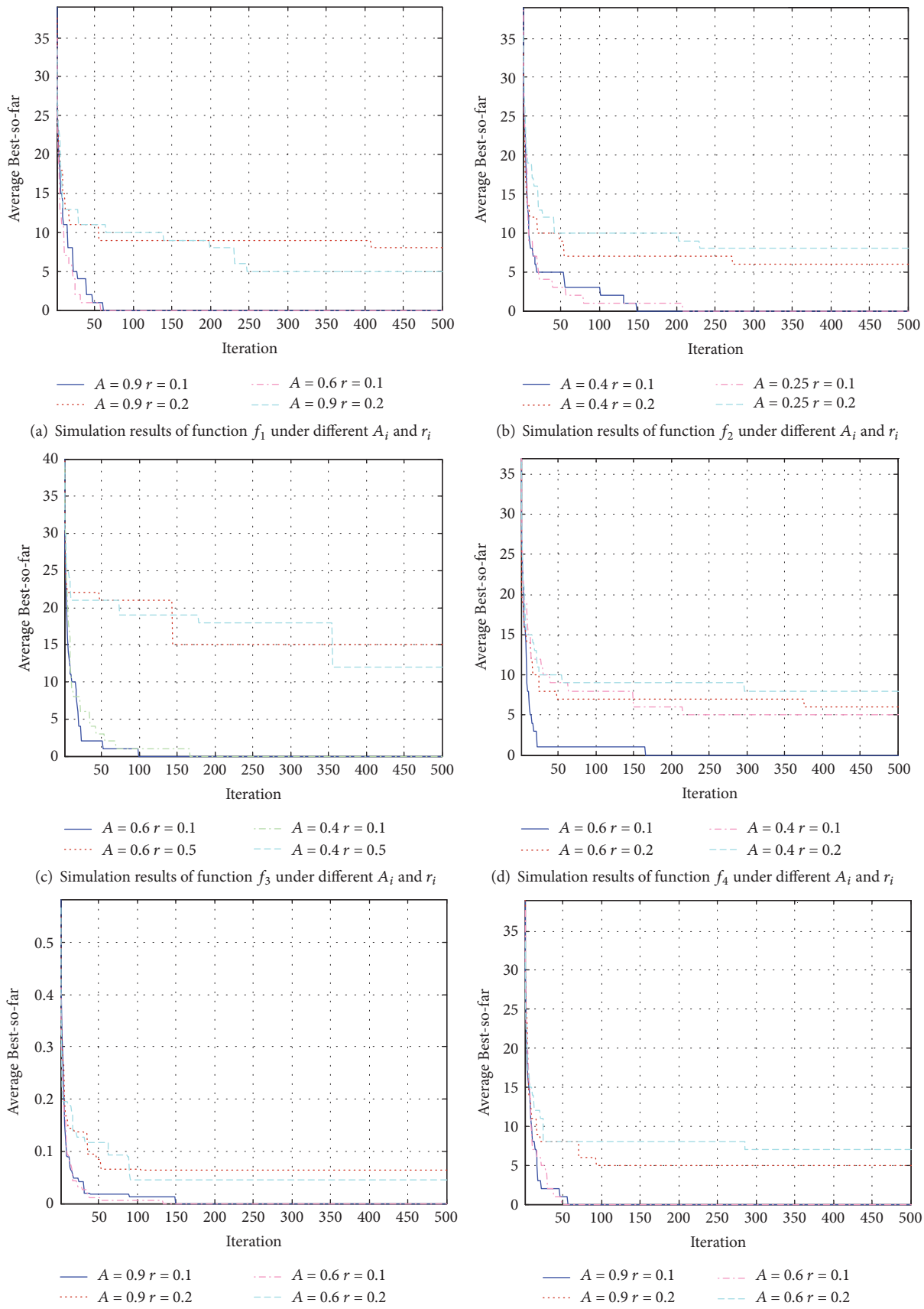
(c) Simulation results of function f_3 under different r_i

(d) Simulation results of function f_4 under different r_i

(e) Simulation results of function f_5 under different r_i

(f) Simulation results of function f_6 under different r_i

Figure 4: Simulation results of BBA under different r_i.



(a) Simulation results of function f_1 under different A_i and r_i

(b) Simulation results of function f_2 under different A_i and r_i

(c) Simulation results of function f_3 under different A_i and r_i

(d) Simulation results of function f_4 under different A_i and r_i

(e) Simulation results of function f_5 under different A_i and r_i

(f) Simulation results of function f_6 under different A_i and r_i

Figure 5: Simulation results of BBA under different A_i and r_i.

Table 3: Performance comparison of BBA under different r_i.

Function	Result	The simulation results of BBA under different r_i				Minimum time (s)
		0.2	0.5	2	10	
f_1	0.9	10	18	24	30	9.844000 (4)
f_2	0.4	37	69	85	81	11.263000 (3)
f_3	0.6	1.9033	1.7515	2.0421	1.9391	11.372000 (4)
f_4	0.6	−77.415	−66.476	−62.269	−60.586	11.934000 (4)
f_5	0.9	0.1069	0.18355	0.34503	0.31911	11.482000 (1)
f_6	0.9	8	19	24	28	11.083000

Table 4: Performance comparison of BBA under different A_i and r_i.

Function	The simulation results of BBA under different A_i and r_i		Optimum	Average
	A_i	r_i		
f_1	0.9	0.1	0	0.79
	0.9	0.2	8	9.272
	0.6	0.1	1	0.58
	0.6	0.2	8	7.474
f_2	0.4	0.1	1	1.292
	0.4	0.2	7	7.07
	0.25	0.1	0	1.152
	0.25	0.2	9	9.376
f_3	0.6	0.1	0	0.772
	0.6	0.5	22	16.872
	0.4	0.1	1	1.158
	0.4	0.5	20	17.022
f_4	0.6	0.1	7	7.15
	0.6	0.2	5	6.568
	0.4	0.1	1	0.75
	0.4	0.2	9	9.03
f_5	0.9	0.1	0.0068217	0.010572
	0.9	0.2	0.0751	0.072148
	0.6	0.1	0.0069992	0.0075418
	0.6	0.2	0.071033	0.060404
f_6	0.9	0.1	1	0.614
	0.9	0.2	9	5.75
	0.6	0.1	0	0.66
	0.6	0.2	8	7.914

f_6 is best. It can be seen from the convergence curves that the convergence precision reduces when increasing r_i alone and the convergence speed increases when reducing A_i alone. Different functions have an impact on optimization effect of BBA. The optimal combination of algorithm parameters is independent for different function.

5. Conclusions

Based on the basic principle of binary bat algorithm (BBA), the optimization performance is verified by carrying out the simulation experiments on six test functions. r_i has contact with the convergence precision. A_i relates to the convergence rate but the different functions require different A_i. Therefore, for different functions, the simulation experiments should be carried out in order to obtain the appropriate parameter setting. The simulation results show that the convergence speed of the algorithm is relatively sensitive to the setting of the algorithm parameters.

Conflicts of Interest

The authors declare that there are no conflicts of interest.

Authors' Contributions

Xiao-Xu Ma participated in the data collection, analysis, algorithm simulation, draft writing, and critical revision of this paper. Jie-Sheng Wang participated in the concept, design and interpretation and commented on the manuscript.

Acknowledgments

This work was supported by the Basic Scientific Research Project of Institution of Higher Learning of Liaoning Province (Grant no. 2017FWDF10), the Program for Research Special Foundation of University of Science and Technology of Liaoning (Grant no. 2015TD04), and the Opening Project of National Financial Security and System Equipment Engineering Research Center (Grant no. USTLKFGJ201502).

References

[1] M. M. Noack and S. W. Funke, "Hybrid genetic deflated Newton method for global optimisation," *Journal of Computational and Applied Mathematics*, vol. 325, pp. 97–112, 2017.

[2] N. Bell and B. J. Oommen, "A novel abstraction for swarm intelligence: particle field optimization," *Autonomous Agents and Multi-Agent Systems*, vol. 31, no. 2, pp. 362–385, 2017.

[3] X.-S. Yang, S. Deb, S. Fong, X. He, and Y.-X. Zhao, "From Swarm Intelligence to Metaheuristics: Nature-Inspired Optimization Algorithms," *The Computer Journal*, vol. 49, no. 9, Article ID 7562319, pp. 52–59, 2016.

[4] L. Wang, H. Geng, P. Liu et al., "Particle Swarm Optimization based dictionary learning for remote sensing big data," *Knowledge-Based Systems*, vol. 79, pp. 43–50, 2015.

[5] Y. Gheraibia, K. Djafri, and H. Krimou, "Ant colony algorithm for automotive safety integrity level allocation," *Applied Intelligence*, pp. 1–15, 2017.

[6] B. R. Adarsh, T. Raghunathan, T. Jayabarathi, and X.-S. Yang, "Economic dispatch using chaotic bat algorithm," *Energy*, vol. 96, pp. 666–675, 2016.

[7] Z.-Z. Liu, D.-H. Chu, C. Song, X. Xue, and B.-Y. Lu, "Social learning optimization (SLO) algorithm paradigm and its application in QoS-aware cloud service composition," *Information Sciences*, vol. 326, pp. 315–333, 2016.

[8] D. Wu, S. Xu, and F. Kong, "Convergence Analysis and Improvement of the Chicken Swarm Optimization Algorithm," *IEEE Access*, vol. 4, pp. 9400–9412, 2016.

[9] X. S. Yang, "A new metaheuristic bat-inspired algorithm," in *Nature Inspired Cooperative Strategies for Optimization (NICSO 2010)*, vol. 284, pp. 65–74, Springer, 2010.

[10] S. Mirjalili, S. M. Mirjalili, and X.-S. Yang, "Binary bat algorithm," *Neural Computing and Applications*, vol. 25, no. 3-4, pp. 663–681, 2013.

[11] N. S. Jaddi, S. Abdullah, and A. . Hamdan, "Multi-population cooperative bat algorithm-based optimization of artificial neural network model," *Information Sciences*, vol. 294, pp. 628–644, 2015.

[12] Y. Duan and M. Zhao, "Bat Algorithm on Economic Load Dispatch in Wind Power System," *Electrical Engineering*, 2015.

[13] D. Zhao and Y. He, "Chaotic binary bat algorithm for analog test point selection," *Analog Integrated Circuits and Signal Processing*, vol. 84, no. 2, article no. A006, pp. 201–214, 2015.

[14] M. Kang, J. Kim, and J.-M. Kim, "Reliable fault diagnosis for incipient low-speed bearings using fault feature analysis based on a binary bat algorithm," *Information Sciences*, vol. 294, pp. 423–438, 2015.

[15] H. Wang and X. Yan, "Optimizing the echo state network with a binary particle swarm optimization algorithm," *Knowledge-Based Systems*, vol. 86, pp. 182–193, 2015.