*ICIC Express Letters*
Volume **11**, Number **9**, September **2017**
ICIC International ©2017 ISSN 1881-803X
pp. 1401–1408

# IMPROVED BACKTRACKING SEARCH OPTIMIZATION ALGORITHM BASED ON MULTIDIMENSIONAL SEARCH AND NEW SELECTION STRATEGY

Zhilong Kang[1], Xueping Zhang[1], Lei Chen[2,3,*] and Yanju Guo[1]

[1]School of Electronics and Information Engineering
Hebei University of Technology
No. 5340, Xiping Road, Beichen District, Tianjin 300401, P. R. China
zhlk@hebut.edu.cn; 201521901029@stu.hebut.edu.cn; guoyanju@163.com

[2]School of Information Engineering
Tianjin University of Commerce
No. 409, Guangrong Road, Tianjin 300134, P. R. China
*Corresponding author: chenleitjcu@139.com

[3]School of Precision Instrument and Opto-Electronics Engineering
Tianjin University
No. 92, Weijin Road, Nankai District, Tianjin 300072, P. R. China

Received April 2017; accepted June 2017

Abstract. *The backtracking search optimization algorithm (BSA) is a relatively new optimization algorithm. However, BSA also has the disadvantages of slow convergence speed and precision similar to other algorithms. To solve these problems, a new improved algorithm, based on multidimensional search in the population evolution process and during the second choice using the selection probability, is proposed to overcome the demerits of BSA, named Fast and Tbest-guided Backtracking Search Optimization Algorithm (FTBSA). The first method can increase convergence speed, and the second method can prevent the algorithm from local optimum. FTBSA combined the above methods and can obtain the splendiferous optimization results with smaller population size and evolution generation. Experiments are conducted on a set of 6 benchmark functions. The results show that FTBSA can improve the convergence speed and precision of the algorithm and is more suitable for solving practical problems.*
**Keywords:** Backtracking search optimization algorithm, Improved algorithm, Multidimensional search, Selection probability

1. **Introduction.** In recent years, people have developed many optimization computation methods to solve complicated problems, such as differential evolution (DE) algorithm [1], artificial bee colony (ABC) algorithm [2], particle swarm optimization (PSO) algorithm [3], and differential search (DS) algorithm [4]. We call this kind of algorithms as "artificial life algorithm" [5]. Now a lot of optimization algorithms have been used in practical problems, such as solving the numerical optimization and combinatorial optimization problems, network and path planning and signal processing [6-8].

With the practical problems getting more and more complicated, experts begin to propose improved optimization algorithms based on the originals. For example, [9,10] transformed the search equation by using optimal solution of particles in the process of PSO algorithm. [11] introduced adaptive evolutionary strategy based on the DE algorithm. These improved algorithms all improve the global convergence ability and solving accuracy by introducing some study mechanisms and mutation operators.

The backtracking search optimization algorithm (BSA) [12] proposed by Civicioglu in 2013 is a new evolutionary algorithm based on swarm intelligence. Since BSA was introduced, it has captured much attention and it has been applied to various optimization problems [13]. For example, BSA has been used in antenna array design and the design of robust PSS in multimachine power systems. BSA's algorithm framework is similar to differential evolution algorithm, but very different in the process of mutation and crossover. Furthermore, its crossover strategy and mutation operation are novelty and efficiency, making the algorithm has a higher global searching capability and convergence result. BSA greatly improved the optimization efficiency by producing experimental population and controlling the search direction and boundary. At the same time, the operator of algorithm is easy for it only has one control parameter.

However, with the increasing of the iterations, BSA traps into local optimum easily and its convergence speed will slow down. To deal with the problems, this paper put forward two improvement strategies. The first strategy improved the generation mode of perturbation coefficient $F$ based on multi-dimensional search. This strategy is better at balancing the global and the local search ability than the original. The second strategy uses the selection probability during the second choice. That can avoid the algorithm falling into the local optimum to some extent. Through the above strategies, this paper proposed Fast and Tbest-guided Backtracking Search Optimization Algorithm (FTBSA), which has the new perturbation strategy and the new selection strategy at the same time, and also has a great improvement in convergence precision and speed [14]. Simulation results show that the FBSA possesses superior performance in convergence speed, convergence precision and robustness, as compared to the other algorithms. For high dimension problems, FTBSA can also have better results than other related algorithms. Furthermore, FTBSA can improve the performance and the efficiency of the optimization effectively.

The rest of the paper is organized as follows. Section 2 introduces the basic BSA. Second 3 shows the detail of the improved algorithm. Section 4 presents the experiments and results analysis. Finally, conclusions and some future works are made in Section 5.

## 2. **Backtracking Search Optimization Algorithm.**

BSA is a new evolutionary algorithm based on the swarm intelligence. It can be divided into five parts: initialization, select-I, mutation, crossover and select-II.

### 2.1. **Initialization.**

BSA initializes the population $P$ with Equation (1):

$$P_{i,j} \sim U(low_j, up_j) \tag{1}$$

where $U$ is the uniform distribution, $P_i$ is a target individual in the population $P$, *low* and *up* are the lower bound and upper bound of the search region and $i = [1, 2, 3, \ldots, popsize]$, $j = [1, 2, 3, \ldots, \dim]$.

### 2.2. **Select-I.**

After generating *popsize* food source randomly and setting the history population to control the search direction, the initial historical population is determined using Equation (2):

$$oldP_{i,j} \sim U(low_j, up_j) \tag{2}$$

The algorithm idea is as Equation (3):

$$\text{if } a < b \text{ then } oldP = P \tag{3}$$

where $a$, $b$ are random numbers that obey the uniform distribution on $(0, 1)$, we can choose a history population from the previous population through Equation (3), and remember the history population until it changes.

2.3. **Mutation.** When $oldP$ is confirmed, create new population according to Equation (4):

$$\text{Mutant} = P + F \cdot (oldP - P) \tag{4}$$

In Equation (4), $F$ is the parameter that controls the amplitude of the search direction matrix $(oldP - P)$, $F = 3 \cdot randn$ and $randn$ is a random number that obeys the standard normal distribution. The Mutant may be beyond the value range of the corresponding component, if the element of the population is beyond this range, then it will produce a new population within this range such as Equation (1).

2.4. **Crossover.** BSA's crossover strategy is a joint crossover strategy based on two crossover ways equal probability call. Set the mix proportion parameter to control the number of particles between the populations.

2.5. **Select-II.** This stage compared the individual fitness of $T$ and $P$ in the same position, recorded the population with better fitness as Equation (5).

$$P_i = \begin{cases} T_i & \text{if } fitness(T_i) < fitness(P_i) \\ P_i & \text{otherwise} \end{cases} \tag{5}$$

After the selection is completed, the global minimum value is updated to be the value with better fitness.

3. **Improvement of Backtracking Search Optimization.** The performance of the evolutionary algorithm is mainly determined by the convergence speed and precision; many improvements are carried out with this premise. BSA's convergence speed is fast at the beginning, but it easily trapped into the local optimum with the increasing of iterations. In this paper, we improved the perturbation coefficient and the second selection process of the BSA, which makes the algorithm have better convergence speed and precision.

3.1. **New perturbation strategy.** The generation mode of perturbation coefficient $F$ is as Equation (6) when we do a multidimensional search:

$$F = sign(0.5 - rand) \times \sqrt{CH3} \tag{6}$$

In Equation (6), $CH3$ is a random number that obeys the Chi-square with 3 degrees of freedom. Using the new perturbation strategy in the population evolution process, it can produce smaller scale amounts conducive to the local search and can also produce bigger scale amounts conducive to the global search. To be convenient for comparing, we call the basic BSA with the new perturbation strategy as Fast Backtracking Search Optimization Algorithm, briefly named FBSA.

3.2. **New selection strategy.** As the BSA is easy to fall into the local optimum, we add the selection probability $P$ during the second choice. The equation is defined as Equation (7):

$$P = \frac{fitness_i}{\sum\limits_{i=1}^{SN} fitness_i} \tag{7}$$

The selection strategy uses the roulette method to select a new population, and it can avoid the algorithm falling into the local optimum to a certain extent. We call the basic BSA with the selection probability during the second choice as Tbest-guided Backtracking Search Optimization Algorithm, briefly named TBSA.

3.3. **FTBSA.** For BSA's convergence speed is slow and the BSA is easy falling into the local optimum, according to above two improved strategies, we proposed an optimization algorithm, namely FTBSA. First, FTBSA improved the generation mode of perturbation coefficient $F$, then FTBSA used the selection probability during the second choice. Based on above two operations, the proposed algorithm has higher convergence speed and precision. We present the complete computational procedure of FTBSA as follows:

1) Population initialization;
2) Use the new perturbation coefficient as Equation (6), and produce a new population according to Equation (4);
3) Calculate the fitness of the original population and the newly produced population;
4) Select the optimum by using the greedy mechanism and take notes;
5) Add the selection probability as Equation (7), and select a new population by roulette method;
6) Record the current optimal position;
7) Judge the current iterations, if it is the max epoch, record the position; otherwise, return to 2.

## 4. Experiments and Results Analysis.

4.1. **Experimental design.** In order to test the performance of FBSA, TBSA and FTBSA, the experiment results are compared with the basic BSA and the DS algorithm. This paper designed five kinds of experiments: (I) BSA optimization experiment; (II) FBSA optimization experiment; (III) TBSA optimization experiment; (IV) FTBSA optimization experiment; (IV) DS optimization experiment. The experiments select 6 benchmark functions which are used commonly as the optimization algorithm from [15,16]. There are parameter setting, the functions name, formula, scale and dimension in Table 1.

Table 1. Benchmark functions used in experiments

| **Fun** | **Name of Fun** | **Formula** | **Scale** | **D** |
|---|---|---|---|---|
| F1 | SumPower | $f(x)=\sum_{i=1}^{n}\lvert x_i\rvert^{(i+1)}$ | $(-1,1)$ | 30,60 |
| F2 | Apline | $f(x)=\sum_{i=1}^{n}\lvert x_i\cdot\sin(x_i)+0.1\cdot x_i\rvert$ | $(-5.12,5.12)$ | 30,60 |
| F3 | SumSquare | $f(x)=\sum_{i=1}^{n}ix_i^2$ | $(-10,10)$ | 30,60 |
| F4 | Schwefel2.22 | $f(x)=\sum_{i=1}^{n}\lvert x_i\rvert+\prod_{i=1}^{n}\lvert x_i\rvert$ | $(-10,10)$ | 30,60 |
| F5 | Quartic | $f(x)=\sum_{i=1}^{n}ix_i^4$ | $(-1.28,1.28)$ | 30,60 |
| F6 | Exponential | $f(x)=\exp\left(0.5*\sum_{i=1}^{n}x_i\right)-1$ | $(-1.28,1.28)$ | 30,60 |

In Table 1, F2 is the multi-modal and multi-extreme function; F6 is the exponential function; the rest four test functions are all uni-modal functions. The difficulty of the optimization increases with the number of dimensions; in this paper, the dimensions of the search space are 30 and 60. The max iteration is 5000, and the number of the population is 100; all of the final results use the average value after 30 times independent running.

Performance evaluation uses the following methods: (1) fix the number of iterations and evaluate the convergence speed and precision of the algorithm; (2) fix the target value of convergence precision and evaluate the number of iterations that the algorithm needed to achieve the target value.

4.2. **Convergence performance in fixed iterations.** In this section, we experimented on six typical benchmark functions. The times of iterations are fixed at 5000 in the five tests, the test results are shown in Figure 1 when the dimension is 30, and the test results are shown in Figure 2 when the dimension is 60.

From the convergence curves of six benchmark functions, we can see that in terms of the convergence precision and speed of the algorithm, FBSA, TBSA and FTBSA algorithms are more greatly improved than BSA. Furthermore, BSA is easy to trap into the local

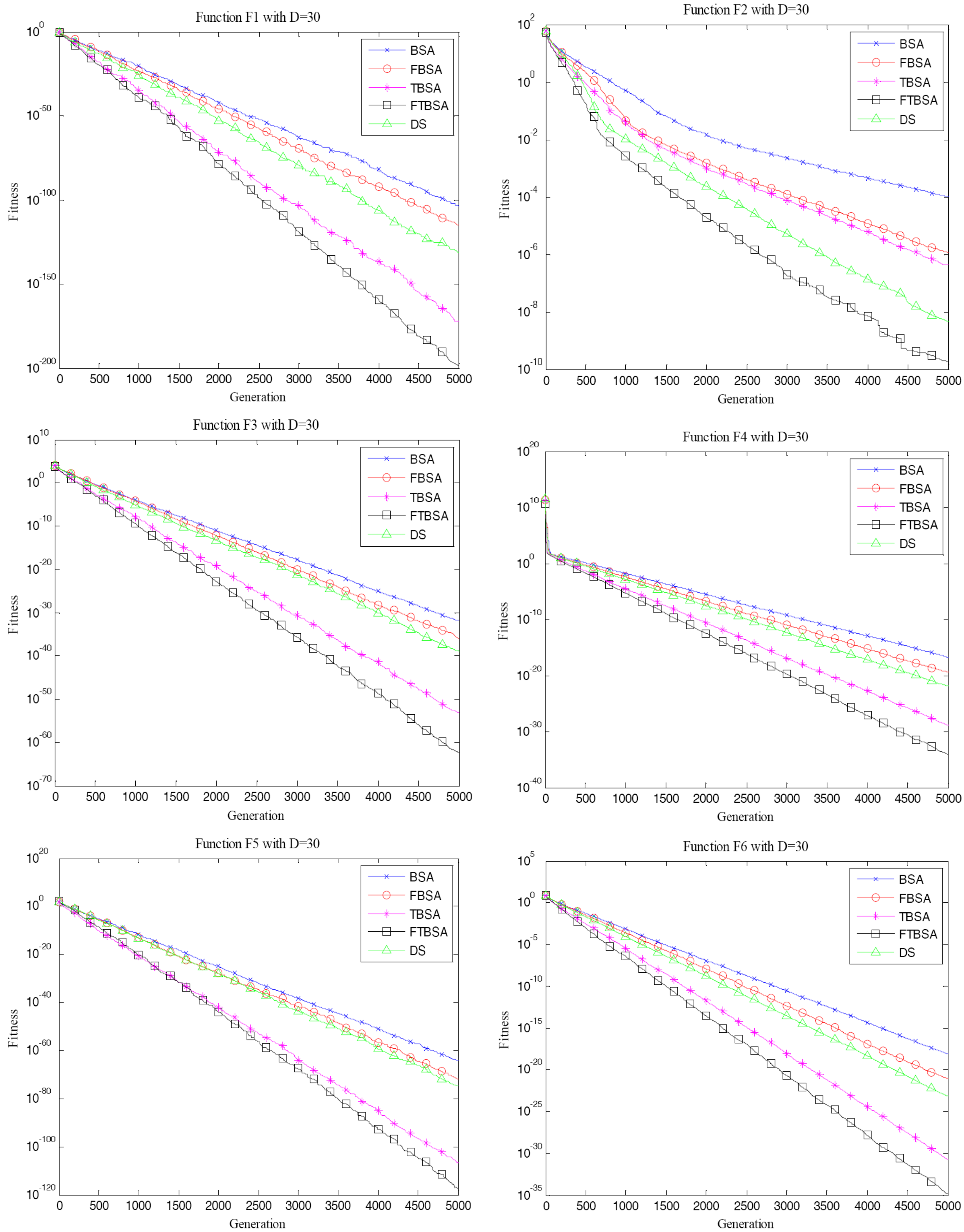


FIGURE 1. Evolutionary curves of five tests with 30 dimensions

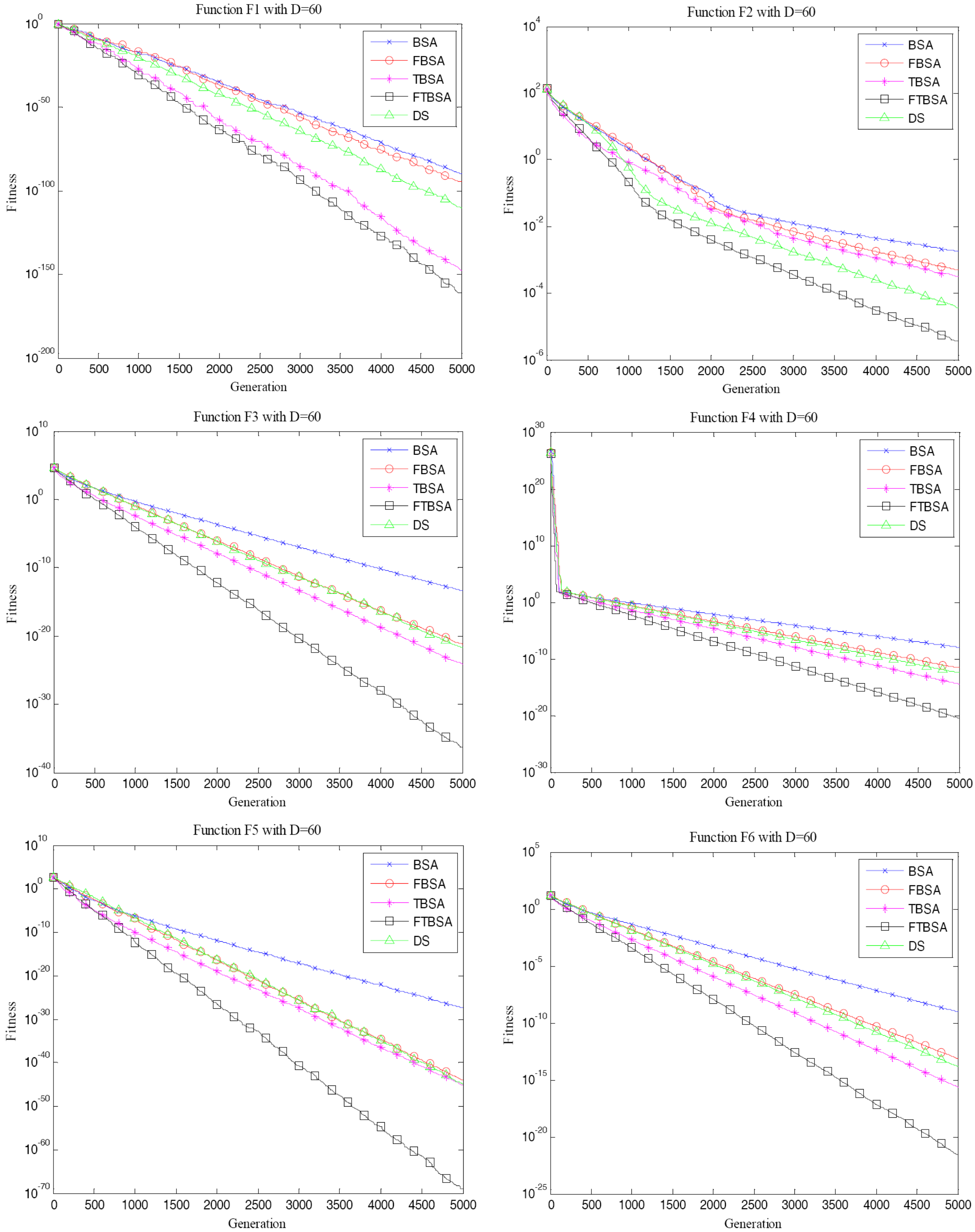


FIGURE 2. Evolutionary curves of five tests with 60 dimensions

optimum, and the proposed algorithms can solve this problem well. Among them, FTBSA has the highest convergence precision and the fastest convergence speed, and also has the better robustness and good operational performance. At the same time, this experimental result shows the effectiveness of the improved algorithms.

4.3. **Iterative number needed in fixed convergence precision.** In the following experiments, we used a fixed convergence precision $10^{-6}$. The population number is 100, dimension is 60 and the maximum number of iterations is 5000. After 30 running times,

Table 2. Iterative number after 30 times independent running at fixed precision

| Fun | D | Algorithm | Mean | Min | Max | SR |
|---|---|---|---|---|---|---|
| F1 | 60 | BSA | 257 | 177 | 380 | 1 |
| | | FTBSA | 178.6333 | 92 | 267 | 1 |
| | | DS | 268.0667 | 171 | 344 | 1 |
| F2 | 60 | BSA | 5000 | 5000 | 5000 | 0 |
| | | FTBSA | 4846.7 | 3982 | 5000 | 0.27 |
| | | DS | 4972.8 | 4331 | 5000 | 0.07 |
| F3 | 60 | BSA | 2711.5 | 2471 | 2892 | 1 |
| | | FTBSA | 1230.9 | 1069 | 1391 | 1 |
| | | DS | 1894.5 | 1713 | 2051 | 1 |
| F4 | 60 | BSA | 4010.5 | 3823 | 4224 | 1 |
| | | FTBSA | 1811.2 | 1639 | 2006 | 1 |
| | | DS | 2753.9 | 2559 | 3011 | 1 |
| F5 | 60 | BSA | 944.2667 | 838 | 1073 | 1 |
| | | FTBSA | 522.5667 | 402 | 636 | 1 |
| | | DS | 875.9000 | 706 | 1013 | 1 |
| F6 | 60 | BSA | 3370.1 | 3184 | 3562 | 1 |
| | | FTBSA | 1516.7 | 1372 | 1642 | 1 |
| | | DS | 2361.3 | 2245 | 2497 | 1 |

we can get the average value of the iterative number when the precision is reached. In this section, we calculated the Success Rate (SR) of the algorithms. Using $H$ represents the running times to reach the precision and $M$ represents the total number of iterations, $SR$ can be calculated according to the formula as Equation (8):

$$SR = \frac{H}{M} \tag{8}$$

Table 2 shows the value of mean, minimum, maximum and success rate of the algorithms.

From Table 2, we can see that for function Apline, BSA and DS cannot achieve the target precision; while FTBSA achieved 100% success rate. Furthermore, it needs less iterations to reach the target precision. The test results show that FTBSA has best optimization effect, and the results on six test functions are better than BSA and DS.

5. **Conclusion.** For BSA has the disadvantages of slow convergence speed and low precision, this paper proposed a new improved algorithm, FTBSA. FTBSA includes two kinds of strategies: new perturbation strategy and new selection strategy. Combining above strategies, FTBSA can greatly improve the convergence speed and precision. The simulation experiments on several test functions show that: improved algorithm has better convergence speed and precision in lower iterative number. In addition, the results show that FTBSA is not easy to fall into local optimum by comparing with the DS algorithm and BSA. FTBSA is new and may be a good alternative to deal with complex problems. Future works may focus on how to balance the exploring ability and the developing capability of the algorithm. In addition, we can also improve the speed of algorithm by realizing the parallel of the optimization algorithms on graphics processing units (GPU).

**Acknowledgement.** The work is supported by National Natural Science Foundation of China (No. 61401307), China Postdoctoral Science Foundation (No. 2014M561184), Tianjin Research Program of Application Foundation and Advanced Technology of China (No. 15JCYBJC17100), Science and Technology Commissioners Project of Tianjin (No.

16JCTPJC48400). The authors also gratefully acknowledge the helpful comments and suggestions of the reviewers, which have improved the presentation.

## REFERENCES

[1] R. Storn and K. Price, Differential evolution – A simple and efficient heuristic for global optimization over continuous spaces, *Journal of Global Optimization*, vol.11, no.4, pp.341-359, 1997.

[2] D. Karaboga and B. Basturk, A powerful and efficient algorithm for numerical function optimization: Artificial bee colony (ABC) algorithm, *Journal of Global Optimization*, vol.39, no.3, pp.459-471, 2007.

[3] R. Eberhart and J. Kennedy, A new optimizer using particle swarm theory, *Proc. of the 6th International Symposium on Micro Machine and Human Science*, pp.39-43, 1995.

[4] P. Civicioglu, Transforming geocentric cartesian coordinates to geodetic coordinates by using differential search algorithm, *Computers & Geosciences*, vol.46, no.3, pp.229-247, 2012.

[5] D. Wang, Colony location algorithm for assignment problems, *Journal of Control Theory and Applications*, vol.2, no.2, pp.111-116, 2004.

[6] N. T. Linh and N. Q. Anh, Application artificial bee colony algorithm (ABC) for reconfiguring distribution network, *The 2nd International Conference on Computer Modeling and Simulation*, pp.102-106, 2010.

[7] C. Xu, H. Duan and F. Liu, Chaotic artificial bee colony approach to uninhabited combat air vehicle (UCAV) path planning, *Aerospace Science and Technology*, vol.14, no.8, pp.535-541, 2010.

[8] Y. X. Zhang, X. M. Tian and X. G. Deng, Blind source separation based on modified artificial bee colony algorithm, *Tien Tzu Hsueh Pao/Acta Electronica Sinica*, vol.40, no.10, pp.2026-2030, 2012.

[9] J. J. Liang, A. K. Qin, P. N. Suganthan and S. Baskar, Comprehensive learning particle swarm optimizer for global optimization of multimodal functions, *IEEE Trans. Evolutionary Computation*, vol.10, no.3, pp.281-295, 2006.

[10] Y. Li, Z.-H. Zhan, S. Lin, J. Zhang and X. Luo, Competitive and cooperative particle swarm optimization with information sharing mechanism for global optimization problems, *Information Sciences*, vol.293, no.1, pp.370-382, 2015.

[11] K. Price, R. M. Storn and J. A. Lampinen, *Differential Evolution: A Practical Approach to Global Optimization (Natural Computing Series)*, Springer-Verlag New York, Inc., 2005.

[12] P. Civicioglu, Backtracking search optimization algorithm for numerical optimization problems, *Applied Mathematics and Computation*, vol.219, no.15, pp.8121-8144, 2013.

[13] A. E. Chaib, H. R. E. H. Bouchekara, R. Mehasni and M. A. Abido, Optimal power flow with emission and non-smooth cost functions using backtracking search optimization algorithm, *International Journal of Electrical Power & Energy Systems*, vol.81, pp.64-77, 2016.

[14] D. Karaboga and E. Kaya, An adaptive and hybrid artificial bee colony algorithm (aABC) for ANFIS training, *Applied Soft Computing*, vol.49, pp.423-436, 2016.

[15] E. Nabil, A modified flower pollination algorithm for global optimization, *Expert Systems with Applications*, vol.57, pp.192-203, 2016.

[16] M. J. Mahmoodabadi and A. R. Nemati, A novel adaptive genetic algorithm for global optimization of mathematical test functions and real-world problems, *Engineering Science and Technology, an International Journal*, vol.19, no.4, pp.2002-2021, 2016.