Novelty Search-based Bat Algorithm adjusting Direction and Distance between Solutions for Multimodal Optimization

ABSTRACT

This paper proposes two methods: k-nearest neighbor bat algorithm (k-NNBA), and novelty search-based bat algorithm (NSBA) for finding global and local minima of multimodal function. Multimodal optimization aims to find solutions which are located global and local peaks on the various benchmark problems. However, conventional methods like as particle swarm optimization (PSO) and differential evolutionary (DE), tend to straightforward converge global optimum or worse-fitness value. To solve multimodal optimization problems, we have to consider the balance of performance which unites global and local search. Bat algorithm (BA) is one of these algorithms has this balance using characteristic of echolocation. BA behavior is to shift global to local search as increasing iteration. Using this behavior, we proposes k-NNBA for letting solutions move to sparse area in search domain, and NSBA for keeping distance between each solution, and validates the performance of them. The result revealed two things: (i) k-NNBA and NSBA are enable solutions to spread widely and keep from each other without converging a global optimum or a high-fitness minimum; (ii) NSBA is effective algorithm to distribute solutions despite changes of neighbors and adaptable for any multimodal functions.

CCS CONCEPTS

• **Mathematics of computing → Bio-inspired optimization**;

KEYWORDS

Bat Algorithm, Multimodal function, Metaheuristic Optimization

ACM Reference Format:

. 2018. Novelty Search-based Bat Algorithm adjusting Direction and Distance between Solutions for Multimodal Optimization. In *Proceedings of the Genetic and Evolutionary Computation Conference 2018 (GECCO '18).* ACM, New York, NY, USA, 8 pages. https://doi.org/

1 INTRODUCTION

In the past couple of decades, metaheuristic algorithms became major method for optimization problem. Generally, they are based on biological evolution in nature-inspired system. These various methods are adaptable for a specific situation using non-linear objective functions. Particle Swarm Optimization (PSO) which is one of metaheuristic algorithms, modeled fish swarm if a fish find a global optimum, the other fishes converge to the fish [2]. Meanwhile, there is another algorithm called Firefly Algorithm (FA), which is particularly well to local search with flashing light of fireflies [7]. In two fireflies, a brighter firefly attracted the other one. Although these algorithms are widely used for optimization problem, the performance of search considerably depends on the scale and contour of multimodal functions. To adapt any optimization problem, we have to consider the balance of performance which unites global search and local search. Bat algorithm (BA) is one of bio-inspired algorithms for both search with characteristic of echolocation [8]. The behavior of bat is determined echolocation which is sound wave for finding food in the dark. Ideally All bats move to a bat which found food or prey, with loudness and pulse rate to sense the distance each other. Simultaneously, some of them fly randomly for searching the other prey globally. After finding a prey, they will drop loudness down and raise pulse rate up automatically, for adjusting to search spatial domain. However, the performance of global search is still higher than local search, bat algorithm is easily fallen global minimum or high-fitness value on multimodal problems. For this reason, we propose distributed BA (k-NNBA and NSBA) for migrating solutions away and keeping distance each other. Besides for the performance measurement, we set different changes: (a) for guiding local search using personal best solution or previous position of solution instead of global best solution; (b) existence or nonexistence of a new solution generated by flying randomly. For solving multimodal optimization, there are the other approaches that based on Genetic Programming(GP) [5], PSO [1][3]and Differential Evolutionary(DE) [4][6]. However, for practical optimization problems, it is desirable to use multimodal functions to reach the peak of both global optima and local minima located, with small population size.

This paper is composed of 7 chapters. After introduction, 2nd chapter and 3rd chapter demonstrate Bat Algorithm of prototype and proposed. 4th chapter about Multimodal function , after that experiment including the result are followed in 5th chapter, and we discuss the result in 6th chapter. Finally, conclusion is mentioned.

2 BAT ALGORITHM

BA based on echolocation behavior of microbat uses frequency and loudness for adaptive global search on a multimodal function. In this algorithm, loudness A^0 is used as a parameter to adjust frequency. When microbat moves toward target, loudness A^0 is also gradually decreased in proportion to travel distance of microbat decreases. Behavior of microbat is consists of following three rules:

- Each bat measures the distance between own location and target using frequency f_i.
- On the location x_i, bat with velocity v_i moves to another bat closed target randomly.
- Loudness A^0 varies to sense how far approaching toward the target.

GECCO '18, July 15–19, 2018, Kyoto, Japan

ACM ISBN 978-x-xxxx-xxxx-x/YY/MM.

Each bat with velocity v_i, location x_i, and frequency f_i is updated as follows:

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

$$d_i^{t-1} = x_* - x_i^{t-1} \tag{2}$$

$$v_i^t = v_i^{t-1} + d_i^{t-1} * f_i \tag{3}$$

$$x_i^t = x_i^{t-1} + v_i^t \tag{4}$$

Velocity v_i varies by tuning frequency f_i from $[f_{max}\ f_{min}]$ as $f_{max} = 1$ and $f_{min} = 0$. β is uniform random distribution from 0 to 1. Firstly in global search step, BA calculates the distance from all bats position to current global best solution x_*, when population is generated. Afterward, each bat moves to location x_i with velocity v_i toward global best solution, as shown in Fig. 1. Secondly in local search step, generates a new solution x_{new} around global best solution, as shown 2nd phase of Fig. 1. The equation as below

$$x_{new} = x_* + \epsilon A^t\ , \tag{5}$$

where ϵ is uniform random distribution between $[0\ \ 1]$. A^t is the average loudness of all bats. Initialized all bats start searching target using loudness A_i and the reflect wave as pulse emission rate r_i. Loudness and pulse rate are updated as follows:

$$A_i^{t+1} = \alpha A_i^t \tag{6}$$

$$r_i^{t+1} = r_i^0[1 - exp(-\gamma t)] \tag{7}$$

Both of them are also updated when new solution is updated by equation (5) for each iteration, as shown final phase of Fig. 1. Loudness gradually decreases as approaching to target, pulse rate increases in contrast. BA initializes pulse rate as a uniform random distribution r_i^0 between [0 1] or a number closed around zero. α and γ are symbolized damping coefficient. In simulated experiment, they are set $\alpha = \gamma = 0.9$. The pseudo code and the process of BA presented as below (shown in Fig. 1 & Algorithm 1).

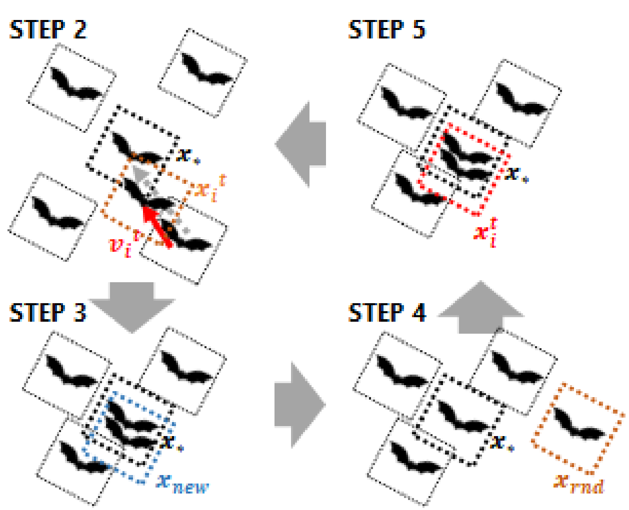


Figure 1: Bat Motion of BA

- STEP1: Initialize population of bats (line 1 to 3)
  Initialize location $x_i(i = 1, 2, ..., n)$ with velocity $v_i(i = 1, 2, ..., n)$ randomly. Each bat has loudness A_0, parse rate r_i and frequency f_i as initial value.
- STEP2: Generate new solutions (line 6 to 7)
  Generate new solutions x_i^t based on equation (4).
- STEP3: In local search phase, Generate a new solution around global best solution x_* (line 8 to 12)
  In case of a random distribution higher than parse rate r_i, generate a new solution x_{new} around x_*.
- STEP4: Generate a new solution randomly (line 13)
  Generate a new solution x_{rnd} by random generation of bat.
- STEP5: Rank and update solutions (line 14 to 17)
  In case of $rand < A_i$, choose the best from all solutions which are x_i, x_{new}, and x_{rnd}, and cross over as personal best solution unless it is higher than the value of former iteration.
- STEP6: Loop to STEP2

Algorithm 1 Bat Algorithm

Require: *Objective Function* $f(x)$
1: Initialize Population $x_i(i = 1, 2, ..., n)$ and v_i
2: Define frequency f_i at location x_i
3: Initialize pulse rates r_i, and loudness A_i
4: **while** ($t <$ Max number of iterations) **do**
5:     **for** i=1 to n **do**
6:         Generate new solutions x_i by tuning frequency f_i
7:         Update location x_i and velocity v_i [eqs.(1) to (4)]
8:         **if** ($rand > r_i$) **then**
9:             Generate a new solution x_{new} around global best solution x_i [eq.(5)]
10:         **else**
11:             Continue
12:         **end if**
13:         Generate a new solution x_{rnd} randomly
14:         **if** ($rand < A_i \& f(x_i), f(x_{new}), f(x_{rnd}) < f(x_*)$) **then**
15:             Accept the new solution, and update pulse rate r_i & loudness A_i [eqs. (6)(7)]
16:         **end if**
17:         Evaluate the all bats and select a best solution x_* in the current solutions
18:     **end for**
19: **end while**

3 DISTRIBUTED BAT ALGORITHM

For reaching peaks of local minima and global minima located, we have to make bats spread widely. In k-nearest neighbor bat algorithm (k-NNBA), we focus on difference between the number of population. In Novelty Search-based bat algorithm (NSBA), we consider as written the difference above, and distance of each bat.

3.1 k-Nearest Neighbor Bat Algorithm

k-nearest neighbor (k-NN) method is used for classification for data with discrete label basically. The mechanism of k-nearest neighbor is to find a new object (a new point) with closest distance between the other objects around it, and predict discrete label from these factors. Here, we use a new object as a new solution with the distance for keeping each bat away. The distance equation is written as below.

$$d_i^{t-1} = \frac{1}{K}\sum_{j=1}^{K}(x_{i*} - x_j^{t-1}) \tag{8}$$

$$d_i^{t-1} = \frac{1}{K}\sum_{j=1}^{K}(x_i^{t-1} - x_j^{t-1}) \tag{9}$$

K describes the number of nearest neighbor. In equation (11), x_{i*} means personal best solution. k-NN is very simple method and is easy to implement, but depending on number of neighbors, we have to choose proper k. Pseudo code is described in Algorithm 2.

3.2 Novelty Search-based Bat Algorithm

3.2.1 Novelty Search. Novelty search is used as evolutionary search approach to expand dense solutions into sparse area and to measure the distance between current solutions to reward or delete it. The sparseness of solutions is calculated as below,

$$\rho(x) = \frac{1}{k}\sum_{i=0}^{k} dist(x, \mu_i), \tag{10}$$

where the sparseness $\rho(x)$ at a point x shows the scatter of solutions. The dist in k-nearest neighbors is the average distance between the point x and μ_i, which is the ith nearest neighbor of x. This is an example in case of k neighbor = 3 (shown in Fig. 2). It describes that a solution is migrated away from three neighbors.

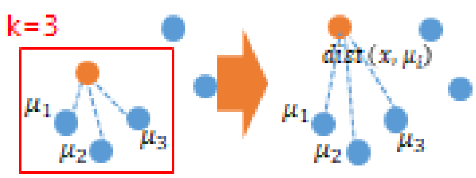


Figure 2: distributed a solution to sparse area

3.2.2 Novelty Search-based Bat Algorithm. In order to adapt multimodal optimization not only single objective optimization, Novelty Search-based Bat Algorithm (NSBA) enables all population to reach local minima. This paper proposes a method of keeping over a certain distance between each location of bat, and letting population remain around local minima. Using this behavior, all population are updated by the equation as bellow,

$$d_i^{t-1} = \frac{1}{K}\sum_{j=1}^{K}\frac{(x_{i*} - x_j^{t-1})}{|x_{i*} - x_j^{t-1}|^2} \tag{11}$$

$$d_i^{t-1} = \frac{1}{K}\sum_{j=1}^{K}\frac{(x_i^{t-1} - x_j^{t-1})}{|x_i^{t-1} - x_j^{t-1}|^2} \tag{12}$$

where K is population size of nearest neighbor, and x_{i*} indicates personal best solution. x_i^{t-1} is previous position of solution. In addition, bats with velocity v_i^t and location x_i^t are updated same as (3) and (4) of conventional method. Used distance function in Novelty search describes scalar equation. However in this proposes, we alter scalar to vector equation for determining search direction.

3.2.3 Distance of each bat. Above-mentioned the vector equation 8 and 9, as distance of each bat is closer, they hardly move to sparse area. Conversely, as they located far away each other, they move greatly up to a boundary of search area. To control this movement, we introduce the denominator as equation (11)(12). Here is the Algorithm flow on global minimum optimization. The NSBA pseudo code is described in Algorithm 2.

- STEP1: Initialize population of bats (line 1 to 3)
  Initialize location $x_i (i = 1, 2, ..., n)$ with velocity $v_i (i = 1, 2, ..., n)$ randomly. Each bat has loudness A_0, parse rate r_i and frequency f_i as initial value.
- STEP2: Generate new solutions (line 6 to 7)
  Generate new solutions x_i^t based on equation (3)(4) with (12) or (11).
- STEP3: In local search phase, Generate a new solution around solutions x_i (line 8 to 12)
  In case of a random distribution higher than parse rate r_i, generate a new solution x_{local} around x_i.
- STEP4: Generate a new solution randomly (line 13)
  Generate a new solution x_{rnd} by random walk of bat.
- STEP5: Rank and update solutions (line 14 to 18)
  If $rand < A_i$, choose the best from all solutions which are x_i, x_{local}, and x_{rnd}. After that, cross over as personal best solution unless it is higher fitness value than previous iteration.
- STEP6: Loop to STEP2

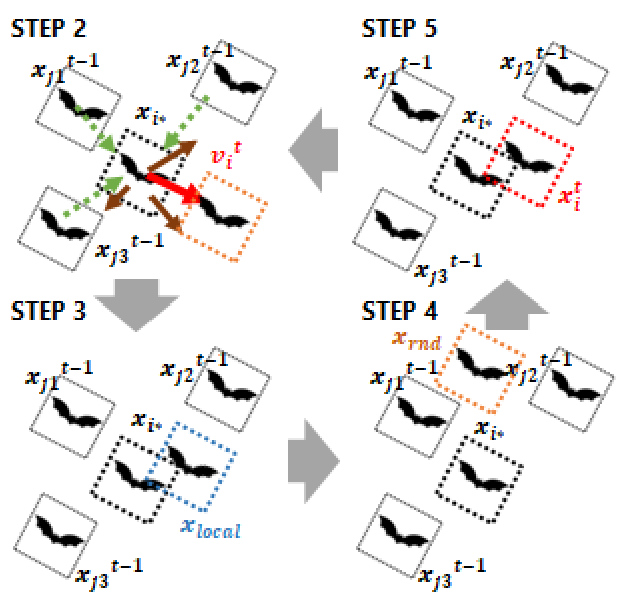


Figure 3: Bat motion of NSBA

Algorithm 2 Distributed Bat Algorithm

Require: *Objective Function* $f(x)$

Initialize Population $x_i (i = 1, 2, ..., n)$ and v_i
Define frequency f_i at location x_i
Initialize pulse rates r_i, and loudness A_i
while ($t <$ Max number of iterations) **do**
  for i=1 to n **do**
    Generate new solutions x_i by tuning frequency f_i
    Update location x_i, velocity v_i [eqs.(1)(3)(4)] and k-NNBA for [eq.(8)(9)] NSBA for [eq.(11)(12)]
    if ($rand > r_i$) **then**
      Generate a new solution x_{local} around the solution x_i [eq.(5)]
    else
      Continue
    end if
    Generate a new solution x_{rnd} randomly (or without x_{rnd})
    if ($rand < A_i \& f(x_i) < f(x_i)$) **then**
      Accept the new solution, and update pulse rate r_i & loudness A_i [eqs. (6)(7)]
    end if
  end for
  Evaluate the all bats and select a best solution x_i* in the current solutions
end while

3.3 Comparion with k-NNBA and NSBA

we compare 8 methods in total. There are comparison of these methods on below Table 1.

Table 1: Comparion with k-NNBA & NSBA

x_{rnd}	$\circ$		$\times$	
x	x_{i*}	x_i^{t-1}	x_{i*}	x_i^{t-1}
k-NNBA	I	II	III	IV
NSBA	V	VI	VII	VIII

4 MULTIMODAL FUNCTION

In the contour of function, there are the coordinate that horizontal axis is x1 and vertical axis is x2, and the colorbar that color density describes the fitness value shown as Fig. 5. As color becomes darker area, fitness value gets lower. For validating NSBA to distribute spread widely, there are some multimodal functions. Focused on depth of fitness value, scale of multimodal domain and number of local minima, we used these functions as following section.

4.1 Griewank Function

As an example to demonstrate the bat motion of this algorithm, we use Griewank function as below (shown in Fig. 4(a))

$$f(x) = \sum_{i=1}^{d} \frac{x_i}{4000} - \prod_{i=1}^{d} \cos(\frac{x_i}{\sqrt{i}}) + 1, \qquad (13)$$

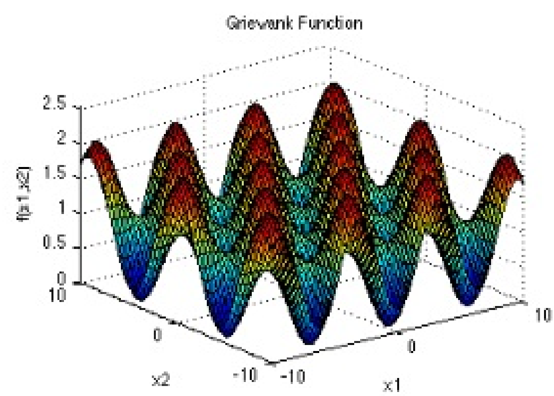
where global optimum is $f(x_*) = 0$, at $x_* = [0 \;\; 0]$. There are 17 local minima at $\pm x \approx [6.2800 \;\; 8.8769]$, $[3.1400 \;\; 4.4385]$, $[0 \;\; 8.8769]$, $[6.2800 \;\; 0]$, $[9.4200 \;\; 4.4385]$ in the range of this function is between $-10 \le x_i \le 10$ with i=1,2,...,d. The function $f(x)$ has global minimum $f(x_*) = 0$ and also the other local minima $f(x_{i*}) \approx 0$ for $d = 2$.

4.2 Rastrigin Function

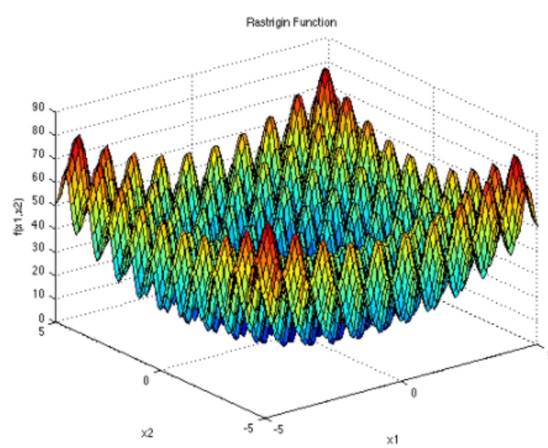
This function has 121 local minima in the spatial domain, at $\pm x = [0, ..., 11 \;\; 0, ..., 11]$. And global minimum is $f(x_*) = 0$ at $x = [0 \;\; 0]$. The function equation is

$$f(x) = 10d + \sum_{i=1}^{d} [x_i^2 - 10\cos(2\pi x_i)] \qquad (14)$$

3D model and contour of this function are showed in Fig. 4(b) & 5(b).

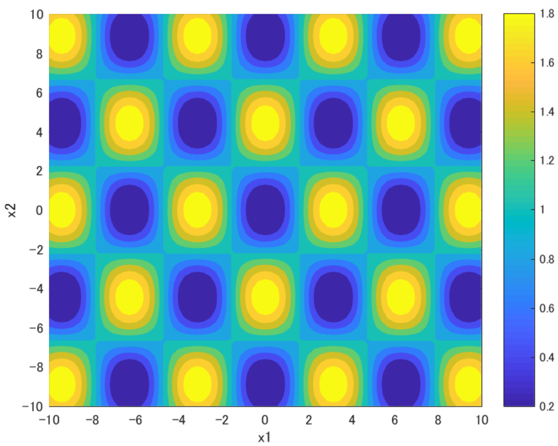


(a) Griewank function

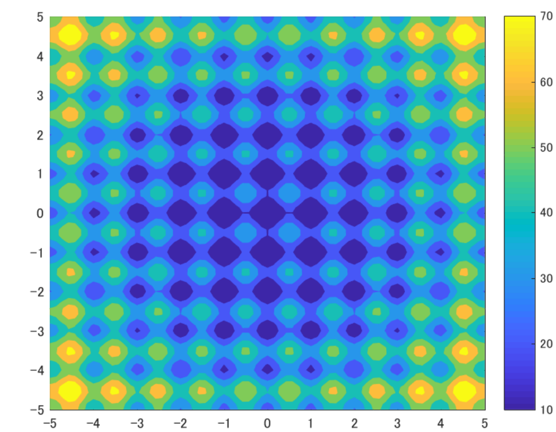


(b) Rastrigin function

Figure 4: 3D model Function



(a) Griewank function



(b) Rastrigin function

Figure 5: Contour of Functions

5 EXPERIMENT

We compared proposed NSBA with the other algorithms, which NNBA and BA. Each algorithm was run for 10 seeds to validate the performance of NSBA. In this paper, the algorithm is implemented on MATLAB for the benchmark function.

5.1 Evaluation Criteria

dist is total amount of the distance between local minima and nearest neighbor population, in case of initializing population randomly each algorithm. In this experiment, we focus on how many found local minima, and *dist* which total amount of the distance between local minima and the closest solutions, as below.

$$dist = \sum_{i=1}^{M} min_{j \in N} |s_i - x_j|, \tag{15}$$

where M is maximum number of local minimum, and N is population size of bats. s_i means the coordinate of local minimum. As *dist* is closed zero, the number of bats located local minima increases. We compare with the performance of these algorithms in term of the population size and the bat behavior by iteration.

5.2 Experimental Parameter

All experiments use same parameters, where population size $N = 20$, frequency $f_{max} = 1$, $f_{min} = 0$, loudness $A^0 = 1$, parse rate $r^0 \in [0 \;\; 1]$ with $\alpha = \gamma = 0.9$.

5.3 Result

5.3.1 Comparison with I and V. On griewank function, dist of k-NNBA and NSBA are nearly same in any neighbors, but k-NNBA is slightly better performance than NSBA on each function. From rastrigin function, k-NNBA is smaller than NSBA in any neighbors.

5.3.2 Comparison with II and VI. On griewank function, NSBA is almost better than k-NNBA in each neighbor except for K=4, dist of k-NNBA is a bit smaller. Overall, *dist* is higher than the other methods on griewank function. In rastrigin function, k-NNBA gradually increased. However, NSBA slowly decreased until K=16.

5.3.3 Comparison with III and VII. Method III and VII are better performance than the other mothods in griewank function. However, k-NNBA became worse as increasing neighbors. By contrast, NSBA was very unchanged in any neighbors. Besides, averages of k-NNBA and NSBA almost unchanged.

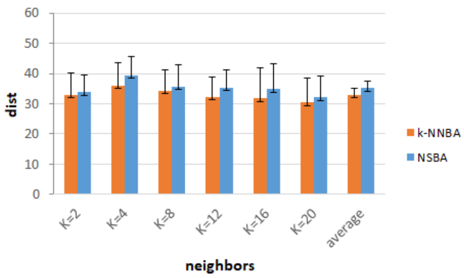
5.3.4 Comparison with IV and VIII. The dist of NSBA is smaller than k-NNBA in K=2 to 20 on both functions, except for K=2 and 4 on rastrigin function. The average of NSBA also smaller than k-NNBA. Comparison to NSBA in Fig. 8, dist of k-NNBA is lowest of the other number of neighbors. However, dist of k-NNBA rose up gradually from K=4. By contrast, dist of NSBA remains fairly on griewank function, but the performance gets worse after K=4. Overall,

6 DISCUSSION

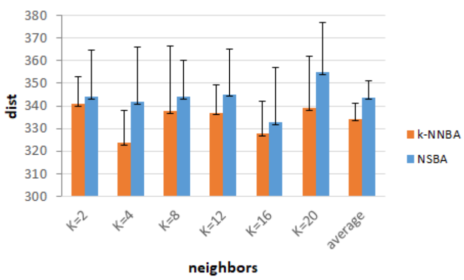
There are line graphs for all methods on Griewank function. The horizontal axis describes iteration of evaluating solutions, and vertical axis describes the sum of distance between each local minima and the closest solution shown as Fig. 10 to 13. Besides distributed solutions at 1000th iteration step from Fig. 14 and 15 for rastrigin function from Fig. 16 and 17 for griewank function.

6.1 k-NNBA vs NSBA

From Fig. 7 to 9, k-NNBA tends to decline as neighbors increase. NSBA is hardly affected by changes of neighbors, so that it performed better than k-NNBA relatively. Especially in k=4 from Fig. 14, 15 and 17, k-NNBA more distributed than NSBA obviously. It means that equation of considering distance still performed weakly. For this reason, we have to adjust the number of neighbors and population size and updating equation.

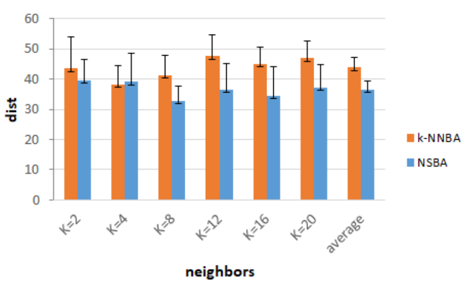


(a) Griewank Function

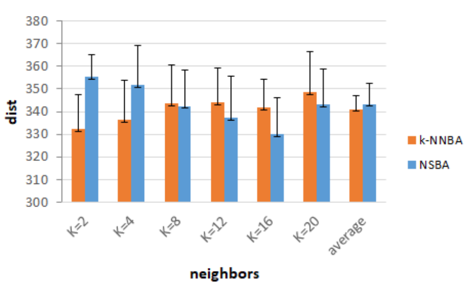


(b) Rastrigin Function

Figure 6: Comparison with method I & V

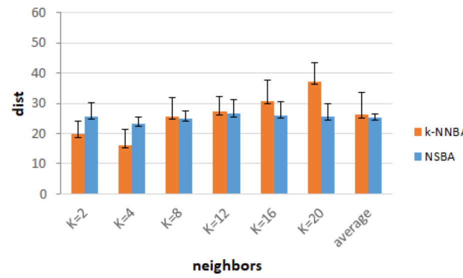


(a) Griewank Function

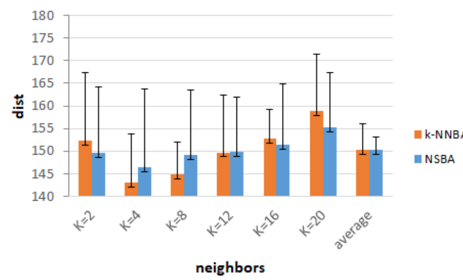


(b) Rastrigin Function

Figure 7: Comparison with method II & VI

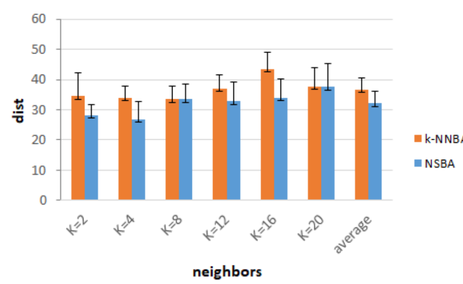


(a) Griewank Function

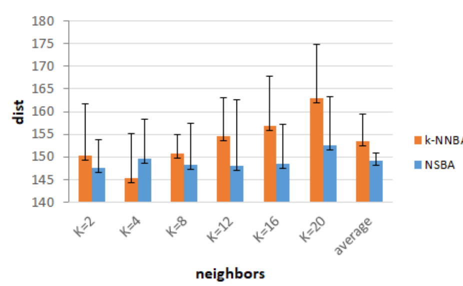


(b) Rastrigin Function

Figure 8: Comparison with method III & VII

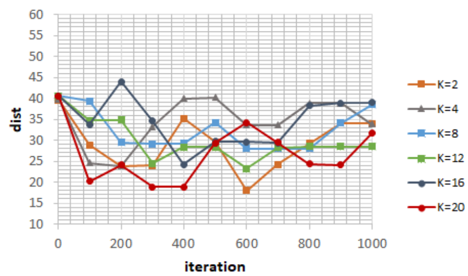


(a) Griewank Function

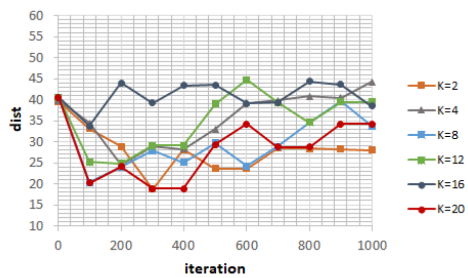


(b) Rastrigin Function

Figure 9: Comparison with method IV & VIII

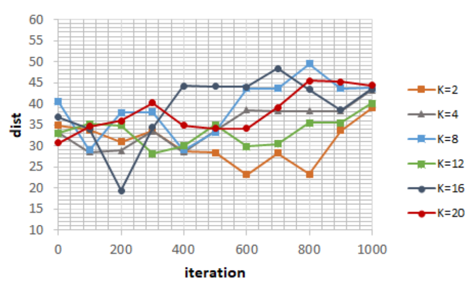


(a) k-NNBA

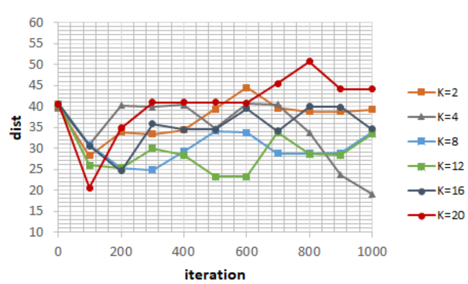


(b) NSBA

Figure 10: Comparison with method I & V

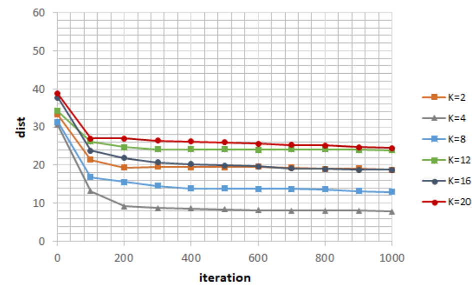


(a) k-NNBA

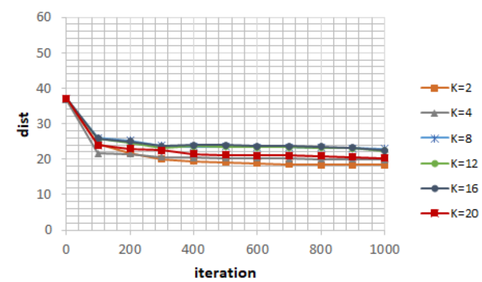


(b) NSBA

Figure 11: Comparison with method II & VI

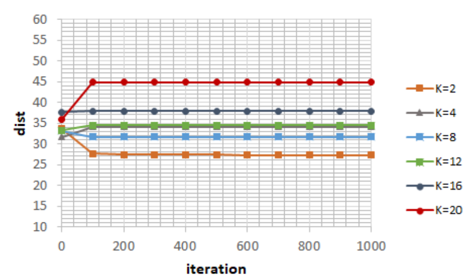


(a) k-NNBA

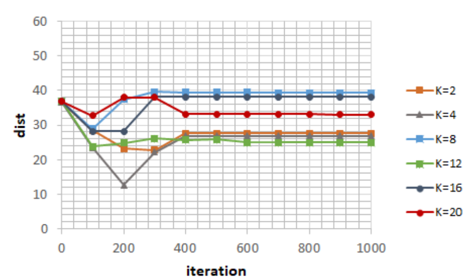


(b) NSBA

Figure 12: Comparison with method III & VII

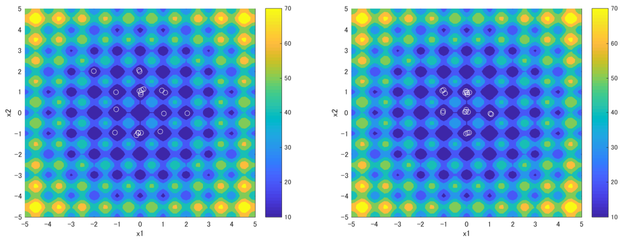


(a) k-NNBA



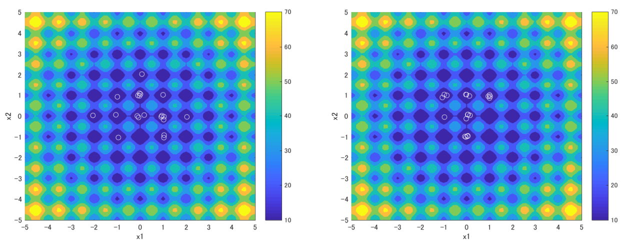
(b) NSBA

Figure 13: Comparison with method IV & VIII



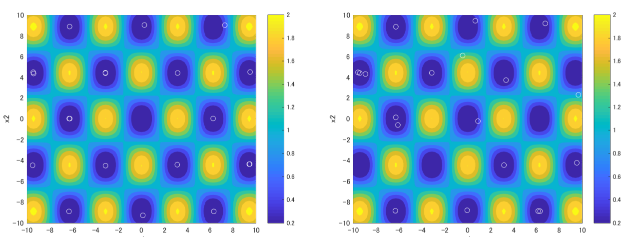
(a) k-NNBA (b) NSBA

Figure 14: method I & V (K=4)



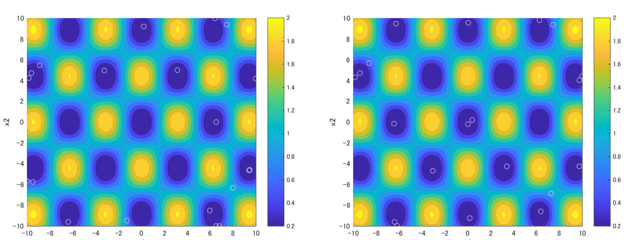
(a) k-NNBA (b) NSBA

Figure 15: method II & VI (K=4)



(a) k-NNBA (b) NSBA

Figure 16: method III & VII (K=4)



(a) k-NNBA (b) NSBA

Figure 17: method IV & VIII (K=4)

6.2 Existence or nonexistence of x_{rnd}

Compared line graph with Fig. 10 & 11 and Fig. 12 & 13, x_{rnd} affected iteration step. Especially in Fig. 10 & 13, *dist* fluctuated until 1000 iteration steps in any neighbors. By contrast in Fig. 12 & 13, *dist* of any neighbors became stable over a certain iteration.

6.3 Differences in x_i^{t-1} and x_{i*}

Focused on 4 line graphs in right side Fig. 12 and 13 without s_{rnd}, x_{i*} indicates personal best solution, these *dist* fell continuously until 300 iterations and became stable to the end. From left side in Fig. 10 & 11, all *dist* fluctuated constantly, as x_{rnd} has strong effect on iteration.

7 CONCLUSION

We validated the performance of proposed bat algorithms for k-nearest neighbor and novelty search with changes of updating solutions and generating a new solution randomly. As a result, both algorithms performed for reaching local minima with global optimum. Especially the method using personal best without s_{rnd}, performed better than the other proposed methods. However, we have to adjust the number of neighbors for feasible multimodal functions. As population size of bat increases, the number of searched local minima also increased. Our future prospects are adapting this algorithm for the other benchmark functions, and blushing up the performance to cover unspecified large number of local minima. Future experiments on the other multimodal functions and investigation will be studied.

ACKNOWLEDGMENTS

REFERENCES

[1] P. Suganthan B. Y. Qu and S. Das. 2013. A Distance-Based Locally Informed Particle Swarm Model for Multimodal Optimization. *IEEE Transactions on Evolutionary Computation* 17, 3 (June 2013), 387–402.

[2] R. C. Eberhart and Kennedy. 1995. A New Optimizer Using Particle Swarm Theory. *Proc. Sixth International Symposium on Micro Machine and Human Science (Nagoya, Japan), IEEE Service center, Pis-cataway, NJ* 1 (1995), 39–43.

[3] C. G. Heo J. K. Kim H. K. Jung J. H. Seo, C. H. Lim and C. C. Lee. 2006. Multimodal Function Optimization Based on Particle Swarm Optimization. *IEEE Transactions on Magnetics* 42, 4 (April 2006), 1095–1098.

[4] X. Li. 2005. Efficient differential evolution using speciation for multimodal function optimization. *GECCO Proceedings of the 7th annual conference on Genetic and evolutionary computation* (2005), 873–880.

[5] T. Harada S. Yoshida and R. Thawonmas. 2017. Multimodal Genetic Programming by Using Tree Structure Similarity Clustering. *IEEE 10th International Workshop on Computational Intelligence and Applications, (Hiroshima, Japan)* (November 2017).

[6] R. Thomsen. 2004. Multimodal Optimization using crowding-based differential evolution. *IEEE Congress on Evolutionary Computation* 2 (2004), 1382–1389.

[7] X. S. Yang. 2009. Firefly Algorithms for Multimodal Optimization. *in:Stochastic Algorithms: Foundations and Applications, SAGA* 5792 (2009), 169–178.

[8] X. S. Yang. 2010. A Metaheuristic Bat-Inspired Algorithm. *in: Nature Inspired Cooperative Strategies for Optimization(NICSO 2010), Springer, Berlin* 284 (2010), 65–74.