Accelerated ABC (A-ABC) Algorithm for Continuous Optimization Problems

Ahmet Ozkis and Ahmet Babalik

Abstract—In this paper, accelerated artificial bee colony (A-ABC) method is presented. In A-ABC, two modifications are used on the Artificial Bee Colony (ABC) algorithm to progress its local search ability and convergence speed. Modifications are called as modification rate (MR) and step size (SS). In this study, we aim to investigate effects of using MR value and SS modification. Performances of the A-ABC and standard ABC are compared on well-known 7 different benchmark functions. Results show that A-ABC has generally better performance and faster convergence than standard version of the ABC algorithm.

Index Terms—ABC algorithm, meta-heuristic algorithms, swarm intelligence, optimization technique.

I. INTRODUCTION

Popularity of meta-heuristic algorithms has increased in recent years. Lots of modern meta-heuristic algorithms [1]-[4] have been developed for solving different optimization problems. They include population based, iterative based, stochastic based approaches. One of the most important meta-heuristic algorithms is the artificial bee colony (ABC) algorithm in literature.

Karaboga proposed the ABC algorithm in 2005 by examining intelligent behavior of honey bees. Bees exhibit some intelligent behavior while they perform their tasks such as foraging, navigation and task selection. The foraging mechanism as one of these tasks is more significant because the ABC algorithm was improved by mimicking food searching behavior of honey bees. In the mechanism of foraging, honey bees are divided into three groups depending on their task; that is, employed, onlooker and scout bees. Employed bees fly on a source and exploit that source, while onlooker bees wait in the hive and look for a nectar source to take advantage of the information given by the employed bees. The scouts randomly search the environment to find a new source based on either internal motivation or possible external clues [5]. In foraging, intelligent behaviours of honey bees can be summed up as follows [15]:

1) The first positions of food sources are determined randomly by scouts in the search area.
2) Scouts turn to employeds and try to exploit the discovered source. The employed bees return to the hive and pour the nectar brought from the food source. After the pouring process, employeds share the information about quality of the food source by dancing on the dance area and return the food sources that belong to themselves. When a source is exhausted, the bee which exploiting that source becomes a scout bee and find randomly a new food source.
3) Onlookers wait in the hive and watch the dances of the employeds. Onlookers determine the source they will exploit with regard to the dancing frequency of the employeds. This frequency points the quality of the food source.

In the ABC algorithm each food source has just one employed bee. So, number of employed bees and food source is equal. Onlooker bees and scout bees are described as unemployed bees. The main process of the standard ABC algorithm is given below [15]:

A. Generating Initial Food Source Position

The first positions of the food sources are generated
randomly depending on the boundary values of the parameters in the search space by the scouts.

\[ x_j = x_j^{\text{min}} + \text{rand}(0, 1)(x_j^{\text{max}} - x_j^{\text{min}}) \]  

(1)

where \( i = 1 \ldots SN \), \( j = 1 \ldots D \). SN is the number of sources and \( D \) is the number of optimization parameters. After determination of the food sources, scouts turn to employeds and onlooker bees along with employeds try to progress existing food source by exploiting. The stopping criteria of the algorithm is its reaching maximum number of function evaluations (MaxFES).

### B. Exploitation Mechanism of Employed Bees

The exploitation mechanism implements local searching by using employed and onlooker bees near the food sources. Each employed and onlooker bee tries to develop the quality of nectar by searching the neighborhood surrounding the source area. Employed and onlookers use Eq. (2) for the exploitation process.

\[ v_j = x_j + \theta_j(x_j - x)^{\text{max}} \]  

(2)

where \( \theta_j \) describes the existing position of the food source and \( v_j \) is a candidate source formed by modifying one parameter randomly selected from \( x_j \). \( J \) is a random integer between \([1, D]\) and \( k \) is another random integer between \([1, SN]\), \( k \) must be different from \( i \) value. \( \theta_j \) is a real random number in the range \([-1, 1]\).

If the value of changing parameter of \( v_j \) exceeds the boundary of search space, this parameter is set to boundary value. This process is shown in Eq. (3).

\[ v_j = \begin{cases} x_j^{\text{min}}, & v_j < x_j^{\text{min}} \\ x_j^{\text{max}}, & v_j > x_j^{\text{max}} \\ v_j, & \text{otherwise} \end{cases} \]  

(3)

The fitness value of \( v_j \) is calculated for the minimization problem by using Eq. (4).

\[ \text{fitness}_i = \begin{cases} \frac{1}{1 + f_i}, & f_i \geq 0 \\ 1 + \text{abs}(f_i), & f_i < 0 \end{cases} \]  

(4)

\( f_i \) is the cost value of \( v_i \) solution. Greedy selection mechanism is applied between fitness values of \( x_i \) and \( v_i \) source. If fitness value of \( x_i \) is the better than \( v_i \), \( x_i \) stays in colony and failure counter holding the number of trials of \( x_i \) source is increased by 1. In case fitness value of \( v_i \) is better, candidate solution \( v_i \) becomes the new solution of the colony and the counter is set to 0.

### C. Selection Probability of Food Sources by Onlooker Bees

When employeds complete their search for a cycle, they return to the hive and share the information with onlooker bees about the amount of nectar in the sources. Onlookers choose a food source stochastically depending on nectar amount of food sources. The standard ABC algorithm uses Eq. (5) for this selection process [16].

\[ p_i = \frac{0.9 \times \text{fitness}_{\text{best}} + 0.1}{\text{fitness}_i} \]  

(5)

where \( \text{fitness}_i \) represents the nectar amount of the \( i \)-th food source, \( \text{fitness}_{\text{best}} \) represents nectar amount of the best food sources found until that cycle and \( p_i \) is chosen probability of \( i \)-th food source. As long as \( p_i \) value increases, chosen probability of the source also increases.

### D. Quitting Criteria from Food Source: Exceeding Limit Parameter and Scouting

In the standard ABC, when all employed and onlooker bees complete their research for a cycle, all the sources are checked if there is any runs out. Counters record the number of unsuccessful attempts during modification process. If the value of a counter exceeds the limit parameter, the employed bee which is responsible for the source abandons the source position and finds a new food source as a scout bee by using Eq. (1).

### III. ACCELERATED-ABC (A-ABC) ALGORITHM

The ABC algorithm proposed by Karaboga is one of the well-known swarm-based algorithm in literature. The ABC algorithm has reasonable performance for problems which have different characteristic. In addition that, it almost constantly produces similar solutions for a particular problem whenever it’s solved in each runtime. It means that the ABC has a robust structure.

In this study two modifications are proposed called as MR and SS to improve weak part of the algorithm. While MR provides the opportunity to changing more than one parameter in a modification, SS is used to scale the exploitation process. In other words, MR contributes faster convergence than the standard ABC and SS contributes local search ability. In addition to MR and SS, objective-value is used instead of fitness value while comparing \( x_i \) and \( v_i \) in greedy selection mechanism.

#### A. MR Modification

In the standard ABC, to produce a new solution, just one parameter of the solution \( x_i \) is changed. It causes a slow convergence rate. To overcome this problem, several proposals have been made [15], [17]. For example, Akay added a control parameter called modification rate [15]. For each parameter of the \( x_i \) solution is generated a random number between \([0, 1]\). If this number is small than the MR, that parameter is modified by using Eq. (7). Parameter is modified by using Eq. (7). Akay’s modification has already improved the performance of the algorithm. However, other forms of modifications are available. This exploratory study aims to investigate another similar modification related to MR. In our work, different MR value is used for food sources according to their fitness value. For the 1/5 of the best food sources, MR value is set as 0.1. The following 1/5 of the best food sources is set as 0.2 and this process is applied until the last partition of the food sources which has relatively the worst fitness value. MR value of the last partition is set as 0.5.
This modification is introduced to ensure that inefficient sources are modified more than the better ones while better ones are modified less often.

B. Step Size (SS) Modification

Previous studies show that exploitation ability of the ABC is weak [13, 14]. To overcome this problem, a modification called SS is introduced in this study. This modification scales the exploitation ability of the sources according to progress of the gbest value. To illustrate the process, an account is provided below. First of all, the very first gbest value generated for the first cycle obtained as usual. The second cycle’s gbest value is compared against the first cycle’s gbest value. To see if it has progressed or not. If it has progressed GB succ counter is increased by 1. If not, GB fail counter is increased by 1. The next step involves a calculation related to GB succ and GB fail counters. The value generated by GB fail counter is subtracted from the value generated by GB succ. This calculation provides us with a difference value.

where \( v_j = x_j + \theta_j (x_j - x_g) \times SS \) (7)

After the generation of the SS value, it is used in employed and onlooker processes by using Eq.(7).

\[
\text{diff} = GB_{succ} - GB_{fail}
\]

\[
SS \approx \left[ \frac{rc + \text{diff}}{rc} \right]^{\frac{-|\text{diff}|}{r}}
\]

(6)

In experiments A-ABC and standard ABC was employed equally as number of MaxFES.

A. Convergence Graph

In this section we focused convergence performance of A-ABC and standard ABC. Convergence performance of some functions for 30 dimensions are shown in Fig. 1- Fig. 5. Results show that A-ABC has successful performance thanks to MR and SS modifications.

IV. Parameter Settings

We set population size (PS) as 100 bees and limit value as 200. We tested A-ABC and standard ABC algorithm with 7 different benchmark functions. MaxFES value was set 100,000, 300,000, 500,000 for 10, 30, 50 dimension respectively for all functions except Bohachevsky3. Bohachevsky3 has two dimensions and was employed 20,000 MaxFES. Informations about functions are given in Table I.

<table>
<thead>
<tr>
<th>No</th>
<th>Function</th>
<th>C</th>
<th>Range</th>
<th>Formulation</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Sphere</td>
<td>U/S</td>
<td>[-100,100]</td>
<td>( f(x) = \sum_{i=1}^{n} x_i^2 )</td>
</tr>
<tr>
<td>2</td>
<td>Rosenbrock</td>
<td>U/N</td>
<td>[-30,30]</td>
<td>( f(x) = \sum_{i=1}^{n-1} \left[ 100(x_{i+1}^2 - x_i^2)^2 + (x_i - 1)^2 \right] )</td>
</tr>
<tr>
<td>3</td>
<td>Ackley</td>
<td>M/N</td>
<td>[-32,32]</td>
<td>( f(x) = -20\exp \left{ -0.2 \sqrt{\frac{1}{n} \sum_{i=1}^{n} x_i^2} + \exp \left( \frac{1}{n} \sum_{i=1}^{n} \cos(2\pi x_i) \right) \right} + 20 + \epsilon )</td>
</tr>
<tr>
<td>4</td>
<td>Griewank</td>
<td>M/N</td>
<td>[-600,600]</td>
<td>( f(x) = \frac{1}{4000} \sum_{i=1}^{n} x_i^2 - \prod_{i=1}^{n} \cos \left( \frac{x_i}{\sqrt{i}} \right) + 1 )</td>
</tr>
<tr>
<td>5</td>
<td>Rastrigin</td>
<td>M/S</td>
<td>[-5.12,5.12]</td>
<td>( f(x) = \sum_{i=1}^{n} x_i^2 - 10\cos(2\pi x_i) + 10 )</td>
</tr>
<tr>
<td>6</td>
<td>Zakharov</td>
<td>U/N</td>
<td>[-5,10]</td>
<td>( f(x) = \sum_{i=1}^{n} x_i^2 + \left( \sum_{i=1}^{n} 0.5i x_i \right)^2 + \left( \sum_{i=1}^{n} 0.5ix_i \right)^4 )</td>
</tr>
<tr>
<td>7</td>
<td>Bohachevsky3</td>
<td>M/N</td>
<td>[-100,100]</td>
<td>( f(x) = x_1^2 + 2x_2^2 - 0.3\cos(3\pi x_1) + 4\pi x_2 + 0.3 )</td>
</tr>
</tbody>
</table>

### TABLE II: RESULTS OF F1-F6 FUNCTIONS

<table>
<thead>
<tr>
<th>Dim</th>
<th>ABC</th>
<th>Accelerated -ABC</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Best</td>
<td>Worst</td>
</tr>
<tr>
<td>10</td>
<td>6.09E-17</td>
<td>2.15E-16</td>
</tr>
<tr>
<td></td>
<td>0.023029</td>
<td>0.029684</td>
</tr>
<tr>
<td></td>
<td>8.3E-14</td>
<td>0.012247</td>
</tr>
<tr>
<td></td>
<td>0</td>
<td>4.83E-13</td>
</tr>
<tr>
<td>30</td>
<td>2.489346</td>
<td>21.94919</td>
</tr>
<tr>
<td></td>
<td>4.77E-16</td>
<td>9.64E-16</td>
</tr>
<tr>
<td></td>
<td>0.077905</td>
<td>1.352967</td>
</tr>
<tr>
<td></td>
<td>2.6E-10</td>
<td>1.29E-09</td>
</tr>
<tr>
<td></td>
<td>2.22E-16</td>
<td>3.44E-12</td>
</tr>
<tr>
<td></td>
<td>5.68E-14</td>
<td>3.31E-09</td>
</tr>
<tr>
<td></td>
<td>1.18E-15</td>
<td>2.08E-15</td>
</tr>
<tr>
<td></td>
<td>0.146785</td>
<td>1.711159</td>
</tr>
<tr>
<td></td>
<td>4.59E-10</td>
<td>3.02E-09</td>
</tr>
<tr>
<td></td>
<td>8.88E-16</td>
<td>2.8E-11</td>
</tr>
<tr>
<td></td>
<td>7.96E-12</td>
<td>0.001047</td>
</tr>
<tr>
<td></td>
<td>421.4083</td>
<td>578.6242</td>
</tr>
</tbody>
</table>

PS: 100, Limit:200, Dim:Dimension

### TABLE III: RESULTS OF F7 FUNCTION

<table>
<thead>
<tr>
<th>Dim</th>
<th>ABC</th>
<th>Accelerated -ABC</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Best</td>
<td>Worst</td>
</tr>
<tr>
<td>2</td>
<td>4.08E-07</td>
<td>0.000463</td>
</tr>
</tbody>
</table>

PS: 100, Limit:200, MaxFES:20.000

---

**Fig. 1.** Convergence performance of f1 function in 30 dimension.

**Fig. 2.** Convergence performance of f2 function in 30 dimension.

**Fig. 3.** Convergence performance of f3 function in 30 dimension.

**Fig. 4.** Convergence performance of f4 function in 30 dimension.
standard version of the ABC algorithm and A-ABC algorithm modifications named as MR and SS were applied on the in a local minima. To overcome these issues, two different inefficient points such as slow convergence and getting stuck has faster convergence than basic version of the algorithm. Through the compare on the performances of standard ABC and A-ABC algorithms, the study in this paper accured. Through the compare on the performances of unimodal and multimodal functions. In addition that A-ABC shows that A-ABC has better results than standard ABC both.

V. CONCLUSION
As described in Section IV, the ABC algorithm has some inefficient points such as slow convergence and getting stuck in a local minima. To overcome these issues, two different modifications named as MR and SS were applied on the standard version of the ABC algorithm and A-ABC algorithm accured. Through the compare on the performances of standard ABC and A-ABC algorithms, the study in this paper shows that A-ABC has better results than standard ABC both unimodal and multimodal functions. In addition that A-ABC has faster convergence than basic version of the algorithm.

REFERENCES

Ahmet Ozkis received the B.E. Degree from Department of Electronic and Computer Education at Selçuk University, Konya, Turkey in 2010. He is studying on M.E. at Department of Computer Engineering at Selçuk University. Also he is a research assistant in same department. His researches are included swarm intelligence, optimization algorithms and artificial intelligence.

Ahmet Babalik received the B.E. and M.E. degrees from Department of Electronic and Computer Education at Gazi University, Ankara, Turkey in 1996 and 2000 respectively. His Ph.D. degree was received from the Department of Electrical and Electronics Engineering at Selçuk University, Konya, Turkey in 2007. He has been an assistant professor in the Department of Computer Engineering in Selçuk University since 2007. His researches are included artificial neural networks, swarm intelligence and artificial intelligence.