An Improved Artificial Bee Colony Algorithm Based on a New Probability Selection Mechanism and a Movement Equation

Chunfeng Wang, Qingyuan Li, Zhijian Duan, and Keke Zhang

Abstract—**This paper presents an improved artificial bee colony (ABC) algorithm, named ABC_NPME, to overcome the limitations of traditional ABC approaches. ABC_NPME introduces a novel probability selection mechanism that enhances the basic ABC's fitness-based selection. This mechanism considers not only the fitness values of individuals but also their ability to improve search efficiency in their vicinity, allowing for the selection of higher-quality potential solutions. Additionally, we propose a new movement equation for onlooker bees, derived from an asymmetric opposite search strategy near the population's optimal solution, which facilitates more effective position updates. The advantages of ABC_NPME include enhanced exploration, robust performance, adaptability to various optimization landscapes, and improved convergence. Comprehensive numerical experiments validate that ABC_NPME outperforms some existing state-of-the-art optimization algorithms.**

Index Terms—**Artificial bee colony; Swarm intelligence; Probability mechanism; Movement equation**

I. Introduction

THE ABC algorithm, introduced in 2005, simulates the collective intelligence of bee swarms for solving optimization problems [1]. Since then, it has been successfully applied in various fields. It is important to note that the selection mechanism within the algorithm significantly affects its performance. Studies indicate that both the selection mechanism and the movement equation have a substantial impact on the evolution of populations In these two areas, researchers have conducted extensive studies. For example, in the study of selection mechanisms, they have suggested various strategies to improve the selection of individuals within the swarm, thereby improving the overall efficiency and quality of the algorithm's solutions [2-9]. (1) Fitness-based selection: This approach prioritizes individuals with higher fitness values, increasing the chances of identifying optimal solutions. While it can improve search efficiency and typically lead to better solution quality, a strong focus on fitness can result in premature convergence, limiting exploration of the search space. (2) Diversity-based selection: This mechanism promotes population diversity by selecting high-fitness individuals that are distant from current optimal solutions, thereby preventing premature convergence. By emphasizing diversity within the swarm, it helps avoid local optima and enhances the global search capabilities of the algorithm. However, it may overlook some high-quality individuals, which can limit the fine-tuning of local optimal solutions. (3) Dynamic selection: This approach adapts its strategy based on real-time feedback during the iterative process, transitioning from random selection to focusing on high-fitness individuals. This adaptability enhances the search process, allowing it to accommodate complex problem dynamics. However, its implementation can be more complex and may introduce additional parameters and computational overhead. (4) Heuristic-based selection: This mechanism uses domain knowledge to prioritize individuals likely to yield better solutions. By incorporating heuristic insights into the selection process, it aims to improve the accuracy and efficiency of individual selection. This approach can significantly enhance algorithm efficiency, particularly in complex scenarios. However, its effectiveness may be limited to specific domains or problems, which could impact its overall applicability.

Additionally, substantial research is focused on improving the movement equation [10-19].

Inspired by previous research, this paper proposes a novel selection mechanism within the framework of ABC. The proposed mechanism considers not only the fitness values of individuals but also the frequency with which they have improved potential solutions throughout the optimization process. By evaluating both fitness and improvement frequency, this mechanism aims to promote individuals that consistently enhance their solution quality. Additionally, based on the asymmetric opposite search strategy near the population's optimal solution, a new movement equation is designed for onlooker bees to determine their new positions. The advantages of this new algorithm are summarized as follows:

(1) Enhanced Exploration: By considering the improvement frequency, the algorithm retains individuals who continuously contribute to the enhancement of solutions, helping to maintain a diverse population and reducing the likelihood of premature convergence.

Manuscript received November 16, 2024; revised February 22, 2025.

This work was supported by the Basic Research Program of Natural Science of Shaanxi Province (2024JC-YBMS003,2024JC-YBMS013); the Key Cultivation Project of Xianyang Normal University (XSYK21044); the Research Project on Teaching Reform of Xianyang Normal University (2023C117); the 14th Five-Year Plan for Education Science in Shaanxi Province (SGH23Y2506); the Doctoral Scientific Research Foundation of Xianyang Normal University (1052003610); the Major Education Project of Shaanxi Province(22JK0601); the Key Research Projects of Higher Education Institutions of Shaanxi Province(XSYK21029).

Chunfeng Wang is a professor of the School of Mathematics and Statistics, Xianyang Normal University, Xianxiang, 712000, PR China. Email:wangchunfeng09@126.com

Qingyuan Li is an undergraduate student of the School of Mathematics and Statistics, Xianyang Normal University, Xianyang, 712000, PR China. Email: 2979437170@qq.com

Zhijian Duan is an associate professor of the School of Mathematics and Statistics, Xianyang Normal University, Xianyang, 712000, PR China. Email: zhijian_duan@126.com

Keke Zhang is an assistant professor of the School of Mathematics and Statistics, Xianyang Normal University, Xianyang, 712000, PR China. Email: 325119113@qq.com

(2) Robust Performance: The dual evaluation criteriałfitness and improvement potentialłallow for a more comprehensive assessment of individuals, potentially leading to the discovery of higher-quality solutions.

(3) Adaptability: This enhanced selection mechanism adds flexibility to the algorithm, enabling it to adapt more effectively to various optimization landscapes by balancing exploration and exploitation.

(4) Improved Convergence: By rewarding individuals that exhibit a pattern of improvement, the algorithm is likely to converge towards optimal solutions more reliably, resulting in a stable and efficient optimization process.

The rest of this paper is organized as follows: Section II introduces the basic ABC process, while Section III provides detailed insights into ABC_NPME. Section IV discusses the experiments and their results, and Section V presents the conclusions and outlines directions for future work.

II. Basic ABC algorithm

In the basic ABC, there are three types of bees: employed bees, onlooker bees, and scout bees. Employed and onlooker bees each make up half of the colony size Sn, with each employed bee corresponding to one food source. Each type of bee has distinct roles and responsibilities. Employed bees explore the search space and record food source locations, sharing this information with onlooker bees, which then select food sources based on the received data. When a food source is exhausted, the employed bee transitions into a scout bee and begins searching for new food sources. The main steps of ABC are outlined below. Assume that the dimensionality of the search space is D.

(1) **Initialization**

The initial step is to initialize the population by generating a random population of food sources using Equation (1):

$$x_i^j = l_i^j + \text{rand}(0,1) \times (u_i^j - l_i^j), \tag{1}$$

where the upper bound and $l_i^j (j = 1, \cdots, D)$ denotes the lower bound for the i-th food source($i = 1, \cdots, Sn$).

(2) **Employed bee phase**

During this phase, Equation (2) below is used to search near the existing solution:

$$v_i^j = x_i^j + \text{rand}(0,1) \times (x_i^j - x_k^j), \tag{2}$$

where $k \in \{1, \cdots, Sn\}(k \neq i)$, and $j \in \{1, \cdots, D\}$ are randomly selected indices.

Next, the fitness value of the new solution is determined by Equation (3):

$$fit(v_i) = \begin{cases} \frac{1}{1+f(v_i)}, & \text{if} \quad f(v_i) \geq 0 \\ 1 + |f(v_i)|, & \text{if} \quad f(v_i) < 0. \end{cases} \tag{3}$$

The current solution is updated based on the fitness value by a greedy rule: if $fit(v_i) > fit(x_i)$, set $x_i = v_i,\ fit(x_i) = fit(v_i)$.

(3) **Onlooker bee phase**

Onlooker bees begin their task after the employed bee phase is complete. During this phase, each onlooker bee selects a food source based on the quality information derived from the food sources indicated by the employed bees. Equation (4) calculates the probability of selecting these sources, with a higher fitness value corresponding to a greater likelihood of selection.

$$p_i^1 = \frac{fit(x_i)}{\sum_{i=1}^{Sn} fit(x_i)}. \tag{4}$$

(4) **Scout bee phase**

If a food source remains unchanged for a specified number of iterations, denoted as $limit$, it is considered exhausted. Subsequently, the employed bee transitions to a scout bee and randomly generates a new food source within the search area using Equation (1).

The framework of the ABC is outlined in Algorithm 1 below.

Algorithom 1. Basic ABC

01. Given the population size Sn, the upper limit for iterations $MaxIt$, and the given threshold $limit$.
02. Generate the initial population $\{x_i | i = 1, \cdots, Sn\}$. Compute the function values of the population $\{f(x_i) | i = 1, \cdots, Sn\}$, the fitness of the population $\{fit(x_i) | i = 1, \cdots, Sn\}$, and determine the best solution x_{best}.
03. When *the stopping criterion is not met* do
04. For $i = 1$ to Sn do
05. Employed bees use (2) to obtain the new candidate v_i.
06. If $fit(v_i) > fit(x_i)$
07. set $x_i = v_i, fit_i = fit(v_i)$
08. End if
09. End for
10. For $i = 1$ to Sn do
11. Select the food source by the probability p_i^1, and search the candidate near the food source for each onlooker bee according to (2).
12. Identify new food sources by the greedy rule.
13. End for
14. For the ith food source, if it remains unchanged for the set number of iterations $limit$, obtain a new candidate according to (1).
15. Set $it = it + 1$.
16. End when
17. Output the final result.

III. The proposed ABC (ABC_NPME)

As previously noted, the basic ABC algorithm relies solely on the fitness-based probability determined by Equation (4) to select search individuals and uses Equation (2) to generate new solutions for the onlooker bees. This probability selection mechanism may cause the algorithm to become trapped in local optima. Additionally, the movement equation in the basic ABC algorithm employs a unit search strategy, resulting in slow convergence rates.

To address these limitations, this paper introduces two improvement strategies: (1) newly designed probability selection mechanism and (2) a movement equation that adopts a multidimensional search approach to determine food sources. The detailed processes are outlined as follows.

A. Improvement strategy 1: the new probability selection mechanism

In the onlooker bee phase, let us assume that the i-th food source, denoted as x_i, is selected. We use $Num(i)$ to record the improvements observed by the onlooker bees while searching at the i-th food source, and v_i represents the new food source generated near x_i. Initially, $Num(i)$ is set to 0. After generating the new food source v_i, $Num(i)$ is updated according to the following formula:

$$Num(i) = \begin{cases} Num(i)+1 & \text{if } f(v_i) < f(x_i), \\ Num(i) & \text{else.} \end{cases}$$

In the scout phase, if the i-th food source is discarded, its corresponding $Num(i)$ is reset to 0. After completing one iteration, a new probability, denoted as p_i^2, is defined based on $Num(i)$ as follows:

$$p_i^2 = \frac{Num(i)}{\sum_{i=1}^{Sn} Num(i) + 0.01}. \tag{5}$$

According to Equation (5), it can be observed that the more times a new solution is improved after selecting the food source x_i, the larger the corresponding p_i^2 value becomes. To identify food sources with favorable fitness values and those that can be further improved after the search, this study proposes a novel probability selection mechanism based on p^1 and p^2:

$$p_i = (1 - \frac{it}{MaxIt}) \times p_i^1 + \frac{it}{MaxIt} \times p_i^2. \tag{6}$$

where it represents the current iteration count.

From (6), it can be seen that in the early stage of the algorithm, the probability proportion of p_i^1 is relatively high. As the algorithm iterates, the proportion of p_i^2 increases. In other words, during the initial stages, the algorithm tends to choose individuals with better fitness values as search targets, while in the later stages, it tends to select food sources with more improvement instances as search targets. Therefore, this probability selection mechanism demonstrates good adaptability.

B. Improvement strategy 2: the new movement equation

After selecting food sources that require further search based on the new probability, it is necessary to generate a new location in some way. This paper proposes a new search equation with multidimensional changes. The detailed process is given below.

Assume that x_{best} is the optimal position of the current population and x_i is the selected food source to be further searched. a new position v_i is obtained as follows: for $j \in \{1, 2, \cdots, D\}$, if $rand < MR$, set

$$\begin{aligned} \eta &= l_j + (u_j - x_{best,j}), \\ v_i^j &= x_{best}^j + rand \times (rand \times \eta - x_{best}^j), \end{aligned} \tag{7}$$

where MR is defined as:

$$MR = \exp(-\frac{it}{MaxIt}) \times MR_{max}.$$

Here, MR_{max} denotes the initial value, which is set to 0.5 in this paper. According to the definition of MR, it is evident that the initial MR of the algorithm is relatively large, and as the iteration progresses, MR gradually decreases. Therefore, this strategy enhances the algorithms global search capability in the earlier stages while gradually strengthening its local search in the later stages.

After obtaining the new location v_i, update the food source location using the following greedy rule: if $f(v_i) < f(x_i)$, set

$$x_i = v_i, Num(i) = Num(i) + 1.$$

The following presents the pseudo-code for the proposed algorithm ABC_NPME:

Algorithom 2. ABC_NPME

01. Given the population size Sn, the upper limit for iterations $MaxIt$, and the given threshold $limit$.
02. Generate the initial population $\{x_i | i = 1, \cdots, Sn\}$. Compute the function value of the population $\{f(x_i) | i = 1, \cdots, Sn\}$, the fitness of the population $\{fit(x_i) | i = 1, \cdots, Sn\}$, and determine the best solution x_{best}.
03. When *the stopping criterion is not met* do
04. For $i = 1$ to Sn do
05. Employed bees use (2) to obtain the new candidate v_i.
06. If $fit(v_i) > fit(x_i)$
07. set $x_i = v_i, fit_i = fit(v_i)$
08. End if
09. End for
10. Calculate p_i according to (6).
11. For $i = 1$ to Sn do
12. Select the food source by the probability p_i, and search the candidate near the food source for each onlooker bee according to (7).
13. Identify new food sources in a greedy way.
14. End for
15. For the ith food source, if it remains unchanged for the set number of iterations $limit$, obtain a new candidate according to (1).
16. Set $it = it + 1$.
17. End when
18. Output the final result.

The flowchart of ABC_NPME is shown in Figure 1.

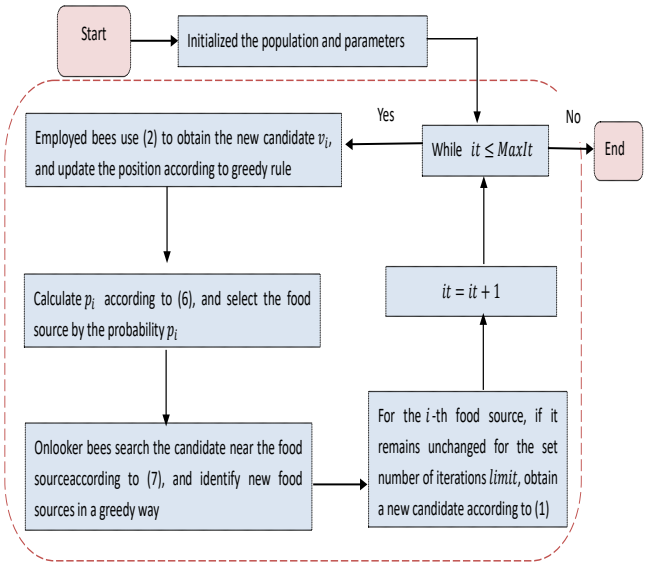

Figure 1: The flowchart of ABC_NPME

IV. Numerical experiments

To evaluate the performance of the ABC_NPME algorithm, we tested it alongside six variants of the ABC algorithm: GABC [20], GBABC [21], KFABC[22], LFABC[23], AABC [24], and NABC [25] on 16 benchmark functions and two practical problems. The experiments were carried out on a computer equipped with an Intel (R) Core (TM) i7-6500U CPU running at 2.50 GHz, 8 GB of RAM, and Windows 10 operating system, utilizing Matlab 2017a for implementation.

A. Test on Benchmark functions

1) Benchmark functions: Table I presents the 16 benchmark functions used in this study. In this table, D denotes the dimensions, Range indicates the bounds of the search space, and $f(x^*)$ represents the global minimum values for each function. Among these benchmark functions, f_1 to f_{10} are unimodal functions, with f_9 identified as a discontinuous step function, and f_{10} as a noisy quadratic function. In contrast, f_{11} to f_{16} are multimodal functions.

TABLE I: Benchmark test functions

Functions	Range	Optimal value
$f_1 = \sum_{i=1}^{D} x_i^2$	[-100,100]	0
$f_2 = \sum_{i=1}^{D} \mid x_i \mid + \prod_{i=1}^{D} \mid x_i \mid$	[-10,10]	0
$f_3 = \sum_{i=1}^{D} (\sum_{j=1}^{i} x_j)^2$	[-100,100]	0
$f_4 = \max\{\mid x_i \mid, 1 \leq i \leq D\}$	[-100,100]	0
$f_5 = \sum_{i=1}^{D} i x_i^2$	[-10,10]	0
$f_6 = \sum_{i=1}^{D} i x_i^4$	[-1.28,1.28]	0
$f_7 = \sum_{i=1}^{D} \mid x_i \mid^{(i+1)}$	[-1,1]	0
$f_8 = \sum_{i=1}^{D} (10^6)^{\frac{i-1}{D-1}} x_i^2$	[-100,100]	0
$f_9 = \sum_{i=1}^{D} (\lfloor x_i + 0.5 \rfloor)^2$	[-1.28,1.28]	0
$f_{10} = \sum_{i=1}^{D} i x_i^4 + random[0,1)$	[-1.28,1.28]	0
$f_{11} = \sum_{i=1}^{D} (x_i^2 - 10\cos(2\pi x_i) + 10)$	[-5.12,5.12]	0
$f_{12} = -20\exp(-0.2 * \sqrt{\sum_{i=1}^{D} x_i^2/D})$ $-\exp(\frac{1}{D}\sum_{i=1}^{D}\cos(2\pi x_i)) + 20 + e$	[-32,32]	0
$f_{13} = \frac{1}{4000}\sum_{i=1}^{D} x_i^2 - \prod_{i=1}^{D} cos(\frac{x_i}{\sqrt{i}}) + 1$	[-600,600]	0
$f_{14} = 0.5 + \frac{sin(\sqrt{\sum_{i=1}^{D} x_i^2)^2 - 0.5}}{(1+0.001\sum_{i=1}^{D} x_i^2)^2}$	[-100,100]	0
$f_{15} = \frac{1}{D}\sum_{i=1}^{D} (x_i^4 - 16x_i^2 + 5x_i)$	[-5,5]	-78.3323
$f_{16} = \sum_{i=1}^{D} \mid x_i sin(x_i) + 0.1x_i \mid$	[-10,10]	0

To ensure a fair comparison, each algorithm is executed independently 30 times for each problem, utilizing a population size of 30. The maximum number of iterations is set to 3000, serving as the termination criterion. Other parameters are consistent with those recommended for the comparison algorithms. The results of these comparisons are presented in Tables II and III.

2) Comparison results: When comparing the seven algorithms across multiple performance metrics-Min, Mean, and Std-it is evident that each algorithm has its strengths and weaknesses depending on the function evaluated. From Table II, it is evident that ABC_NPME achieves optimal values for nearly all test functions, accounting for 93% of the total. For f_9, all tested algorithms demonstrate identical performance. For functions f_1 and f_2, ABC_NPME, GABC, GBABC, AABC and NABC all achieve the optimal value and exhibit good performance. In comparison, KFABC and LFABC exhibit poorer performance. For functions f_3 to f_8, f_{10}, f_{12} to f_{14}, and f_{16}, ABC_NPME demonstrates the best performance. However, for function f_{15}, the performance of ABC_NPME is not satisfactory, and its results are slightly worse than those of the other algorithms.

TABLE II: Optimal results obtained by ABC_NPME, GABC, GBABC, KFABC, LFABC, AABC and NABC

Function	Algorithm	Min	Mean	Std
f_1	ABC_NPME	0	0	0
	GABC	0	0	0
	GBABC	0	0	0
	KFABC	9.71E-154	6.98E-13	2.21E-12
	LFABC	5.12E-124	8.74E-108	2.29E-107
	AABC	0	0	0
	NABC	0	0	0
f_2	ABC_NPME	0	0	0
	GABC	0	0	0
	GBABC	0	0	0
	KFABC	1.34E-48	1.37E-08	3.44E-08
	LFABC	1.47E-83	2.30E-75	6.19E-75
	AABC	0	0	0
	NABC	0	0	0
f_3	ABC_NPME	0	0	0
	GABC	3.13E+3	5.07E+3	1.19E+3
	GBABC	2.57E+2	2.39E+3	1.72E+3
	KFABC	3.62E+2	1.01E+4	2.74E+3
	LFABC	1.76E+3	5.0E+3	2.77E+3
	AABC	5.05E+3	7.40E+3	1.47E+3
	NABC	2.18E+3	4.40E+3	1.20E+3
f_4	ABC_NPME	0	0	0
	GABC	3.39E-1	6.39E-1	1.69E-1
	GBABC	5.21E-3	1.20E-2	6.87E-3
	KFABC	6.29E-2	1.27E-1	4.61E-2
	LFABC	1.95E-1	2.42E-1	3.11E-2
	AABC	2.37E-1	3.59E-1	5.49E-2
	NABC	2.30E-1	2.91E-1	4.34E-2
f_5	ABC_NPME	0	0	0
	GABC	1.27E-57	2.31E-56	2.15E-56
	GBABC	4.64E-63	2.10E-59	4.86E-59
	KFABC	8.01E-75	1.56E-09	3.79E-09
	LFABC	3.54E-71	1.89E-70	2.03E-70
	AABC	8.34E-86	8.08E-83	1.17E-82
	NABC	3.94E-81	1.56E-79	2.88E-79
f_6	ABC_NPME	0	0	0
	GABC	2.34E-122	1.00E-119	2.71E-119
	GBABC	4.91E-98	2.39E-95	5.28E-95
	KFABC	4.77E-216	3.42E-20	1.08E-19
	LFABC	1.76E-147	1.67E-144	1.65E-144
	AABC	7.83E-169	5.89E-162	1.86E-161
	NABC	4.17E-164	9.82E-162	9.94E-162
f_7	ABC_NPME	0	0	0
	GABC	8.87E-85	2.52E-78	7.97E-78
	GBABC	1.39E-190	5.88E-180	0
	KFABC	2.46E-71	8.53E-09	1.79E-08
	LFABC	4.18E-107	2.16E-101	6.15E-101
	AABC	1.64E-102	4.86E-90	1.54E-89
	NABC	1.39E-145	3.35E-135	1.06E-134
f_8	ABC_NPME	0	0	0
	GABC	1.73E-43	9.44E-42	1.25E-41
	GBABC	1.52E-67	4.43E-61	7.74E-61
	KFABC	2.48E-36	5.27E-07	1.14E-06
	LFABC	3.08E-64	4.93E-63	7.05E-63
	AABC	8.87E-81	7.85E-79	1.38E-78
	NABC	2.24E-73	4.12E-71	8.04E-71

To visually demonstrate the performance of different algorithms on functions f_1 to f_{16}, we have plotted boxplots for each algorithm (See Figures 2 to 5). In Figures 2-5, the numbers 1,2,3,4,5,6 and 7 represent ABC_NPME, GABC,

TABLE III: Optimal results obtained by ABC_NPME, GABC, GBABC, KFABC, LFABC, AABC and NABC

Function	Algorithm	Min	Mean	Std
f_9	ABC_NPME	0	0	0
	GABC	0	0	0
	GBABC	0	0	0
	KFABC	0	0	0
	LFABC	0	0	0
	AABC	0	0	0
	NABC	0	0	0
f_{10}	ABC_NPME	7.81E-07	2.18E-05	1.82E-05
	GABC	2.78E-2	3.56E-2	4.63E-3
	GBABC	5.34E-3	1.19E-2	4.07E-3
	KFABC	1.26E-3	3.88E-3	1.79E-3
	LFABC	2.07E-2	4.18E-2	1.79E-2
	AABC	2.65E-2	5.15E-2	1.58E-2
	NABC	2.94E-4	4.96E-4	1.72E-4
f_{11}	ABC_NPME	0	0	0
	GABC	0	0	0
	GBABC	0	0	0
	KFABC	0	5.49E-4	1.37E-3
	LFABC	0	2.07E-14	6.51E-14
	AABC	0	0	0
	NABC	0	0	0
f_{12}	ABC_NPME	-8.88E-16	-8.88E-16	0
	GABC	2.75E-14	3.18E-14	3.67E-15
	GBABC	6.22E-15	7.63E-15	2.48E-15
	KFABC	2.04E-14	9.61E-2	2.17E-1
	LFABC	3.11E-14	3.49E-14	4.25E-15
	AABC	2.04E-14	2.36E-14	3.11E-15
	NABC	2.66E-15	2.66E-15	0
f_{13}	ABC_NPME	0	0	0
	GABC	1.85E-3	2.57E-3	7.32E-4
	GBABC	4.33E-3	9.84E-3	4.62E-3
	KFABC	1.05E-4	4.72E-2	1.48E-1
	LFABC	1.67E-3	2.65E-3	7.24E-4
	AABC	8.44E-06	2.56E-05	1.43E-05
	NABC	1.14E-3	2.15E-3	8.11E-4
f_{14}	ABC_NPME	0	0	0
	GABC	2.27E-1	2.98E-1	3.67E-2
	GBABC	3.72E-2	6.58E-2	1.97E-2
	KFABC	9.72E-3	1.26E-1	6.56E-2
	LFABC	2.28E-1	3.24E-1	6.23E-2
	AABC	7.82E-2	3.10E-1	8.70E-2
	NABC	2.81E-2	4.04E-2	1.36E-2
f_{15}	ABC_NPME	-78.32	-78.32	2.70E-3
	GABC	-78.33	-78.33	1.25E-14
	GBABC	-78.33	-78.33	3.07E-14
	KFABC	-78.33	-78.33	3.14E-09
	LFABC	-78.33	-78.33	1.34E-14
	AABC	-78.33	-78.33	1.64E-14
	NABC	-78.33	-78.33	2.27E-14
f_{16}	ABC_NPME	0	0	0
	GABC	1.73E-12	3.03E-9	6.23E-9
	GBABC	1.34E-36	5.12E-12	1.59E-11
	KFABC	1.39E-33	7.40E-05	1.22E-04
	LFABC	2.42E-19	3.86E-07	8.15E-07
	AABC	1.17E-44	6.11E-17	1.93E-16
	NABC	3.87e-17	9.05E-13	2.21E-12

GBABC, KFABC, LFABC, AABC, and NABC, respectively. From Figures 2-5, it is clear that the optimal values and stability of ABC_NPME rank first, except for function f_{15}, which aligns with the results presented in Table I.

In addition, we conducted a Friedman test on the comparative algorithms, and the resulting Friedman scores are shown in Table IV. The results indicate that, except for f_{15}, ABC_NPME achieves the best Friedman scores. For function f_{15}, the Friedman score of ABC_NPME is 7, which is the worst among all algorithms. This suggests that the performance of ABC_NPME needs to be further improved for function f_{15}. To demonstrate the average Friedman score of different algorithms across all functions, we have drawn a bar chart (See Figure 6). As shown in Figure 6, the average score of ABC_NPME is 2.0, significantly better than that of the other algorithms.

B. Test on two practical problems

In the engineering domain, numerous practical challenges can be simplified to optimization problems that are modeled mathematically. Practical evidence indicates that incorporating intelligent algorithms into engineering optimization can effectively address various complex issues, yielding significant economic and social benefits. In this section, to further demonstrate the effectiveness of ABC_NPME, we apply it to two engineering challenges: tension-pressure spring design and welded beam design. We then compare the optimal solutions achieved by ABC_NPME with those obtained using other intelligent algorithms.

1) Tension-pressure spring design problem: The structural design of tension-pressure springs (See Figure 6) primarily focuses on minimizing the spring's weight while adhering to a set of constraint conditions. These constraints include minimum deflection, shear stress, surge frequency, and restrictions on the outer diameter, as well as the design variables. Specifically, there are three design variables involved: coil diameter of spring ($d(x_1)$), average diameter of spring coil ($D(x_2)$), and the effective number of circles ($P(x_3)$).

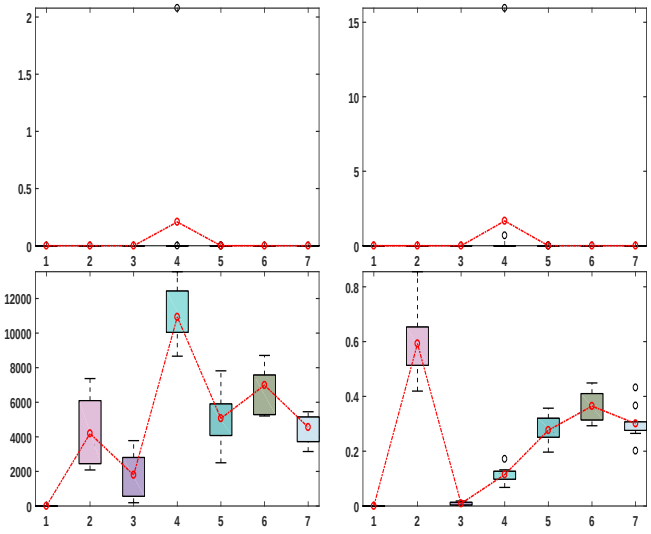

Figure 2: Boxplot for different algorithms on $f_1 - f_4$

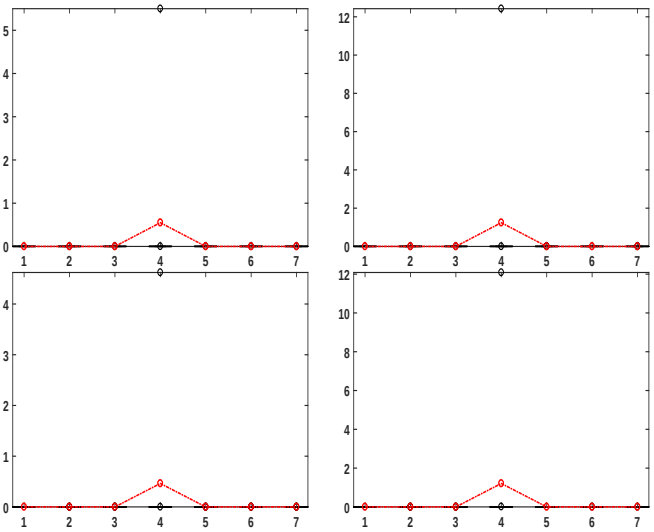

Figure 3: Boxplot for different algorithms on $f_5 - f_8$

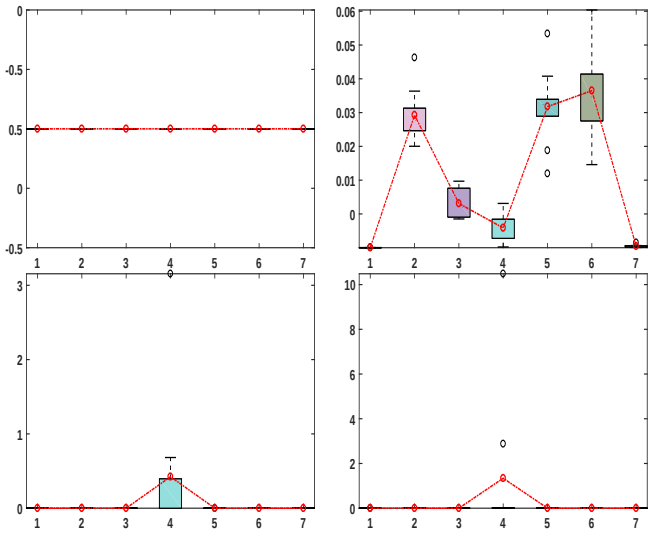

Figure 4: Boxplot for different algorithms on $f_9 - f_{12}$

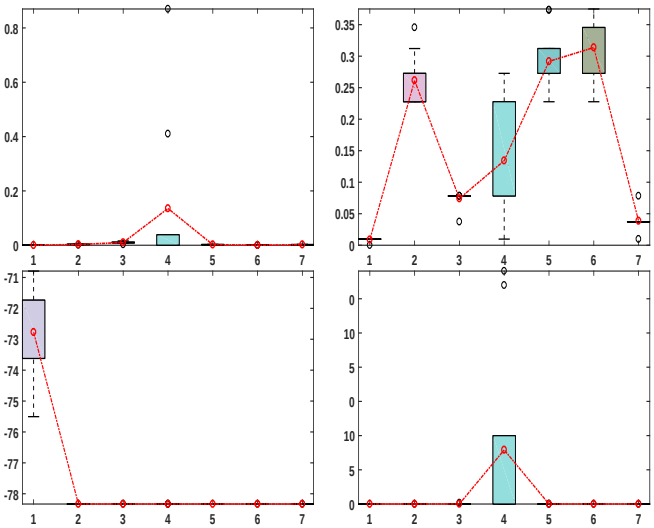

Figure 5: Boxplot for different algorithms on $f_{13} - f_{16}$

TABLE IV: Friedman scores of IMABC, GABC, GBABC, KFABC, LFABC, AABC and NABC for each function

Function	ABC_NPME	GABC	GBABC	KFABC	LFABC	AABC	NABC
f_1	3.0	3.0	3.0	6.7	6.3	3.0	3.0
f_2	3.0	3.0	3.0	7.0	6.0	3.0	3.0
f_3	1.0	4.5	2.4	6.7	3.9	5.7	3.8
f_4	1.0	6.9	2.0	3.0	4.2	5.9	5.0
f_5	1.0	6.2	5.2	6.4	4.2	2.0	3.0
f_6	1.0	5.1	6.2	6.4	4.1	2.2	3.0
f_7	1.0	6.0	2.0	7.0	4.0	5.0	3.0
f_8	1.0	6.0	4.5	7.0	4.5	2.0	3.0
f_9	4.0	4.0	4.0	4.0	4.0	4.0	4.0
f_{10}	1.0	5.7	4.0	3.0	5.8	6.5	2.0
f_{11}	3.5	3.5	3.5	6.4	4.1	3.5	3.5
f_{12}	1.0	5.45	3.0	6.45	5.95	4.15	2.0
f_{13}	1.0	5.0	6.9	3.4	5.1	2.0	4.6
f_{14}	1.0	5.6	2.9	3.9	6.1	6.2	2.3
f_{15}	7.0	3.0	5.05	2.8	3.7	2.95	3.5
f_{16}	1.0	6.0	3.3	6.4	3.9	2.1	4.3

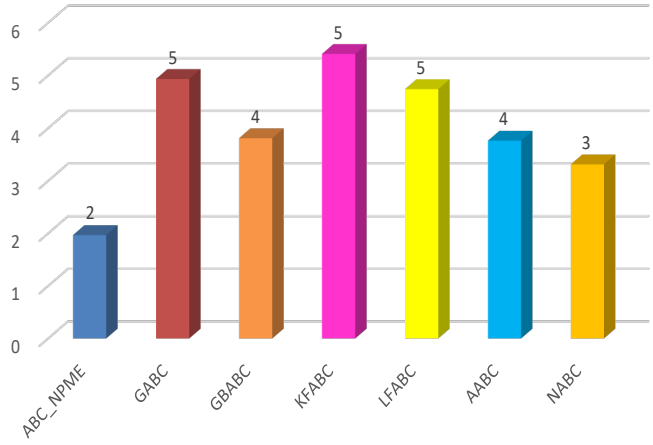

Figure 6: The average Friedman scores of different algorithms

The optimization problem can be formulated as follows:

$$\min f(x) = (x_3 + 2)x_2x_1^2$$

subject to

$$g_1(x) = 1 - \frac{x_2^3 x_3}{71785x_1^4} \leq 0,$$
$$g_2(x) = \frac{4x_2^2 - x_1x_2}{12566(x_2x_1^3 - x_1^4)} + \frac{1}{5108x_1^2} - 1 \leq 0,$$
$$g_3(x) = 1 - \frac{140.45x_1}{x_2^2x_3} \leq 0,$$
$$g_4(x) = \frac{x_1 + x_2}{1.5} - 1 \leq 0,$$

where

$$0.05 \leq x_1 \leq 2,$$
$$0.25 \leq x_2 \leq 1.3,$$
$$2 \leq x_3 \leq 15,$$

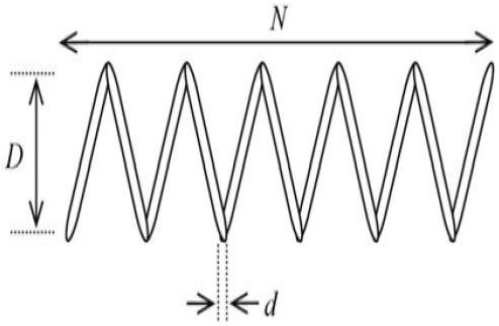

Figure 6 Tension-pressure spring design problem

For this optimization problem, each algorithm is run independently 30 times. The best solutions obtained by the different algorithms are presented in Table V.

TABLE V: Optimal results obtained by ABC_NPME, GABC, GBABC, KFABC, LFABC, AABC and NABC

Optimum	ABC_NPME	GABC	GBABC	KFABC	LFABC	AABC	NABC
x_1	0.0519	0.1362	0.0748	0.0603	0.0510	0.0575	0.0505
x_2	0.3628	1.2754	1.1608	0.6030	0.3397	0.5101	0.3286
x_3	10.9388	11.8539	4.7075	4.3427	12.4166	6.2172	13.1530
$f(x)$	**0.0127**	**0.0127**	0.1313	**0.0127**	0.0129	0.0139	**0.0127**

As illustrated in Table V, for the tension-pressure spring design problem, ABC_NPME outperforms the other algorithms by identifying the optimal solution and achieving the

highest objective function value. Consequently, ABC_NPME demonstrates superiority in addressing this problem.

2) Welded beam design problem: The object of this practical problem is to minimize the manufacturing cost of the welded beam while adhering to constraints related to shear stress(τ), end deflection of the beam(δ), buckling load on the bar (P_c), and bending stress(σ). Additionally, there are four design parameters: $h(x_1)$, $l(x_2)$, $t(x_3)$, and $b(x_4)$, as shown in Figure 7.

The optimization problem can be stated as follows:

$$\min f(x) = 1.10471x_1^2x_2 + 0.04811x_3x_4(14.0 + x_2)$$

subject to

$$\begin{aligned}
&g_1(x) = \tau(x) - \tau_{max} \leq 0,\\
&g_2(x) = \sigma(x) - \sigma_{max} \leq 0,\\
&g_3(x) = x_1 - x_4 \leq 0,\\
&g_4(x) = 0.10471x_1^2 + 0.04811x_3x_4(14.0 + x_2) - 5.0 \leq 0,\\
&g_5(x) = 0.125 - x_1 \leq 0,\\
&g_6(x) = \delta(x) - \delta_{max} \leq 0,\\
&g_7(x) = P - P_c \leq 0,
\end{aligned}$$

where

$$\begin{aligned}
&\tau(x) = \sqrt{(\tau')^2 + 2\tau'\tau''\frac{x_2}{2R} + (\tau'')^2},\\
&\tau' = \frac{P}{\sqrt{2}x_1x_2},\\
&\tau'' = \frac{MR}{J},\\
&M = P(L + \tfrac{x_2}{2}),\\
&R = \sqrt{\tfrac{x_2^2}{4} + (\tfrac{x_1+x_3}{2})^2},\\
&J = 2[\sqrt{2}x_1x_2(\tfrac{x_2^2}{12} + (\tfrac{x_1+x_3}{2})^2)],\\
&\sigma(x) = \frac{6PL}{x_4x_3^2},\\
&\delta(x) = \frac{4PL^3}{Ex_3^3x_4},\\
&P_c = \frac{4.013E\sqrt{\frac{x_3^2x_4^6}{36}}}{L^2}(1 - \frac{x_3}{2L}\sqrt{\frac{E}{4G}}),\\
&P = 6000lb, L = 14in, E = 30e6psi,\\
&G = 12e6psi, \tau_{max} = 13600psi,\\
&\sigma_{max} = 3000psi, \delta_{max} = 0.25in,\\
&0.1 \leq x_1 \leq 2.0, 0.1 \leq x_2 \leq 10.0,\\
&0.1 \leq x_3 \leq 10.0, 0.1 \leq x_4 \leq 2.0.
\end{aligned}$$

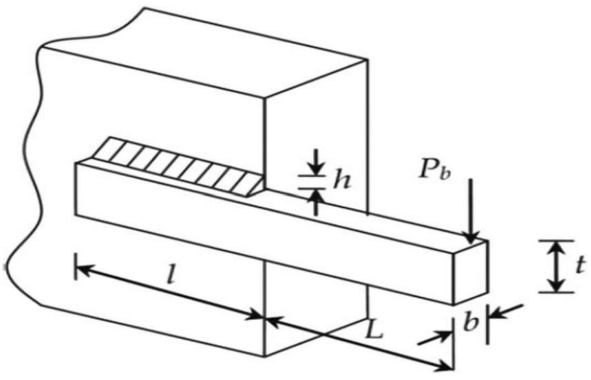

Figure 7. Welded beam design problem.

The optimal solutions and values achieved by each algorithm are presented in Table VI. The minimum cost data indicate that ABC_NPME outperforms all other competing algorithms.

TABLE VI: Optimal results obtained by ABC_NPME, GABC, GBABC, KFABC, LFABC, AABC and NABC

Optimum	ABC_NPME	GABC	GBABC	KFABC	LFABC	AABC	NABC
x_1	0.2054	0.2279	1.1129	0.1993	0.2975	0.2250	0.2035
x_2	3.2415	2.9119	4.7328	3.4006	2.7278	3.1078	3.2933
x_3	9.0358	8.8740	-4.9098	9.0315	7.4561	8.5517	9.0172
x_4	0.2058	0.2776	1.1565	0.2060	0.3125	0.2068	0.2068
$f(x)$	**1.6930**	1.8823	2.5104	1.6994	1.9180	2.9905	1.6995

V. Conclusion

In this paper, we presented an enhanced ABC algorithm, ABC_NPME, which combines the information from selected individuals to design a new probability selection mechanism, and proposed an adaptive movement equation based on multidimensional variation for onlooker bees. Numerical experiments demonstrated effectiveness of ABC_NPME. In the future, we will apply ABC_NPME to solve more complex practical problems

References

[1] D. Karaboga, "An idea based on honey bee swarm for numerical optimization", Technical Report-TR06, Erciyes University, Turkey, 2005.

[2] W.F. Gao, Z.F. Wei, Y.T. Luo, J. Cao, "Artificial bee colony algorithm based on Parzen window method", *Applied Soft Computing*, vol. 74, pp. 679-692, 2019.

[3] Huseyin Hakli, "A qualified search strategy with artificial bee colony algorithm for continuous optimization", *Arabian Journal for Science and Engineering*, vol. 45, no. 12, pp. 10891-10913, 2020.

[4] D. Zhang, Z.H. Zhan, J.J. Li, J. Zhang, "Tournament selection based artificial bee colony algorithm with elitist strategy", in: Technologies and Applications of Artificial Intelligence, Springer International Publishing, Cham, 2014.

[5] H. Gao, Z. Fu, C.M. Pun, J. Zhang, S. Kwong, "An efficient artificial bee colony algorithm with an improved linkage identification method", *IEEE Transactions on Cybernetics*, vol. 52, no. 6, pp. 4400-4414, 2022.

[6] C.F. Wang, P.P. Shang, P.P. Shen, "An improved artificial bee colony algorithm based on Bayesian estimation", *Complex & Intelligent Systems*, vol. 8, no. 6, pp. 4971-4991, 2022.

[7] S. Aslan, H. Badem, D. Karaboga, "Improved quick artificial bee colony (iqABC) algorithm for global optimization", *Soft Computing*, vol. 23, no. 24, pp. 13161-13182, 2019.

[8] X.Y. Zhou, J.X. Lu, J.H. Huang, M.S. Zhong, M.W. Wang, "Enhancing artificial bee colony algorithm with multi-elite guidance", *Information Science*, vol. 543, pp. 242-258, 2021.

[9] X.Y. Zhou,Y.L. Wu, M.S. Zhong, M.W. Wang, "Artificial bee colony algorithm based on multiple neighborhood topologies", *Applied Soft Computing*, vol. 111, Article Number:107697, 2021.

[10] W.F. Gao, S.Y. Liu, L.L. Huang, "Enhancing artificial bee colony algorithm using more information-based search equations", *Information Sciences*, vol. 270, pp. 112-133, 2014.

[11] W.F. Gao, S.Y. Liu, L.L. Huang, "A novel artificial bee colony algorithm based on modified search equation and orthogonal learning", *IEEE Transactions on Cybernetics*, vol. 43, pp.

[12] A. Alrosan, W. Alomoush, N. Norwawi, M. Alswaitti, S.N. Makhadmeh, "An improved artificial bee colony algorithm based on mean best-guided approach for continuous optimization problems and real brain MRI images segmentation", *Neural Computing and Applications*, vol. 33, no. 5, pp. 1671-1697, 2021.

[13] L.Z. Cui, G.H. Li, Q.Z. Lin, Z.H. Du, W.F. Gao, J.Y. Chen, N. Lu, "A novel artificial bee colony algorithm with depth-first search framework and elite-guided search equation", *Information Science*, vol. 367, pp. 1012-1044, 2016.

[14] R. Lu, H.D. Hud, M.L. Xi, H. Gao, C.M. Pun, "An improved artificial bee colony algorithm with fast strategy, and its application", *Computers and Electrical Engineering*, vol. 78, pp. 79-88, 2019.

[15] H. Peng, C.S. Deng, Z.J. Wu, "Best neighbor-guided artificial bee colony algorithm for continuous optimization problems", *Soft Computing*, vol. 23, no. 18, pp. 8723-8740, 2019.

[16] C.F. Wang, P.P. Shang, L.X. Liu, "Improved artificial bee colony algorithm guided by experience", *Engineering Letters*, vol. 30, no. 1, pp. 261-265, 2022.

[17] C.F. Wang, Y.H. Zhang, "An improved artificial bee colony algorithm for solving optimization problems", *IAENG International Journal of Computer Science*, vol. 43, no. 3, pp. 336-343, 2016.

[18] Z. Wang, X.Y. Kong, "An empirical balanced artificial bee colony algorithm", *IAENG International Journal of Computer Science*, vol. 51, no. 2, pp. 91-103, 2024.

[19] Z. Wang, X.Y. Kong, "An enhanced artificial bee colony algorithm for constraint optimization", *Engineering Letters*, vol. 32, no. 2, pp. 276-283, 2024.
[20] G.P. Zhu, S. Kwong, "Gbest-guided artificial bee colony algorithm for numerical function optimization", *Applied Mathematics and Compution*, vol. 217, no. 7, pp. 3166-3173, 2010.
[21] X. Zhou, Z. Wu, H. Wang, S. Rahnamayan, "Gaussian bare-bones artificial bee colony algorithm", *Soft Computing*, vol. 20, no. 3, pp. 907-924, 2016.
[22] H. Wang, W. Wang, X. Zhou, J. Zhao, Y. Wang, S. Xiao, M. Xu, "Artificial bee colony algorithm based on knowledge fusion", *Complex & Intelligent Systems*, vol. 7, no. 3, pp. 1139-1152, 2021.
[23] M. Alam, M. Islam, "Artificial bee colony algorithm with self-adaptive mutation: a novel approach for numeric optimization", In Tencon 2011-2011 IEEE Region 10 Conference, 2011.
[24] W.J. Yu, Z.H. Zhan, J. Zhang, "Artificial bee colony algorithm with an adaptive greedy position update strategy", *Soft Computing*, vol. 22, no. 2, pp. 437-451, 2018.
[25] H. Peng, C.S. Deng, Z.J. Wu, "Best neighbor-guided artificial bee colony algorithm for continuous optimization problems", *Soft Computing*, vol. 23, no. 18, pp. 8723-874, 2019.