Dynamic Artificial Bee Colony Algorithm with Hybrid Initialization Method

Sabreen Fawzi Raheem
Department of Computer Science, College of CSIT, University of Basrah, Basrah, Iraq
E-mail: sabreen.fawzi@uobasrah.edu.iq

Maytham Alabbas
Department of Computer Science, College of CSIT, University of Basrah, Basrah, Iraq
E-mail: ma@uobasrah.edu.iq

Keywords: dynamic ABC, good-point set, chaotic maps, self-adaptive population

Received: July 18, 2021

An improved basic artificial bee colony (ABC) algorithm with a self-adaptive technique, called dynamic ABC, is proposed. The dynamic ABC algorithm first uses a hybrid method combining the good-point-set method with the chaotic maps method to generate the initial population. Then, it applies self-adaptive population size at each generation, meaning that the population increases or decreases depending on some criteria to enhance global convergence and avoiding local solutions. Experiments are carried out on a range of 10 popular benchmark functions. The results indicate that the dynamic ABC algorithm is superior to the basic ABC algorithm when considering both the speed and quality of the solution obtained.

Povzetek: Dinamični ABC je predlagan za izboljšanje algoritma ABC, ki uporablja hibridno metodo za generiranje začetne populacije in uporablja samoprilagodljivo velikost populacije, ki izboljšuje globalno konvergenco in preprečuje lokalne rešitve.

1 Introduction

In optimization problems, the primary goal is to find the optimum solution among a range of possible solutions. In some situations, the solution area to a problem takes a long time to obtain the optimum solution. Moreover, there are different fields of computational intelligence that provide a collection of approaches for solving search and optimization problems. These approaches can produce highly competitive outcomes but not the best solutions. In addition, using heuristic algorithms, there are alternative approaches, but these do not ensure the optimum solution is found, although they can provide a successful solution in a sensible period of time.

Population-based algorithms are a new model of collective intelligence. They are capable of obtaining effective solutions to optimization problems at a reasonable cost and time, such as genetic algorithm (GA) (Agrawal & Jain, 2020), ant colony optimization (ACO) (Gu, Zhou, & Guo, 2019), particle swarm optimization (PSO) (Zhou et al., 2020), immune algorithm (IA) (L.-P. Zhang, Yang, He, & Cai, 2017), and electromagnetism-like algorithm (EM) (S.-J. Wang & Cui, 2018). A swarm-based stochastic optimization technique, namely the artificial bee colony (ABC) algorithm, is recently proposed by Karaboga (2005). The algorithm has been inspired by a bee's smart conduct in food search, and it has gained a lot of attention since it began. It has the benefit of having fewer control parameters, being simple to execute, easy integration with other algorithms, high robustness, and reliability. In fact, it has been vastly applied to solve various kinds of problems such as real-world problems and optimization problems since it is developed. It was applied to numerical optimization problems (Karaboga, 2005), neural networks (Karaboga & Akay, 2007), image processing (Benala, Villa, Jampala, & Konathala, 2009), scheduling problems (Yao, Yang, Hu, Yin, & Yu, 2010), optimize the BP neural network (Huang & Xia, 2017), it was used to estimate the effective software effort in software engineering (Khuat & Le, 2016), and recently it used to solve the shortest path problem with mixed interval-rated (Ebrahimnejad, Enayattabr, Motameni, & Garg, 2021).

The ABC algorithm has some limitations to its accuracy, slow converging speed, and it performs well at exploration but not well while exploiting (Zhu & Kwong, 2010). Furthermore, the ABC algorithm is prone to be stuck in the local optimal (Karaboga & Akay, 2009), although it benefits from the advantages listed above.

Different approaches for improving the performance of the ABC algorithm have been suggested in the literature. Some of these approaches generate the initial population using mathematical theories instead of the random initialization method, which follows the standard ABC algorithm. Using such techniques gives various advantages, such as irregularity and ergodicity, to generate candidate solutions to improved global convergence and avoid local solutions. In this regard, various techniques are applied, such as the chaotic maps approach (Alatas, 2010; Kong, Liu, & Wang, 2013), opposition-based learning method (Che, Yang, & Wang, 2021; Rahnamayan, Tizhoosh, & Salama, 2008), combine an opposition-based learning method and chaotic systems (W.-f. Gao & Liu, 2012; Long & Wang, 2021), a good-point-set strategy (Chun-Feng, Kui, & Pei-Ping, 2014; Li, Xiang, Yang, & Liu, 2021; Ouyang, Li, Fei, Zhou, & Duan, 2015; R. Wang, Ru, & Long, 2009; Q. Zhang & Liu, 2019), half space-based symmetry initialization (HSSI) algorithm (Xue, Jiang, Ma, Liu, & Pang, 2018), and k-means method (Pan, Zhang, & Pan, 2020). The latter has been developed to the generated cluster center as the initial honey source rather than the regular method initialization to improve the solution's diversity and to improve the solution's diversity and performance.

Alternatively, a set of techniques is proposed to improve the ABC algorithm by modifying the solution search equation. Zhu and Kwong (2010) proposed a gbest-guided ABC (GABC) algorithm, which is an improved ABC algorithm that enhances the exploitation by utilizing the global best solution (gbest) details to guide the quest of new candidate solutions through incorporating the information of gbest into the equation of solution search. W. Gao, Liu, and Huang (2012) suggested a selective probability and a new search mechanism, namely ABC/best/1, that can generate the new solution around the best solution of the previous cycle. Also, W. Gao and Liu (2011) introduced a modified ABC algorithm, inspired by DE, in which "ABC/rand/1" and "ABC/best/1" were employed as local search mechanisms. Tang, Long, Wu, Zhang, and Shardt (2017) improved a solution search equation by introduces a parameter λ that coordinates the impact of the previous position on the new one. N. A. Elkhateeb and Badr (2014) presented a new dynamic parameter named inertia weight, which helped control the effect of previous foods on the current anticipated one.

Other approaches have modified the size of the population by making it dynamic instead of being fixed in the basic ABC algorithm. The population size plays an essential role in the strength of the algorithm and the cost of its calculation. T. K. Sharma, Pant, and Singh (2011) proposed a variant to generate food sources adaptively by starts with some initial randomly generated solutions while the population of the food source of subsequent generations is adaptively changing depending on the average size of the population of all individuals in the current population. An incremental population size strategy was proposed, adding new individuals to the population biased towards the best-so-far solution (De Oca, Stutzle, Van den Enden, & Dorigo, 2010; Liao, Montes de Oca, Aydin, Stützle, & Dorigo, 2011). N. Elkhateeb and Badr (2017) introduced a novel variable population size ABC algorithm that gradually reduces the population size and moves the bees, in each process of re-initialization, towards the global best food source. Ding et al. (2017) proposed a novel ABC algorithm with a dynamic population that the bee can dynamically reproduce and die throughout the foraging process and the size of the population varies at each generation. Table 1 summarizes the related works with their methodology, performance, and results.

So, in the current work, two different mechanisms are proposed to enhance the intensification and diversification of the ABC algorithm's performance. First, a hybrid initialization technique based on good-point-set theory and chaotic maps method to generate the initial population is used. Second, different strategies for generating a self-adaptive population size are applied because the algorithm's effectiveness can be maintained by maintaining acceptable population size.

The rest of the current paper is structured as follows: Section 2 explains a brief overview of the basic ABC algorithm. Section 3 describes the main idea of the dynamic ABC algorithm. The test functions, the results of the simulation, and comparison results are discussed in Section 4. Section 5 presents a summary discussion.

2 Artificial bee colony algorithm (ABC)

2.1 The basic concept of the ABC algorithm

Karaboga (2005) suggested the artificial bee colony (ABC) algorithm as a population-based search technique dependent on the intelligent foraging nature of a honeybee hive. The ABC algorithm follows four steps: (1) initialization; (2) employed bee phase: working bees returning to a previously visited food supply (a potentially optimized solution to the problem); (3) onlooker bee phase: some onlookers are waiting to choose a food supply; and (4) scout bee phase: scouts conduct a random quest. The working artificial bees make up the first half of the colony, while onlookers and scouts make up the second half. In other words, the number of bees working equals the number of food supplies. The employed bee (forager bee) of an abandoned food source is transformed into a scout. Initialization happens first and only one time, while the other three steps are replayed repeatedly before the stop conditions are satisfied. The main steps of the ABC algorithm are as follows.

Initialization step: Through the current step, an initial population of SN solutions (food sources) is randomly generated according to Eq. (1).

$$X_{ij} = Lb_j + rand\ (0,1) * \left(Ub_j - Lb_j\right), \qquad (1)$$

where $X_{i=1\ldots,SN} = \{x_{i1}, x_{i2}, \ldots, x_{iD}\}$ denote the i^{th} solution in the swarm, D is the number of optimization parameters, and $\mathrm{Lb_j}$ and $\mathrm{Ub_j}$ are the lower and upper bounds of the solution location in dimension j, respectively. Then each individual is evaluated by Eq. (2).

$$fitness_i = \begin{cases} \frac{1}{1+f(X_i)} & if\ f(X_i) \geq 0, \\ 1 + abs\ (X_i) & if\ \ f(X_i) < 0, \end{cases} \qquad (2)$$

where $fitness_i$ is the fitness value of solution i, and $f(X_i)$ is the objective function associated with X_i.

Reference	Methodology	Performance/Results
(Alatas, 2010)	• Modify the initial population	• Three benchmark functions (one unimodal and two multimodal) are used to measure the performance of the proposed technique. • The proposed technique results showed an increase in the quality of the solution; that is, it improved in some cases the global search ability by escaping from local solutions.
(Kong, Liu, & Wang, 2013)	• Modify the initial population • Developed a new search equation	• The proposed algorithm was applied to test 27 standard functions with different dimensions to verify their performance. • The search equation has been modified to balance the exploration and exploitation capabilities well and preserve society's diversity. • The proposed configuration methods affected the quality of solutions and the speed of convergence.
(Che, Yang, & Wang, 2021)	• Modify the initial population • Developed a new search mechanism	• The proposed algorithm is used to cluster identification in two-mode graphs with two kinds of vertices in the cluster. • To validate the proposed method's findings, many tests are conducted using synthetic and actual bipartite graphs. • The experiments demonstrate that this method is an effective candidate for cluster identification in a two-mode network.
(W.-f. Gao & Liu, 2012)	• Modify the initial population • Modify the search mechanism	• A set of 28 benchmark functions are used to evaluate the proposed MABC technique. • The findings show that MABC outperforms two ABC-based algorithms in addressing difficult numerical optimization problems.
Tang et al. (2017)	• Modify the initial population • Modified search equation.	• A set of 12 benchmark functions are used to evaluate the proposed SABC technique. • The results show that the proposed SABC algorithm outperforms many existing ABC versions in terms of search capability.
(Xue, Jiang, Ma, Liu, & Pang, 2018)	• Modify the initial population • Modified search equation.	• A set of 25 benchmark functions are used to evaluate the proposed SABC-SI technique. • In experiments, the SABC-SI beats many state-of-the-art algorithms, including GABC, NSHS, and ABC/best/1. This suggests that it has considerable promise for application to a wide variety of optimization issues.
(Zhu and Kwong (2010))	• Modified search equation.	• The experimental findings on six benchmark functions demonstrate that the GABC method outperforms the ABC algorithm when the proper parameters are used.
(W. Gao, Liu, and Huang (2012))	• Modified search equation.	• The experimental findings on 26 benchmark functions demonstrate that the ABC/best/1 algorithm beats the ABC and GABC algorithms.
(W. Gao and Liu (2011))	• Modified search equation.	• The experiments are carried out on a collection of unimodal/multimodal benchmark functions. • The findings show that the IABC method outperforms 13 algorithms in addressing difficult numerical optimization problems, including standard DE, standard PSO, standard ABC, OEA, HPSO-TVAC, CLPSO, APSO, SaDE, jDE, JADE, CABC, GABC, and RABC.
(N. A. Elkhateeb and Badr (2014))	• Modified search equation.	• The inertia weight ABC method was compared to the conventional ABC, PSO, and GA algorithms for tuning proportional-integral-derivative (PID) controllers for the ball and hoop system, which depicts a collection of complicated industrial processes are known to be nonlinear and time variable. • The simulation results demonstrate that the inertia weight ABC method is very competitive, often beating PSO and GA algorithms and attaining a higher convergence rate than the conventional ABC algorithm.
T. K. Sharma, Pant, and Singh (2011)	• Adaptive colony size	• The new algorithms are verified using a collection of common benchmark problems of various dimensions from the literature. • The numerical results are compared to those of the basic ABC and its latest version, Gbest ABC, to demonstrate the suggested algorithms' competency.
Aydın & Yavuz, (2016)	• Incremental population size	• The proposed algorithm, SSEABC, was tested on 19 benchmark functions. The authors compared the performance of the proposed algorithm with five ABC variations and 15 SOCO participant algorithms. • SSEABC is more effective than the examined ABC variations and is highly competitive with the 15 SOCO rival algorithms, according to the comparative findings.
N. Elkhateeb and Badr (2017)	• Reducing the population size	• To demonstrate the effectiveness of the VPS-ABC method, it is compared to the traditional ABC, PSO, and GA algorithms while adjusting proportional-integral-derivative (PID) controllers. • The simulation results demonstrate that the VPS-ABC method is very competitive, often beating both the PSO and GA algorithms.

Table 1: Summarization table on the related works.

Employed bee step: Each working bee X_i here generates a new candidate solution V_i in the neighborhood of its present position as in Eq. (3).

$$V_{ij} = X_{ij} + \varphi_{ij} * \left(X_{ij} - X_{kj} \right), \tag{3}$$

where j is a random dimension index, X_k is a random solution *(i≠k)*, and φ is a random number ∈[-1 1].

If V_{ij} is greater than the previous position, the new solution in the bee's memory replaces the previous; otherwise, the previous one's position is maintained. When the working bees have finished their search, they waggle dance to remind the onlooker bees about their food sources.

Onlooker bee step: Here, an onlooker bee analyses the nectar data collected from all working bees and selects an individual with a probability proportional to its nectar number. This probabilistic selection is a mechanism for selecting roulette wheels, defined as in Eq. (4).

$$P_i = \frac{fitness_i}{\sum_{i=1}^{SN} fitness_i}, \tag{4}$$

where P_i is the probability value of the food source, *fitness*$_i$ is the fitness value of the individual *i*. Eq. (2) is used to calculate a fitness value for a minimization problem.

Scout bee step: In this step, any food source not improving above a specific 'limit' of trials is abandoned and replaced by a new location, which serves as a scout for the bee employed, using Eq. (1).

Algorithm 1: Summarizes the steps of the ABC algorithm (Karaboga, 2005).

3 The improved artificial bee colony algorithm

The current work makes some improvements to the ABC algorithm to maximize its performance.

3.1 Initialization population

In metaheuristic optimization methods, population initialization is a significant and sensitive step. Indeed, starting with a suitable initial population increases the chance of finding a good solution and speeds up convergence. On the contrary, starting from a poor initial population decreases the opportunity to reach a better solution or requires a substantial number of cycles to achieve a good solution. In the basic ABC algorithm, the stochastic method is used to generate the initial population. These solutions may be good or bad for the problem and not cover the whole search space. To improve the initial population uniformity, several techniques to initialize the population are proposed in the literature, such as the good-point-set method (Hu, Tian, Guo, & Ouyang, 2015; H. Liu, Cai, & Wang, 2007; J. Liu & Wang, 2014; Tang et al., 2017; C.-f. Wang, Liu, & Shen, 2020) and chaotic maps method (Alatas, 2010) but still, these techniques cannot produce as many excellent solutions as possible close to the global optimal one. Therefore, the strategy of hybridizing the good-point-set method with the chaotic maps method is adopted to generate the initial population to ensure that most of the individuals in the population are evenly dispersed to improve the population diversity and cover the whole search space and accelerating convergence speed, as shown by Algorithm 2. In the current work, the circle map is selected by using Eq (5).

Algorithm 1: Main steps of the basic ABC algorithm

```
    SN size of the population.
    D number of optimization parameters.
    $x_{ij}$  solution$_{i,j}$,  $i = 1...SN$,  $j = 1... D$

Initialize the population of solutions $x_{i,j}$ , $trial_i = 0$,  by using equation (1);
Evaluate the population;
cycle = 1;
Repeat
Produce new solutions $v_{i,j}$ for the employed bees using equation (3) and evaluate its quality;
Apply the greedy selection process between the new solutions ($v_{i,j}$) and the old solution ($x_{i,j}$) and select the better one.
Calculate the probability values by using equation (4) for the solutions xi;
Produce the new solutions $v_{i,j}$ for the onlookers from the solutions xi selected depending on a pi and evaluate them;
Apply the greedy selection process for the onlookers;
Determine the abandoned solution for the scout, if exists, and replace it with a new randomly produced solution xi by using equation (1);
Memorize the best solution achieved so far;
cycle = cycle+1;
Until (cycle = maximum cycle number);
```

Algorithm 2: A hybrid population initialization method

```
P represents the minimum prime number content with P ≥ 2*D +3.
For i=0 to SN do
    For j=0 to D do
        If i mod 2 == 0   do
            (# Good-point-set method #)
            $GPS_{i,j} = i * 2 * \cos\left(2 * \pi * \frac{j}{P}\right)$
            $X_{ij} = Lb_j + GPS_{i,j} * \left(Ub_j - Lb_j\right)$
        Else
            (# chaotic maps (Circle) method #)
        Initialize variables $Ch_{0,j} \in (0,1)$ at random
    $Ch_{z+1} = Ch_z + 1.2\left(\frac{0.5}{2\pi}\right)\sin(2\pi\, Ch_z) mod\ (1)$
        $X_{ij} = Lb_j + Ch_{z,j} * \left(Ub_j - Lb_j\right)$
        End for
End for
```

$$Ch_{z+1} = Ch_z + 1.2 - \left(\frac{0.5}{2\pi}\right)\sin(\,2\pi\; Ch_z) mod\,(1), \qquad (5)$$

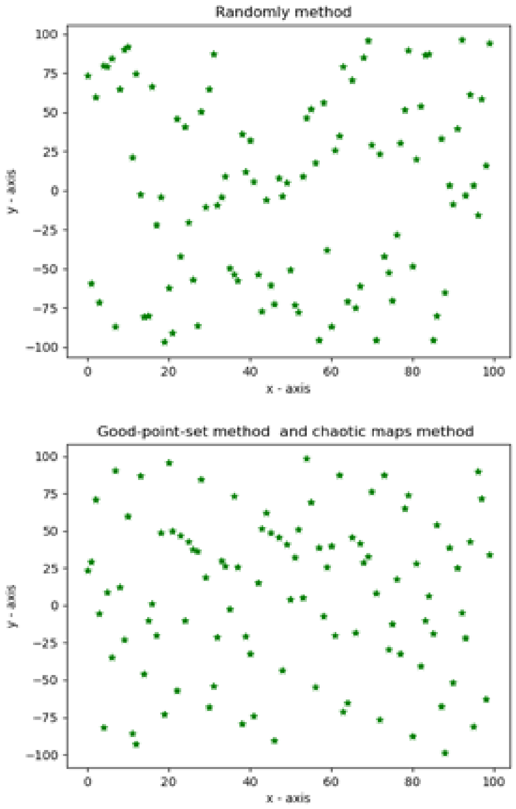
where $Ch_z \in (0,1)$, $z = 0,1,2\ldots, Z$, is the chaotic sequence, $Z = SN * D$.

To illustrate that hybrid method sampling is better than random method sampling, the uniformity properties of both methods are compared as given in Figure 1. We display 100 points, in the range [−100, 100], on a unit square generated using a uniform random number generator and the current hybrid approach, respectively.

Obviously, from Figure 1, the random method distribution is scattered, as shown in Figure 1 (left), but the hybrid method distribution is relatively more uniform, as shown in Figure 1 (right). Thus, the hybrid method technique is the preferred initialization technique for generating a good initial population instead of a purely random one.

3.2 Modified Generating Equation

In the basic ABC algorithm, the new individual is produced by adopting Eq. (3) by moving the individual towards another randomly selected individual from the population that can be a good or bad individual. As a result, the current candidate individual does not seem to be different from the previous one. Therefore, the search for bees is adept in exploration but weak in exploitation. So, in the current work, the following equation proposed by (Liang & Lee, 2015) is used to cope with the above challenge.


Randomly method

Good-point-set method and chaotic maps method

Figure 1: Illustrates the difference between the generation of the initial population using the random method and the hybrid good-point-set and chaotic maps method.

$$V_{ij} = X_{ij} + \varphi_1 * \left(X_{kj} - X_{ij}\right) + \varphi_2 * \left(X_{best,j} - X_{ij}\right), \qquad (6)$$

where $\varphi_1 \in [-1,1]$ and $\varphi_2 \in [0,1]$, and X_k is randomly chosen from the population provided that $(k \neq i)$ and X_{best} is the best solution in the current population.

3.3 Dynamic population size

Population size is very significant in evolutionary methods; it also influences the robustness and the algorithm computing cost. Small sizes of the population can lead to local convergence, whereas large sizes of the population can lead to slow convergence due to increased computational efforts. As a result, sufficient population size will preserve the algorithm's efficacy. We explore a range of techniques for generating a self-adaptive population size by adopting several computational methods, as shown below.

3.3.1 DABC1

In this method, we have relied on the principle of increasing population size, as the increasing population size approach has been shown to improve algorithm efficiency (Aydın & Yavuz, 2016). The principle of increment strategy is adopted to generate the size of a self-changing population. Therefore, the DABC1 starts with a small number of individuals at the beginning. Then in each cycle, the population size gradually increases until the population reaches a predetermined maximum size, depending on the diversity of the population.

The average distance around the swarm center (H. Sharma, Bansal, & Arya, 2013) method is applied to calculate the population diversity. Firstly, calculating the center of the population depends on an average of all food sources in the population using Eq. (7).

$$C_i(k) = \frac{\sum_{j=0}^{SN} X_j(k)}{SN}. \qquad (7)$$

Where the center population is defined as form following:

$$C_i = [C_i(0)\,, C_i(1)\,, C_i(2)\,, \cdots, C_i(D)]$$

Then, the average distance between each individual in the population with the center C is calculated using Eq. (8).

$$div\_pop1 = \frac{1}{SN}\sum_{i=0}^{SN}\left(\sum_{i=0}^{D}|X_i - C_i|\right), \qquad (8)$$

where div-pop1 is the diversity of the population.
Finally, the new population size is calculated using Eq. (9), which is dependent on the ratio between the diversity of the population at generations i and i-1.

$$SN_i = SN_{i-1} + \frac{div\_pop1_i}{div\_pop1_{i-1}}, \qquad (9)$$

where SN_i is the new population size, $div\text{-}pop1_i$ and $div\text{-}pop1_{i-1}$ are the diversity of populations in generations i and i-1, respectively.

3.3.2 DABC2

Throughout the foraging process, the bee can reproduce and die adaptively, and the size of the population varies at each generation (Ding et al., 2017). So, in the current method, an attempt is to generate population size adaptively change with each cycle, meaning that the size of the population increases or decreases after each cycle, depending on the population diversity. The measurement of the population diversity is calculated based on phenotype properties using Eq. (10).

$$div-pop2_i = \frac{fit_{Max,i} - fit_{avg,i}}{fit_{Max,i}}, \tag{10}$$

where $fit_{Max,i}$ is the maximum fitness value in generation i, $fit_{avg,i}$ is the average fitness value in generation i and the variable $div\text{-}pop2_i$ measures the diversity of the population in generation i that belong to the interval [0,1]. This technique requires less time than the diversity measurement, which is used in Eq. (8).

$$SN = \begin{cases} SN+2, & if\ div\_pop2 \leq 0.5 \\ SN-2, & if\ div\_pop2 > 0.5 \end{cases}. \tag{11}$$

That is, whenever the value of *div_pop2* is large, the diversity is high and vice versa. In the current strategy, population size increases and decreases depending on the diversity; if the diversity is high (> 0.5), the population size decreases by 2; if the diversity is low (≤ 0.5), the population size increases by 2.

3.3.3 DABC3

In this method, the population size increases and decreases depending on the diversity but with a specific number of individuals K as in Eq. (12).

$$SN_i = SN_{i-1} - [sign\ (div\_pop1_i\ -\ div\_pop1_{i-1}\ )]\ * K, \tag{12}$$

where SN_i is the size of the population in generation i, and K is the number of added/removed individuals, here $K=2$. The diversity in this strategy is calculated by Eq. (8).

3.3.4 DABC4

In the current strategy, the population size is adaptively increased or decreased depending on several successive cycles that the best individual is unchanged. So, for each N number of successive cycles, if the best individual is not improved, the population size is increased by 2; if the best individual is improved more than N/2 times, the population size decreases by 2; otherwise, the population size is unchanged.

4 Experiments

4.1 Test functions and parameter setting:

4.1.1 Benchmark functions

A set of 10 benchmark optimization problems presented in Table 2 is utilized to validate the performance of the systems discussed in Sec. 3. The first five of these functions are unimodal, while the rest are multimodal. In Table 2, the function name, type, search range, optimum, and formulation are listed. Where U: unimodal, M: multimodal, S: separable, and N: non-separable.

4.1.2 Parameter settings

The following were the primary control parameters in the current work:

The colony (swarm) size: NP = 80 (number of employed bees is equal to the number of onlookers = 40).

The number of food sources (SN) = 40.

The number of cycles that a food supply cannot be changed through: limit = SN*D.

A stopping criterion for foraging is the number of cycles: maximum cycle = 5000.

The elite number is k = {1,2,4} individuals.

The running time is 30.

The Schafer and Rosenbrock functions were evaluated with D = {10,30}, the Himmelblau function with D = {100,200}, and the other functions with D = {30,60}.

4.2 Experimental results

Two experiments were conducted to evaluate the performance of the current work.

4.2.1 Experiment A

This experiment was conducted to evaluate the performance of the hybrid technique against the random technique to generate the initial population. We experimented and tested these two techniques on five benchmark functions (f1, f2, f8, f9, and f10) to discover their strengths and weaknesses. The results of this experiment are summarized in Table 3.

Function #	D	Randomly method		Hybrid method (chaotic +GPS)	
		Mean	SD	Mean	SD
f_1	10	2.08×10^{-1}	1.16×10^{-1}	**1.26×10^{-1}**	7.25×10^{-2}
	30	**2.72×10^{-1}**	2.86×10^{-1}	3.25×10^{-1}	3.69×10^{-1}
f_2	30	1.08×10^{-15}	2.23×10^{-16}	**9.70×10^{-16}**	1.48×10^{-16}
	60	8.99×10^{-15}	5.93×10^{-15}	**7.08×10^{-15}**	4.11×10^{-15}
f_8	30	7.45×10^{-14}	6.54×10^{-14}	**6.16×10^{-14}**	5.05×10^{-14}
	60	3.19×10^{-10}	4.44×10^{-10}	**1.48×10^{-11}**	2.84×10^{-11}
f_9	30	**1.84×10^{-13}**	3.69×10^{-13}	9.02×10^{-13}	2.48×10^{-12}
	60	1.72×10^{-12}	4.26×10^{-12}	**9.56×10^{-13}**	1.28×10^{-12}
f_{10}	30	1.32×10^{-13}	2.96×10^{-14}	**1.20×10^{-13}**	2.65×10^{-14}
	60	1.83×10^{-12}	1.50×10^{-12}	**1.45×10^{-12}**	5.75×10^{-13}

Table 2: Shows the mean and standard deviation values obtained in Experiment A.

4.2.2 Experiment B

To test the performance of the four strategies (DABC1, DABC2, DABC3, and DABC4) proposed by the current work, we compared the results of the basic ABC algorithm on ten benchmark functions (see Table 2). The results, in terms of the mean and standard deviation of the solutions obtained in the 30 runs by each algorithm, are shown in Table 4. The convergence performance of four strategies and the basic ABC algorithm on ten benchmark functions are presented in Figure 2 to show the performance of four DABCs more clearly.

5 Discussion

In the current section, the effects of each modification on the performance of the ABC algorithm are discussed. Firstly, the performance of the hybrid good-point-set with chaotic maps technique against the random technique to generate the initial population is tested. As shown in Table 3, the results indicate that the hybrid method for generating the initial population is superior to the pure random one. We, therefore, use hybrid technology in the current dynamic ABC systems. After that, four modification systems of the basic ABC algorithm, namely DABC1, DABC2, DABC3, and DABC4, are evaluated on 10 test popular benchmark functions. We compare the mean and standard deviation values of these for systems, DABC1-DABC4, with these obtained by the basic ABC algorithm through 30 runs on tested functions to find the contributions of the proposed systems to improve the performance of the basic ABC algorithm. The corresponding simulation results are shown in Table 4. We can see that the four strategies outperform the basic ABC algorithm. As for the unimodal functions shown in Figure 2 (1-5), the performance of the four strategies outperforms the basic ABC algorithm, which means that the proposed modification systems have a positive effect on the algorithm's convergence rate. Again, for the multimodal functions shown in Figure 2 (6-10), we also note the superiority of the strategies over the basic ABC algorithm, which achieves good convergence speed and their ability to global optimization.

There are many works in this regard, but comparing them is very difficult because each work used a different set of tested benchmark functions. We, therefore, will compare our results with the recent previous works that used all or some of the current test functions.

The current work outperforms the other works for most of the test functions. For f1 function, DABC achieves better results than (Kong et al., 2013), (W.-f. Gao & Liu, 2012), (Zhu & Kwong, 2010), (W. Gao & Liu, 2011), and (Ruan, Wang, Zhang, Liu, & Fu, 2020). For f2 function, DABC achieves better results than (Kong et al., 2013) and (Zhu & Kwong, 2010) except (W.-f. Gao & Liu, 2012), (Ruan, Wang, Zhang, Liu, & Fu, 2020), and (W. Gao & Liu, 2011), which achieves the best results on this function. For f3 function, DABC achieves better results than (Ruan, Wang, Zhang, Liu, & Fu, 2020). For f4 function, DABC achieves better results than (Kong et al., 2013) except (Ruan, Wang, Zhang, Liu, & Fu, 2020), which achieves the best results on this function. For f5 function, DABC achieves better results than (Ruan, Wang, Zhang, Liu, & Fu, 2020). For f6 function, DABC achieves similar results than (W. Gao & Liu, 2011) except (Ruan, Wang, Zhang, Liu, & Fu, 2020), which achieves the best results on this function. For f7 function, DABC achieves better results than (W.-f. Gao & Liu, 2012) and (Ruan, Wang, Zhang, Liu, & Fu, 2020). For f8 function, DABC achieves better results than (Zhu & Kwong, 2010) except (Kong et al., 2013), (W. Gao & Liu, 2011), and (Ruan, Wang, Zhang, Liu, & Fu, 2020), which achieve the similar results on this function. For f9 function, DABC achieves better results than (Kong et al., 2013) and (Zhu & Kwong, 2010) except (W.-f. Gao & Liu, 2012) and (W. Gao & Liu, 2011), which achieve similar results on this function while (Ruan, Wang, Zhang, Liu, & Fu, 2020), which achieves the best results on this function. Finally, for f10 function, DABC achieves better results than (W.-f. Gao & Liu, 2012), (Zhu & Kwong, 2010), (W. Gao & Liu, 2011), and (Ruan, Wang, Zhang, Liu, & Fu, 2020) except (Kong et al., 2013), which achieves the best results on this function.

6 Conclusions

In the current work, we have proposed different strategies to increase the basic ABC algorithm performance. For overcoming the ABC algorithm’s disadvantage, we adopted the hybrid good-point-set method and chaotic maps method to enhance the initial population distribution in the early stage; then, we also develop four self-adaptive mechanisms, namely DABC1, DABC2, DABC3, and DABC4, to dynamically determine the appropriate size of the population to performance improvement. These algorithms have been tested on ten well-known benchmark functions. The simulation results show that the current algorithms exhibit a magnificent performance that can offer solutions with high accuracy and strong robustness and outperforms the conventional ABC algorithm.

We plan to improve the ABC algorithm performance using other techniques such as rough sets and fuzzy logic in future work.

References

[1] Agrawal, M., & Jain, V. (2020). *Applying Improved Genetic Algorithm to Solve Travelling Salesman Problem.* Paper presented at the 2020 Second International Conference on Inventive Research in Computing Applications (ICIRCA). https://doi.org/10.1109/ICIRCA48905.2020.9182884

[2] Alatas, B. (2010). Chaotic bee colony algorithms for global numerical optimization. *Expert systems with applications, 37*(8), 5682-5687. https://doi.org/10.1016/j.eswa.2010.02.042

[3] Aydın, D., & Yavuz, G. (2016). *A Self-adaptive Artificial Bee Colony Algorithm with Incremental Population Size for Large Scale Optimization.* Paper presented at the International Conference on Soft Computing-MENDEL.

[4] Benala, T. R., Villa, S. H., Jampala, S. D., & Konathala, B. (2009). *A novel approach to image edge enhancement using artificial bee colony optimization algorithm for hybridized smoothening filters.* Paper presented at the 2009 World Congress

on Nature & Biologically Inspired Computing (NaBIC). https://doi.org/10.1109/NABIC.2009.5393866

[5] Che, S., Yang, W., & Wang, W. (2021). An Improved Artificial Bee Colony Algorithm for Community Detection in Bipartite Networks. *IEEE Access, 9*, 10025-10040. https://doi.org/10.1109/ACCESS.2021.3050752

[6] Chun-Feng, W., Kui, L., & Pei-Ping, S. (2014). Hybrid artificial bee colony algorithm and particle swarm search for global optimization. *Mathematical Problems in Engineering, 2014*, 8 pages. https://doi.org/10.1155/2014/832949

[7] De Oca, M. A. M., Stutzle, T., Van den Enden, K., & Dorigo, M. (2010). Incremental social learning in particle swarms. *IEEE Transactions on Systems, Man, and Cybernetics, Part B, 41*(2), 368-384. https://doi.org/10.1109/TSMCB.2010.2055848

[8] Ding, M., Chen, H., Lin, N., Jing, S., Liu, F., Liang, X., & Liu, W. (2017). Dynamic population artificial bee colony algorithm for multi-objective optimal power flow. *Saudi Journal of Biological Sciences, 24*(3), 703-710. https://doi.org/10.1016/j.sjbs.2017.01.045

[9] Ebrahimnejad, A., Enayattabr, M., Motameni, H., & Garg, H. (2021). Modified artificial bee colony algorithm for solving mixed interval-valued fuzzy shortest path problem. *Complex and Intelligent Systems*, 1-19.

[10] Elkhateeb, N., & Badr, R. (2017). A novel variable population size artificial bee colony algorithm with convergence analysis for optimal parameter tuning. *International Journal of Computational Intelligence and Applications, 16*(03), 1750018. https://doi.org/10.1142/S1469026817500183

[11] Elkhateeb, N. A., & Badr, R. (2014). Dynamic inertia weight artificial bee colony versus GA and PSO for optimal tuning of PID controller. *22*(4), 307-317. https://doi.org/10.1504/IJMIC.2014.066262

[12] Gao, W.-f., & Liu, S.-y. (2012). A Modified Artificial Bee Colony Algorithm. *Computers and Operations Research, 39*(3), 687-697. https://doi.org/10.1016/j.cor.2011.06.007

[13] Gao, W., & Liu, S. (2011). Improved artificial bee colony algorithm for global optimization. *Information Processing Letters, 111*(17), 871-882. https://doi.org/10.1016/j.ipl.2011.06.002

[14] Gao, W., Liu, S., & Huang, L. (2012). A global best artificial bee colony algorithm for global optimization. *Journal of Computational and Applied Mathematics, 236*(11), 2741-2753. https://doi.org/10.1016/j.cam.2012.01.013

[15] Gu, S., Zhou, X., & Guo, Q. (2019). *Denoising of Power Quality Disturbance Signal Based on Ant Colony optimization Wavelet Threshold Estimation.* Paper presented at the 2019 IEEE 3rd Information Technology, Networking, Electronic and Automation Control Conference (ITNEC). https://doi.org/10.1109/ITNEC.2019.8729061

[16] Hu, H., Tian, S., Guo, Q., & Ouyang, A. (2015). An adaptive hybrid PSO multi-objective optimization algorithm for constrained optimization problems. *International Journal of Pattern Recognition and Artificial Intelligence, 29*(06), 1559009. https://doi.org/10.1142/S0218001415590090

[17] Huang, H., & Xia, X.-L. (2017). *Wine quality evaluation model based on artificial bee colony and BP neural network.* Paper presented at the 2017 International Conference on Network and Information Systems for Computers (ICNISC). https://doi.org/10.1109/ICNISC.2017.00026

[18] Karaboga, D. (2005). *An idea based on honey bee swarm for numerical optimization*. Retrieved from TECHNICAL REPORT-TR06.

[19] Karaboga, D., & Akay, B. (2007). *Artificial bee colony (ABC) algorithm on training artificial neural networks.* Paper presented at the 2007 IEEE 15th Signal Processing and Communications Applications. https://doi.org/10.1109/SIU.2007.4298679

[20] Karaboga, D., & Akay, B. (2009). A comparative study of artificial bee colony algorithm. *Applied Mathematics and Computation, 214*(1), 108-132. https://doi.org/10.1016/j.amc.2009.03.090

[21] Khuat, T. T., & Le, M. H. (2016). Optimizing parameters of software effort estimation models using directed artificial bee colony algorithm. *Informatica, 40*(4).

[22] Kong, X., Liu, S., & Wang, Z. (2013). A new hybrid artificial bee colony algorithm for global optimization. *International Journal of Computer Science Issues, 10*(1), 287.

[23] Li, H., Xiang, S., Yang, Y., & Liu, C. (2021). Differential evolution particle swarm optimization algorithm based on good point set for computing Nash equilibrium of finite noncooperative game [J]. *AIMS Mathematics, 6*(2), 1309-1323.

[24] Liao, T., Montes de Oca, M. A., Aydin, D., Stützle, T., & Dorigo, M. (2011). *An incremental ant colony algorithm with local search for continuous optimization.* Paper presented at the Proceedings of the 13th annual conference on Genetic and evolutionary computation.

[25] Liu, H., Cai, Z., & Wang, Y. (2007). *A new constrained optimization evolutionary algorithm by using good point set.* Paper presented at the 2007 IEEE Congress on Evolutionary Computation. https://doi.org/10.1109/CEC.2007.4424613

[26] Liu, J., & Wang, Y.-p. (2014). An improved central force optimization algorithm for multimodal optimization. *Journal of Applied Mathematics, 2014*, 12 pages. https://doi.org/10.1155/2014/895629

[27] Long, M., & Wang, L. (2021). S-box design based on discrete chaotic map and improved artificial bee colony algorithm. *IEEE Access, 4*, 1-11. https://doi.org/10.1109/ACCESS.2021.3069965

[28] Ouyang, A., Li, K., Fei, X., Zhou, X., & Duan, M. (2015). A novel hybrid multi-objective population migration algorithm. International Journal of Pattern Recognition and Artificial Intelligence (01), 1559001. https://doi.org/10.1142/S0218001415590016

[29] Pan, X., Zhang, Q., & Pan, H. (2020). Improved artificial bee colony algorithm and its application to fundus retinal blood vessel image binarization. *IEEE Access, 8*, 123726-123734. https://doi.org/10.1109/ACCESS.2020.3001299

[30] Rahnamayan, S., Tizhoosh, H. R., & Salama, M. M. (2008). Opposition-based differential evolution. *IEEE Transactions on Evolutionary computation, 12*(1), 64-79. https://doi.org/10.1109/TEVC.2007.894200

[31] Sharma, H., Bansal, J. C., & Arya, K. (2013). *Diversity Measures in Artificial Bee Colony*. Paper presented at the Proceedings of Seventh International Conference on Bio-Inspired Computing: Theories and Applications (BIC-TA 2012).

[32] Sharma, T. K., Pant, M., & Singh, V. P. (2011). *Artificial bee colony algorithm with self adaptive colony size.* Paper presented at the International Conference on Swarm, Evolutionary, and Memetic Computing.

[33] Tang, M., Long, W., Wu, H., Zhang, K., & Shardt, Y. A. W. (2017). Self-Adaptive Artificial Bee Colony for Function Optimization. *Journal of Control Science and Engineering, 2017*, 13 pages. https://doi.org/10.1155/2017/4851493

[34] Wang, C.-f., Liu, K., & Shen, P.-p. (2020). A Novel Genetic Algorithm for Global Optimization. *Acta Mathematicae Applicatae Sinica, English Series, 36*(2), 482-491.

[35] Wang, R., Ru, Y., & Long, Q. (2009). Improved adaptive and multi-group parallel genetic algorithm based on good-point set. *Journal of Software, 4*(4), 348-356.

[36] Wang, S.-J., & Cui, X.-L. (2018). *A Practical Electromagnetism-Like Mechanism Algorithm.* Paper presented at the 2018 2nd IEEE Advanced Information Management, Communicates, Electronic and Automation Control Conference (IMCEC). https://doi.org/10.1109/IMCEC.2018.8469232

[37] Xue, Y., Jiang, J., Ma, T., Liu, J., & Pang, W. (2018). A self-adaptive artificial bee colony algorithm with symmetry initialization. *Journal of Internet Technology, 19*(5), 1347-1362.

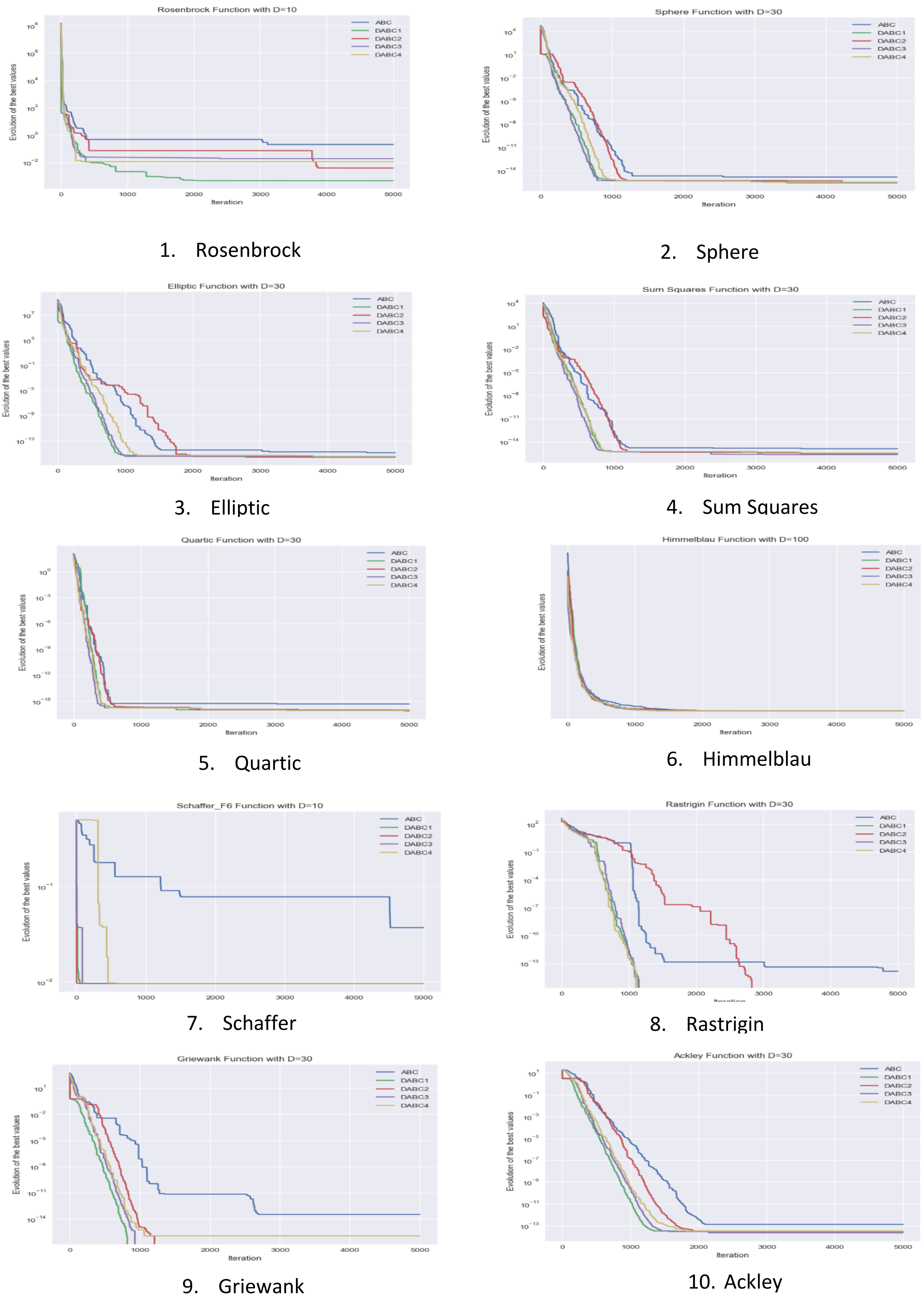
[38] Yao, B. Z., Yang, C. Y., Hu, J. J., Yin, G. D., & Yu, B. (2010). *An improved artificial bee colony algorithm for job shop problem.* Paper presented at the Applied Mechanics and Materials. https://doi.org/10.4028/www.scientific.net/AMM.26-28.657

[39] Zhang, L.-P., Yang, Z.-X., He, Q.-Y., & Cai, D.-M. (2017). *Immune algorithm for minimal Steiner tree problems.* Paper presented at the 2017 International Conference on Advanced Mechatronic Systems (ICAMechS). https://doi.org/10.1109/ICAMechS.2017.8316560

[40] Zhang, Q., & Liu, L. (2019). Whale optimization algorithm based on lamarckian learning for global optimization problems. *IEEE Access, 7*, 36642-36666. https://doi.org/10.1109/ACCESS.2019.2905009

[41] Zhou, J.-L., Wang, J.-S., Zhang, Y.-X., Guo, Q.-S., Li, H., & Lu, Y.-X. (2020). *Particle Swarm Optimization Algorithm with Variety Inertia Weights to Solve Unequal Area Facility Layout Problem.* Paper presented at the 2020 Chinese Control And Decision Conference (CCDC). https://doi.org/10.1109/CCDC49329.2020.9163977

[42] Zhu, G., & Kwong, S. (2010). Gbest-guided artificial bee colony algorithm for numerical function optimization. *Applied Mathematics and Computation, 217*(7), 3166-3173. https://doi.org/10.1016/j.amc.2010.08.049



1. Rosenbrock

2. Sphere

3. Elliptic

4. Sum Squares

5. Quartic

6. Himmelblau

7. Schaffer

8. Rastrigin

9. Griewank

10. Ackley

Figure 2: Convergence Curves of the ABC with DABC1, DABC2, DABC3, and DABC4

Table 2: Test functions

NO	Function	type	Search Range	optimal value	Formulation
f1	Rosenbrock	UN	[-50,50]	0	$f_1(x) = \sum_{i=1}^{D-1} 100\,(x_{i+1} - x_i{}^2)^{\,2} + (x_i - 1)^2$
f2	sphere	US	[-100,100]	0	$f_2(x) = \sum_{i=1}^{D} x_i^2$
f3	Elliptic	UN	[-100,100]	0	$f_3(x) = \sum_{i=1}^{D} (10^6)^{(i-1)/(D-1)} x_i^2$
f4	Sum Squares	US	[-10,10]	0	$f_4(x) = \sum_{i=1}^{D} i\, x_i^2$
f5	Quartic	US	[-1.28,1.28]	0	$f_5(x) = \sum_{i=1}^{D} i\, x_i^4$
f6	Himmelblau	MS	[-5,5]	78.3323	$f_6(x) = \frac{1}{D}\sum_{i=1}^{D}(x_i^4 - 16\,x_i^2 + 5\,x_i)$
f7	Scaffer's F6	MN	[-100,100]	0	$f_7(x) = 0.5 + \frac{sin^2\left(\sqrt{\sum_{i=1}^{D} x_i^2}\right) - 0.5}{(\,1 + 0.001\,(\sqrt{\sum_{i=1}^{D} x_i^2}))^2}$
f8	Rastrigin	MS	[-5.12,5.12]	0	$f_8(x) = \sum_{i=1}^{D} (x_i^2 - 10\cos(2\pi x_i) + 10)$
f9	Griewank	MN	[-600,600]	0	$f_9(x) = \frac{1}{4000}\sum_{i=1}^{D} x_i^2 - \prod_{i=1}^{D} \cos\left(\frac{x_i}{\sqrt{i}}\right) + 1$
f10	Ackley	MN	[-32.768,32.768]	0	$f_{10}(x) = 20 + e - 20exp\left[-0.2\sqrt{\frac{1}{D}\sum_{i=1}^{D} x_i^2}\right] - exp\left[\frac{1}{D}\sum_{i=1}^{D}\cos(2\pi\, x_i)\right]$

Table 4: Mean and standard deviation values obtained by the basic ABC algorithm compared with DABC1-DABC4 through 30 runs on 10 test benchmark functions.

Function	D	ABC		DABC1		DABC2		DABC3		DABC4	
		Mean	SD	Mean	SD	Mean	SD	Mean	SD	Mean	SD
Rosenbrock	10	2.24×10^{-1}	1.65×10^{-1}	**1.56×10^{-3}**	1.75×10^{-3}	**8.12×10^{-2}**	9.40×10^{-2}	**2.05×10^{-1}**	3.12×10^{-1}	**4.05×10^{-2}**	2.45×10^{-2}
	30	2.26×10^{-1}	1.60×10^{-1}	**5.52×10^{-3}**	7.04×10^{-3}	**3.96×10^{-1}**	5.68×10^{-1}	5.13×10^{-1}	8.54×10^{-1}	3.06×10^{-1}	4.64×10^{-1}
sphere	30	9.90×10^{-16}	1.66×10^{-16}	**2.62×10^{-16}**	3.80×10^{-17}	**3.07×10^{-16}**	6.11×10^{-17}	**1.32×10^{-16}**	3.07×10^{-17}	**2.50×10^{-16}**	5.26×10^{-17}
	60	7.65×10^{-15}	3.58×10^{-15}	**1.08×10^{-15}**	1.15×10^{-16}	**8.91×10^{-16}**	1.04×10^{-16}	**4.57×10^{-16}**	4.86×10^{-17}	**1.23×10^{-15}**	1.44×10^{-16}
Elliptic	30	1.18×10^{-15}	3.46×10^{-16}	**2.41×10^{-16}**	4.15×10^{-17}	**2.63×10^{-16}**	3.92×10^{-17}	**1.41×10^{-16}**	2.22×10^{-17}	**2.40×10^{-16}**	4.72×10^{-17}
	60	3.31×10^{-14}	3.37×10^{-14}	**9.42×10^{-16}**	1.32×10^{-16}	**9.15×10^{-16}**	1.12×10^{-4}	**5.45×10^{-16}**	8.99×10^{-17}	**1.02×10^{-15}**	1.31×10^{-16}
Sum Squares	30	9.65×10^{-16}	1.47×10^{-16}	**2.77×10^{-16}**	3.96×10^{-17}	**3.15×10^{-16}**	4.74×10^{-17}	**1.37×10^{-16}**	1.92×10^{-17}	**2.72×10^{-16}**	3.60×10^{-17}
	60	5.79×10^{-15}	3.49×10^{-15}	**1.13×10^{-15}**	1.06×10^{-16}	**1.05×10^{-15}**	1.17×10^{-16}	**4.84×10^{-16}**	8.59×10^{-17}	**1.18×10^{-15}**	1.48×10^{-16}
Quartic	30	3.81×10^{-16}	9.86×10^{-17}	**8.15×10^{-17}**	1.24×10^{-17}	**8.62×10^{-17}**	1.46×10^{-17}	**4.17×10^{-17}**	1.41×10^{-17}	**8.04×10^{-17}**	1.27×10^{-17}
	60	4.33×10^{-15}	2.75×10^{-15}	**2.93×10^{-16}**	3.29×10^{-17}	**3.08×10^{-16}**	4.21×10^{-17}	**1.73×10^{-16}**	4.56×10^{-17}	**3.15×10^{-16}**	4.61×10^{-17}
Himmelblau	100	- 78.3323	9.07×10^{-12}	**- 78.3323**	3.48×10^{-14}	**-78.3323**	2.33×10^{-14}	**-78.3323**	2.80×10^{-14}	**-78.3323**	3.48×10^{-13}
	200	- 78.3318	1.10×10^{-3}	**-78.3040**	5.65×10^{-2}	**-78.2755**	2.46×10^{-3}	-78.3181	4.24×10^{-2}	**-78.2899**	3.01×10^{-10}
Schaffer's F6	10	5.59×10^{-2}	2.32×10^{-2}	**8.42×10^{-3}**	3.30×10^{-3}	**1.70×10^{-2}**	1.21×10^{-2}	**9.71×10^{-3}**	1.54×10^{-10}	**9.71×10^{-3}**	2.17×10^{-9}
	30	4.91×10^{-1}	4.48×10^{-3}	**9.71×10^{-3}**	1.50×10^{-16}	**2.06×10^{-1}**	4.62×10^{-2}	**5.49×10^{-2}**	2.30×10^{-2}	**9.71×10^{-3}**	2.35×10^{-10}
Rastrigin	30	4.61×10^{-14}	3.92×10^{-14}	**0**	0	**0**	0	**0**	0	**2.42×10^{-15}**	1.02×10^{-14}
	60	4.14×10^{-7}	2.03×10^{-6}	**0**	0	**1.65×10^{-12}**	8.54×10^{-12}	**0**	0	**5.39×10^{-11}**	2.69×10^{-10}
Griewank	30	2.10×10^{-13}	5.21×10^{-13}	**1.48×10^{-17}**	3.77×10^{-17}	**0**	0	**7.40×10^{-18}**	2.76×10^{-17}	4.17×10^{-8}	1.48×10^{-7}
	60	1.36×10^{-12}	4.88×10^{-12}	**5.18×10^{-17}**	5.53×10^{-17}	**5.46×10^{-13}**	8.06×10^{-13}	**8.88×10^{-17}**	2.01×10^{-16}	4.43×10^{-6}	2.30×10^{-5}
Ackley	30	1.24×10^{-13}	2.93×10^{-14}	**2.86×10^{-14}**	3.16×10^{-15}	**2.87×10^{-14}**	3.36×10^{-15}	**1.35×10^{-14}**	3.45×10^{-15}	**3.36×10^{-14}**	3.34×10^{-15}
	60	1.53×10^{-12}	9.32×10^{-13}	**6.86×10^{-14}**	3.82×10^{-15}	**6.96×10^{-14}**	5.40×10^{-15}	**7.13×10^{-14}**	8.86×10^{-15}	**7.96×10^{-14}**	5.03×10^{-15}