Neural Computing and Applications (2020) 32:9521–9543
https://doi.org/10.1007/s00521-019-04465-6

ORIGINAL ARTICLE

Hybrid sine cosine artificial bee colony algorithm for global optimization and image segmentation

Shubham Gupta[1] · Kusum Deep[1]

Received: 27 October 2018 / Accepted: 26 August 2019 / Published online: 23 September 2019

Abstract
Artificial bee colony (ABC) algorithm is an efficient biological-inspired optimization method, which mimics the foraging behavior of honey bees to solve the complex and nonlinear optimization problems. However, in some cases, it suffers from inefficient exploration, low exploitation and slow convergence rate. These shortcomings cause the problem of stagnation at local optimum which is dangerous in determining the true solution (optima) of the problem. Therefore, in the present paper, an attempt has been made toward the removal of the drawbacks from the classical ABC by proposing a novel hybrid method called SCABC algorithm. The SCABC algorithm hybridizes the ABC with sine cosine algorithm (SCA) to upgrade the level of exploitation and exploration in the classical ABC algorithm. The SCA is a recently introduced algorithm, which uses the trigonometric functions sine and cosine to perform the search. The validation of the SCABC algorithm is performed on a well-known benchmark set of 23 optimization problems. The various analysis metrics such as statistical, convergence and performance index analysis verify the better search ability of the SCABC as compared to classical ABC, SCA. The comparison with some other optimization algorithms demonstrates a comparatively better state of exploitation and exploration in the SCABC algorithm. Moreover, the SCABC is also employed on multilevel thresholding problems. The various performance measures demonstrate the efficacy of the SCABC algorithm in determining the optimal thresholds of gray images.

Keywords Optimization · Artificial bee colony (ABC) algorithm · Sine cosine algorithm (SCA) · Hybrid algorithms · Multilevel thresholding

1 Introduction

From some past years, the swarm intelligence-based algorithms have gained enormous attention to solve the complex optimization problems because of their simple structure, easy implementation, flexibility and local optima avoidance ability. The swarm intelligence is popular field of nature-inspired algorithms where the information exchange via collaborative communication between the search agents is utilized. The information-exchange mechanism between the search agents helps in exploring the promising areas of the search space, and the communication between the agents is required to avoid local optima during the search. In the literature, numerous algorithms are available which belong to the category of swarm intelligence. The reason behind the existence of large number of algorithms can be answered with the "no free lunch theorem (NFL)" [1] which was the revolutionary development in the field of nature-inspired algorithms. The NFL opposes the existence of an ideal optimization algorithm which can solve all the optimization problems. In other words, it can be said that a particular algorithm may be the best on a particular set of problems, but for some other set of optimization problems, this particular algorithm may perform poorly. This theorem always allows the researchers to develop new algorithms or improve the existing algorithms. The swarm intelligence-based algorithms are widely used because they utilize less number of parameters. Some well-known and wide applicable algorithms based on swarm intelligence are particle swarm optimization (PSO) [2], ant colony optimization (ACO) [3],

✉ Shubham Gupta
sgupta@ma.iitr.ac.in

Kusum Deep
Kusumfma@iitr.ac.in

1 Department of Mathematics, Indian Institute of Technology Roorkee, Roorkee, Uttarakhand 247667, India

artificial bee colony (ABC) algorithm [4], cuckoo search (CS) [5], firefly algorithm (FA) [6], etc.

The artificial bee colony (ABC) algorithm was developed by Karaboga and Basturk [4] by mimicking the social and foraging behavior of honey bees. Although the classical ABC performs well on several real-life problems, in some cases, it suffers from inefficient exploration, low exploitation and slow convergence rate. Therefore, in the literature, several attempts have been performed to tackle these issues. For example, in [7], three modified search equations are introduced to improve the search ability of bees in terms of exploration and exploitation. Zhu and Kwong [8] have proposed a gbest-guided ABC which was inspired by the PSO. Liu et al. [9] have proposed a new search mechanism based on the information of global best and previous best solutions. In this search mechanism, two new scaling factors are introduced to maintain the balance between exploitation and exploration. In [10], the diversity of the bee colony is increased and the exploitation ability of bees is increased by modifying the search equations. To overcome issues in balancing, searching and convergence, several modified search equations are proposed [11, 12]. In [13], a multistrategy ensemble ABC algorithm is proposed which tries to establish an appropriate balance between exploitation and exploration. In order to maintain the diversity in the algorithm and to enhance the exploration ability of bees, multipopulation strategy is adopted [14]. In [15], to balance between diversity and convergence in the ABC, Levy-flight-inspired search strategy is integrated in ABC. In [16], a modified ABC is introduced based on neighborhood search strategy to enhance the exploration ability of bees and to maintain the diversity in the algorithm. In order to boost up the exploration ability of bees, ABC is hybridized with Grey wolf optimizer algorithm [17]. In [18], a Cauchy operator is employed to balance between global and local search strategies. In [19], the ABC algorithm is hybridized with Bat algorithm to enhance the exploration ability and convergence rate in the ABC. The literature presented here on ABC is not covering all the aspects, and therefore, the other developments can be found in [20].

Although, in the literature, various attempts have been done to improve the efficiency of ABC, in many cases, it has been realized experimentally that similar to other meta-heuristic algorithms, ABC also suffers from the imbalance between the search operators exploration and exploitation. Therefore, there is a chance of improving the search ability of the classical ABC algorithm. Also, the NFL theorem allows improving the search efficiency of the algorithm so that it can be employed to solve complex real-life problems with comparatively better accuracy. One way to improve the performance of ABC algorithm can be the hybridization with other algorithms. In the literature, various algorithms are hybridized to propose a comparative better optimization algorithm [21–26]. In these hybrid algorithms, the advantages (better exploration or better exploitation) of various algorithms are integrated. By inspiring these concepts of hybridization, an attempt has been made in the present paper toward the enhancement of search efficiency of employed bees in ABC algorithm through sine cosine algorithm (SCA) [27]. The proposed hybrid method is named as sine cosine artificial bee colony (SCABC). In the SCABC, the modified SCA search equations are integrated for the employed bee phase to level up the exploration and exploitation ability of search agents. The modification in the classical search mechanism of SCA is done to sustain the balance of exploration in SCABC because the classical search equations of SCA show the imbalance between exploration and exploitation [28]. The investigation of the proposed SCABC is performed on a well-known set of 23 benchmark problems. The numerical results and comparison analysis through various performance metrics verify the efficacy of the proposed SCABC algorithm. Moreover, in the paper, the SCABC also employed to determine the optimal thresholds for the multilevel thresholding problem. The comparison of results on this problem ensures that the SCABC is a comparatively better optimization method in finding the optimal thresholds for gray images.

The rest of the paper is arranged as follows: Sect. 2 provides an overview and conceptual description of classical ABC and classical SCA. In Sect. 3, the proposed hybrid method called SCABC is explained in detail. Section 4 provides the numerical experimentation and validation of the SCABC on 23 classical benchmarks. In Sect. 5, the SCABC is employed on multilevel thresholding problem. Finally, the conclusion of the work is presented in Sect. 6 with some future research directions.

2 Overview of classical ABC and SCA

In this section, the overview of classical ABC and SCA has been presented with conceptual details and working mechanism.

2.1 Artificial bee colony algorithm

The ABC algorithm was designed by Karaboga [4] for global optimization by inspiring the collaborative foraging behavior of honey bees. In ABC, the determination of optimal solution can be regarded as the equivalent process of foraging behavior by bees. Moreover, in the ABC, the food sources can be regarded as the possible solutions to the optimization problem and the amount of nectar in food source denotes the fitness of objective function. In ABC algorithm, the food sources are evolved in three phases—

(1) employed bee (2) onlooker bee, and (3) scout bee. In the algorithm, number of food sources, employed bees and onlooker bees are considered to be equal.

In employed bee phase, each employed bee is associated with a unique food source and tries to locate a new position based on the information of associated food source and random food source using the following search equation:

$$u_{i,j} = x_{i,j} + \eta_{i,j} \cdot \left(x_{i,j} - x_{k,j}\right) \tag{1}$$

where j represents the randomly selected dimension coordinate from the set $\{1, 2, \ldots D\}$, $\eta_{i,j}$ is uniformly distributed random number in the interval $(-1,\ 1)$. k represents the random food source different from other than i, $\vec{u}_i$ is a newly evolved food source by employed bee $\vec{x}_i$. If the newly generated solution $\vec{u}_i$ is found better in terms of fitness value, then the old solution $\vec{x}_i$ is replaced by the solution $\vec{u}_i$.

After the completion of employed bee phase, the onlooker bee phase starts. In this phase, the location and nectar amount of food source are shared with onlooker bees. The onlooker bees search for better locations of food sources based on the probability p_i, which can be calculated as follows:

$$p_i = \frac{\text{fitness}_i}{\sum_{i=1}^{N_f} \text{fitness}_i} \tag{2}$$

where fitness_i represents the fitness of the ith food source or nectar amount available at ith food source and N_f denotes the number of food sources. Based on these probabilities, onlooker bees locally exploit the neighborhood areas of food sources with the help of Eq. (1). Similar to the employed bee phase, a greedy selection is applied here between the newly generated food source and its old location.

When the phase of onlooker bee is completed, then the scout bees are used to remove such food sources which are not evolved up to some predefined limit. In the scout bee phase, the undeveloped food sources up to the predefined threshold limit are replaced with the uniform distributed food sources using the following equation:

$$u_{i,j} = \text{lb}_j + \text{rand} \cdot \left(\text{ub}_j - \text{lb}_j\right) \tag{3}$$

where lb_j and ub_j denote the available lower and upper bounds for the dimension j of the ith food source.

The steps of ABC are presented in the form of pseudo-code in Algorithm 1. Some key points in the pseudo-code are: (1) the termination criteria for ABC algorithm are fixed as maximum number of function evaluations (FESs); (2) when the bee is evaluated using objective function, the FES is increased with a number 1; and (3) the limit count parameter (l) of the ABC is increased by 1 when the newly generated food source is not better than the old one.

2.2 Sine cosine algorithm

The sine cosine algorithm [27] uses the features of trigonometric functions sine and cosine in its search strategy. This algorithm was introduced by Mirjalili in 2014 [27] for global optimization problems. Similar to other optimization algorithms, the SCA also searches the solution randomly with the guidance of global best solution. In the SCA, the global best solution is known as destination point. The search equations, which are used in SCA to evolve the position of candidate solutions, are defined as follows:

$$\vec{x}_{i,t+1} = \vec{x}_{i,t} + \alpha \times \sin(\psi) \times \left|c \cdot \vec{x}_{D,t} - \vec{x}_{i,t}\right| \tag{4}$$

$$\vec{x}_{i,t+1} = \vec{x}_{i,t} + \alpha \times \cos(\psi) \times \left|c \cdot \vec{x}_{D,t} - \vec{x}_{i,t}\right| \tag{5}$$

where $\vec{x}_{i,t+1}$ and $\vec{x}_{i,t}$ denote the location of candidate solution $\vec{x}_i$ at the iterations $t+1$ and t, respectively. $\vec{x}_{D,t}$ represents the position of destination point obtained so far. α, ψ and c are random numbers that maintain the randomness in search process to prevent from the situation of getting trapped at local optimum. Equations (4) and (5) can be written jointly as follows:

$$\vec{x}_{i,t+1} = \vec{x}_{i,t} + \begin{cases} \alpha \times \sin(\psi) \times \left|c \cdot \vec{x}_{D,t} - \vec{x}_{i,t}\right| & \text{if rand} < 0.5 \\ \alpha \times \cos(\psi) \times \left|c \cdot \vec{x}_{D,t} - \vec{x}_{i,t}\right| & \text{otherwise} \end{cases} \tag{6}$$

where rand represents the uniformly distributed random number from the interval (0, 1). This random number helps to transit from cosine to sine function and vice versa.

In the search mechanism of the SCA, the random number α is responsible for the exploration and exploitation of search space. The value of α which is adopted in the SCA is formulated as follows:

$$\alpha = 2 - 2 \times \left(\frac{t}{T}\right) \tag{7}$$

where t represents the current iteration and T is the maximum number of iterations. From Eq. (7), it can be seen that random number α linearly decreases from the value 2 to 0. The value of α which lies in the interval $(1, 2)$ supports to the exploration, and in this situation, new promising areas of the search space are discovered. When the value of α lies in the interval $(0,\ 1)$, the regions of the search space which are already discovered are exploited. Thus, the parameter α helps to transit from the exploration phase to the exploitation phase. This transition maintains the balance between exploitation and exploration. The random number ψ decides the movement of the current solution $\vec{x}_i$ either toward the destination point or outwards the destination point. The random number c is introduced to support the exploration of search space when the parameter α fails. The parameter c randomly explores and exploits the search space within the algorithm. The framework of classical SCA is provides in Algorithm 1.

Algorithm 1. Sine Cosine Algorithm

```
1.  Initialize the population of candidate solutions using uniform distribution
2.  Evaluate the fitness of each candidate solution
3.  Select destination point $\vec{x}_D$
4.  Initialize the algorithm parameters $\alpha, \psi$ and $c$
5.  Initialize the iteration count $t = 0$
6.  while $t < T$
7.      for each candidate solution
8.          Update the solution using equations (6) and (7)
9.      end of for
10.     Evaluate the fitness of updated candidate solutions
11.     Update the destination point $\vec{x}_D$
12.     Update the algorithm parameter $\alpha$
13. $t = t + 1$
14. end of while
```

3 Proposed hybrid method: SCABC

This section introduces the proposed hybrid method called SCABC. First, the motivations of hybridizing SCA and ABC are provided in detail. Second, the search strategy of SCABC has been discussed with pseudo-code.

3.1 Motivations

In the ABC algorithm, the employed bees are associated with food sources by one-to-one correspondence. The information of nectar amount available at food source is passed to the onlooker bees by employed bees so that bees can concentrate on more promising food sources. Thus, the employed bee phase plays an important role during the search process to explore more promising regions. The employed bee searches new food source randomly without any guidance which may decrease the convergence rate. Here, the slow convergence occurs due to the worthless and irregular exploration produced by search equation of classical ABC algorithm. Therefore, an intelligent explorative behavior with proper balance between the operators' exploration and exploitation can be prepared for the employed bee phase.

In the literature, it has been found that the SCA is rich in exploration and tries to move from the exploration phase to the exploitation over the iterations. But, sometimes the algorithm faces the problem of high diversity at early stages of the algorithm, which degrade the performance of the algorithm, and skips the true solutions during the search. The low exploitation at early iterations and low exploration at the later iterations of algorithm may cause the problem of slow convergence and premature convergence, respectively. Therefore, an appropriate balance between exploration and exploitation can be maintained with the help of some memory-based information. The shortcoming of high diversity in SCA can be alleviated by integrating the elite guidance, and this guidance may be useful for employed bees to prevent from irregular exploration. Therefore, by inspiring from all the above advantages and to improve the efficiency of both the algorithms, the present work hybridizes the ABC algorithm with SCA.

3.2 The conceptual description of proposed SCABC algorithm

The proposed SCABC algorithm can be considered as an extended ABC algorithm in which the employed bee phase is improved by the modified search equations of SCA. The modifications in the search mechanism of classical SCA are done to provide an elite guidance for the candidate solutions. The integrated elite guidance tries to detract the problem of high diversity from the classical search equation of SCA. The modified search equation of SCA is as follows:

$$\vec{x}_{i,t+1} = \vec{x}_{D,t} + \begin{cases} \left|\dfrac{f_{\text{best}}}{f_{\text{worst}}}\right| \times \sin(\psi) \times \left|c \cdot \vec{x}_{D,t} - \vec{x}_{i,t}\right| & \text{if rand} < 0.5 \\ \left|\dfrac{f_{\text{best}}}{f_{\text{worst}}}\right| \times \cos(\psi) \times \left|c \cdot \vec{x}_{D,t} - \vec{x}_{i,t}\right| & \text{otherwise} \end{cases} \tag{8}$$

where the coefficient c and the parameters ψ, rand are same as defined in Sect. 2.2. f_{best} represents the objective function value at destination point $\vec{x}_{D,t}$, and f_{worst} represents the worst food source in terms of objective function value. Whenever $f_{\text{worst}} = 0$, then it is replaced by sufficiently large real number. From the search equation, it is clear that the ratio $\left|\frac{f_{\text{best}}}{f_{\text{worst}}}\right|$ generates a real number between 0 and 1. This ratio controls the amplification of the difference vector $\left|c \cdot \vec{x}_{D,t} - \vec{x}_{i,t}\right|$ and maintains the exploration and exploitation within the algorithm. The steps involved in the SCABC are presented in Algorithm 2. The flowchart of the algorithm is shown in Fig. 1.

Algorithm 2. Hybrid Sine Cosine Artificial Bee Colony Algorithm (SCABC)

```
1.  Initialize the $N_f$ number of food sources randomly using uniform distribution
2.  Evaluate the fitness of each food source and set $NFS = N_f$
3.  Initialize the parameters $l_{\max}$, number of employed bees (#EB) and
    number of onlooker bees (#OB)
4.  while $FES \le FES_{max}$
5.    %%%%%    employed ee phase    %% %%%
6.        for $i = 1$ to #EB
7.            Generate a food source $\vec{u}_i$ using equation (8) corresponding
              to the food source $\vec{x}_i$
8.            if $f(\vec{u}_i) < f(\vec{x}_i)$
9.                Replace the food source $\vec{x}_i$ with $\vec{u}_i$
10.           else
11.               $l(i) = l(i) + 1$
12.           end of if
13.           $FES = FES + 1$
14.           Memorize the best solution $\vec{x}_{best}$ obtained so far
15.       end of for
16.   Calculate the probability of all the food sources using eq. (2)
17.   %%%%%    onlooker bee phase    %% %%%
18.       for $m = 1$ to #OB
19.           Select the index i using roulette wheel selection and probability $p_i$
20.           Generate a food source $\vec{u}_i$ using equation (1) corresponding
              to the food source $\vec{x}_i$
21.           if $f(\vec{u}_i) < f(\vec{x}_i)$
22.               Replace the food source $\vec{x}_i$ with $\vec{u}_i$
23.           else
24.               $l(i) = l(i) + 1$
25.           end of if
26.           $FES = FES + 1$
27.           Memorize the best solution $\vec{x}_{best}$ obtained so far
28.       end of for
29.   %%%%%    scout bee phase    %% %%%
30.       for $i = 1$ to $N_f$
31.           if $l(i) > l_{\max}$
32.               generate a uniformly distributed random food source $\vec{z}_i$
                  using eq. (3)
33.           end of if
34.           $\vec{x}_i = \vec{z}_i$
35.           $f(\vec{x}_i) = f(\vec{u}_i)$
36.           memorize the best solution $\vec{x}_{best}$ obtained so far
37.       end of for
38. end of while
39. return the best solution $\vec{x}_{best}$
```

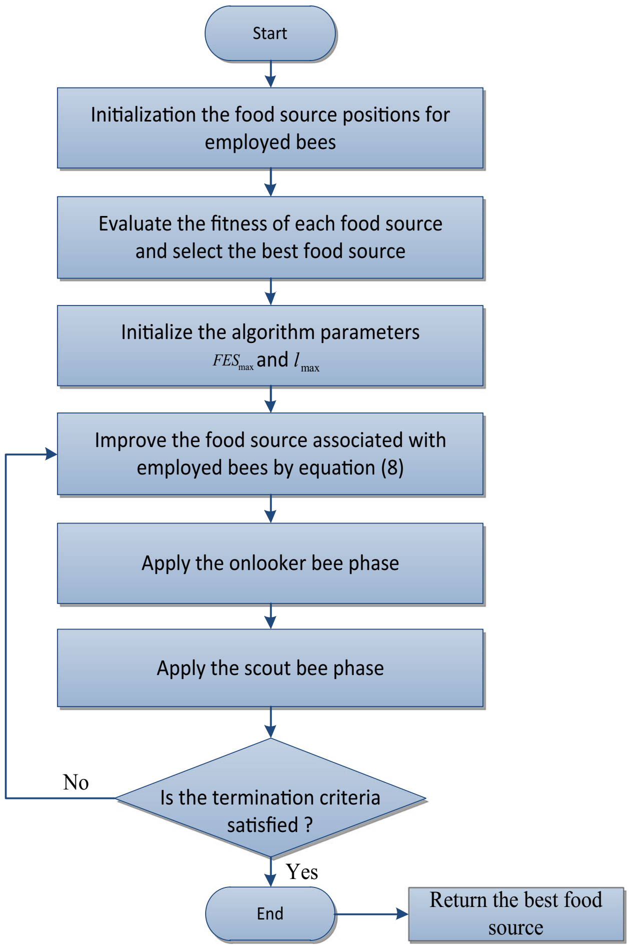

Start
Initialization the food source positions for employed bees
Evaluate the fitness of each food source and select the best food source
Initialize the algorithm parameters $FES_{\max}$ and $I_{\max}$
Improve the food source associated with employed bees by equation (8)
Apply the onlooker bee phase
Apply the scout bee phase
Is the termination criteria satisfied ?
No
Yes
End
Return the best food source

Fig. 1 Flowchart of proposed SCABC algorithm

4 Experimental validation of proposed SCABC algorithm

In this section, the proposed hybrid method called SCABC is evaluated on a classical benchmark set. This test set contains 23 well-known benchmark problems which are collected from various sources, and many researchers have been used them to evaluate their algorithms [29–32]. The description of these problems is presented in Table 1. These problems are of minimization type. In this set, first 10 problems are unimodal while remaining are multimodal problems. In the experiments, the number of search agents is fixed to 50 and maximum function evaluations (the termination criteria) are fixed to 2×10^5.

4.1 Numerical results and analysis

In this section, the numerical results obtained by the classical ABC, classical SCA and the proposed SCABC algorithms are recorded and presented in Tables 2 and 3 corresponding to 10- and 30-dimensional problems, respectively. In these tables, the best, average, median, worst and standard deviation value of the objective function values recorded over 30 independent trails. Since the problems are of minimization type, therefore, between two different objective function values of a particular problem, the one having less value is better. These better values are highlighted with boldface in Tables 2 and 3. The data of results obtained by the SCABC, ABC and SCA algorithms in 30 independent trials are presented in the form of boxplots in Fig. 2. The boxplots clearly demonstrate that the results obtained by the SCABC are satisfactory as compared to the SCA and ABC algorithms. This is due to the reason that 25th and 75th percentiles of samples collected for proposed SCABC algorithm decline toward the minimum solution within a narrow interquartile range.

4.1.1 Comparison of results in terms of exploitation strength

In the benchmark set, the problems from $F1$ to $F10$ are unimodal, and in these problems only one minima is present which is known as global minima. Therefore, these problems are generally used to evaluate the exploitation strength and convergence rate of algorithms. On the test problems $F1$–$F5$, $F7$, $F8$ and $F10$, the proposed SCABC is better than the classical ABC and classical SCA in all the statistics such as best, median, average, maximum and standard deviation values of objective function corresponding to the 10- and 30-dimensional problems. In 10-dimensional $F6$, the SCABC provides better median and best value of the objective function as compared to the other algorithms. Average and worst value is better in ABC, and the standard deviation value is better in the SCA as compared to the other algorithms. In 30-dimensional $F6$, the best value is better in ABC, median value is better in SCABC and other statistics are better in the SCA as compared to the other algorithms. For the 10-dimensional $F9$, the classical SCA is better than the classical ABC and the proposed SCABC algorithms in terms of all the statistics. In 30-dimensional $F9$, the SCABC algorithm is better than the classical ABC and classical SCA in terms of average, worst and standard deviation values of the objective function. In terms of best and median value of the objective function, the classical SCA is better than other algorithms for 30-dimensional $F9$. Hence, from the comparison of results on unimodal problems demonstrates the better search ability of the SCABC algorithm in terms of exploitation strength and convergence rate as compared to the ABC and SCA.

Table 1 Description of the classical benchmark problems

Test problem	Range of search space	$F_{\min}$
$F1(x) = \sum_{i=1}^{d} x_i^2$	$[-100, 100]$	0
$F2(x) = \sum_{i=1}^{d} i x_i^2$	$[-10, 10]$	0
$F3(x) = \sum_{i=1}^{d} \|x_i\| + \prod_{i=1}^{d} \|x_i\|$	$[-10, 10]$	0
$F4(x) = \sum_{i=1}^{d} \left(\sum_{j-1}^{i} x_i \right)^2$	$[-100, 100]$	0
$F5(x) = \max_i \{\|x_i\|, 1 \le i \le d\}$	$[-100, 100]$	0
$F6(x) = \sum_{i=1}^{d-1} \left[100\left(x_{i+1} - x_i^2\right)^2 + (x_i - 1)^2 \right]$	$[-30, 30]$	0
$F7(x) = \sum_{i=1}^{d} ([x_i + 0.5])^2$	$[-100, 100]$	0
$F8(x) = \sum_{i=1}^{d} i \cdot x_i^4$	$[-1.28, 1.28]$	0
$F9(x) = \sum_{i=1}^{d} i \cdot x_i^4 + \mathrm{rand}[0, 1)$	$[-1.28, 1.28]$	0
$F10(x) = \sum_{i=1}^{d} \|x_i\|^{i+1}$	$[-1, 1]$	0
$F11(x) = \sum_{i=1}^{d} -x_i \sin\left(\sqrt{\|x_i\|}\right)$	$[-500, 500]$	$-418.9829 \times d$
$F12(x) = \sum_{i=1}^{d} \left[x_i^2 - 10\cos(2\pi x_i) + 10 \right]$	$[-5.12, 5.12]$	0
$F13(x) = \sum_{i=1}^{d} -20 \exp\left(-0.2\sqrt{\frac{1}{d}\sum_{i=1}^{d} x_i^2} \right) - \exp\left(\frac{1}{d}\sum_{i=1}^{d} \cos(2\pi x_i) \right) + 20 + e$	$[-32, 32]$	0
$F14(x) = \frac{1}{4} \times 10^{-3} \sum_{i=1}^{d} x_i^2 - \prod_{i=1}^{d} \cos\left(x_i/\sqrt{i}\right) + 1)$	$[-600, 600]$	0
$F15(x) = \frac{\pi}{d}\left\{ 10\sin(\pi y_1) + \sum_{i=1}^{d-1} (y_i - 1)^2 \left[1 + 10\sin^2(\pi y_{i+1}) \right] + (y_n - 1)^2 \right\} + \sum_{i=1}^{d} u(x_i, 10, 100, 4)$; $y_i = \frac{x_i + 5}{4}$; $u(x_i, a, k, m) = \begin{cases} k(x_i - a)^m & \text{if } x_i > a \\ k(-x_i - a)^m & \text{if } x_i < -a \\ 0 & \text{otherwise} \end{cases}$	$[-50, 50]$	0
$F16(x) = 0.1 \times \left\{ \sin^2(3\pi x_1) + \sum_{i=1}^{d} (x_i - 1)^2 \left[1 + \sin^2(1 + 3\pi x_i) \right] + (x_d - 1)^2 \left[1 + \sin^2(2\pi x_n) \right] \right\} + \sum_{i=1}^{d} u(x_i, 5, 100, 4)$	$[-50, 50]$	0
$F17(x) = \sum_{i=1}^{d} \|x_i \sin(x_i) + 0.1 x_i\|$	$[-10, 10]$	0
$F18(x) = 0.1d - \left(0.1 \sum_{i=1}^{d} \cos(5\pi x_i) - \sum_{i=1}^{d} x_i^2 \right)$	$[-1, 1]$	0
$F19(x) = \sum_{i=1}^{d} x_i^2 + \left(\sum_{i=1}^{d} 0.5 i x_i \right)^2 + \left(\sum_{i=1}^{d} 0.5 i x_i \right)^4$	$[-5, 10]$	0
$F20(x) = \sum_{i=1}^{d} (10^6)^{(i-1)/(d-1)} x_i^2$	$[-100, 100]$	0

Table 1 (continued)

Test problem	Range of search space	$F_{\min}$
$F21(x) = (-1)^{d+1}\prod_{i=1}^{d}\cos(x_i) \times e^{\left[-\sum_{i=1}^{d}(x_i-\pi)^2\right]}$	$[-100, 100]$	0
$F22(x) = 1 - \cos(2\pi\left(\sqrt{\left(\sum_{i=1}^{d}x_i^2\right)}\right) + 0.1 \times \sqrt{\sum_{i=1}^{d}x_i^2}$	$[-100, 100]$	0
$F23(x) = 0.5 + \frac{\sin^2\left(\sqrt{\sum_{i=1}^{d}x_i^2}\right)-0.5}{\left(1+0.001\left(\sum_{i=1}^{d}x_i^2\right)\right)^2}$	$[-100, 100]$	0

Table 2 Comparison of results obtained by ABC, SCA and proposed SCABC algorithms on classical benchmark problems with dimension 10

Test function	Algorithm	Best	Median	Average	Worst	STD
*F*1	ABC	9.44E−09	6.64E−08	1.03E−07	3.46E−07	9.32E−08
	SCA	3.06E−149	5.33E−141	2.56E−132	7.65E−131	1.40E−131
	SCABC	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**
*F*2	ABC	8.16E−10	5.07E−09	7.93E−09	4.14E−08	8.46E−09
	SCA	3.88E−150	1.45E−140	6.23E−133	1.74E−131	3.17E−132
	SCABC	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**
*F*3	ABC	2.20E−05	6.50E−05	7.24E−05	1.45E−04	3.35E−05
	SCA	1.02E−97	2.11E−89	6.59E−82	1.98E−80	3.61E−81
	SCABC	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**
*F*4	ABC	1.94E+02	4.40E+02	4.45E+02	7.01E+02	1.45E+02
	SCA	3.46E−80	1.20E−67	1.11E−57	3.27E−56	5.97E−57
	SCABC	**1.71E−194**	**3.02E−185**	**8.42E−176**	**2.49E−174**	**0.00E+00**
*F*5	ABC	4.32E+00	6.83E+00	6.69E+00	1.08E+01	1.64E+00
	SCA	2.52E−49	8.21E−47	4.19E−39	1.26E−37	2.29E−38
	SCABC	**4.30E−142**	**5.44E−135**	**2.32E−132**	**5.05E−131**	**9.37E−132**
*F*6	ABC	1.01E+00	4.40E+00	**4.38E+00**	**7.71E+00**	1.92E+00
	SCA	6.10E+00	6.48E+00	6.60E+00	7.24E+00	**3.86E−01**
	SCABC	**4.77E−02**	**1.04E−01**	1.43E+01	1.16E+02	3.46E+01
*F*7	ABC	1.15E−08	1.16E−07	1.49E−07	6.15E−07	1.28E−07
	SCA	8.02E−02	1.47E−01	1.62E−01	3.47E−01	7.33E−02
	SCABC	**6.07E−31**	**1.66E−26**	**1.14E−23**	**2.14E−22**	**4.11E−23**
*F*8	ABC	2.55E−24	8.37E−22	3.55E−21	2.57E−20	6.90E−21
	SCA	1.83E−254	3.08E−234	1.01E−210	3.02E−209	**0.00E+00**
	SCABC	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**
*F*9	ABC	8.12E−03	1.39E−02	1.55E−02	3.22E−02	5.37E−03
	SCA	**2.41E−05**	**1.18E−04**	**2.02E−04**	**6.12E−04**	**1.82E−04**
	SCABC	2.08E−03	3.36E−03	3.40E−03	5.10E−03	7.43E−04
*F*10	ABC	3.67E−19	2.54E−16	6.93E−16	5.84E−15	1.15E−15
	SCA	3.44E−282	2.84E−270	2.45E−257	7.36E−256	**0.00E+00**
	SCABC	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**
*F*11	ABC	− 4.19E+03	− 4.18E+03	− 4.17E+03	− 4.07E+03	2.53E+01
	SCA	− 2.78E+03	− 2.50E+03	− 2.48E+03	− 2.18E+03	1.32E+02
	SCABC	**− 4.19E+03**	**− 4.19E+03**	**− 4.19E+03**	**− 4.19E+03**	**1.08E−01**
*F*12	ABC	1.78E−03	9.41E−02	2.05E−01	1.01E+00	2.71E−01
	SCA	**0.00E+00**	**0.00E+00**	**7.55E−04**	**2.27E−02**	**4.14E−03**
	SCABC	**0.00E+00**	4.48E+00	4.34E+00	9.95E+00	1.74E+00

Table 2 (continued)

Test function	Algorithm	Best	Median	Average	Worst	STD
*F*13	ABC	2.08E−03	6.00E−03	6.62E−03	1.53E−02	3.22E−03
	SCA	**8.88E−16**	4.44E−15	3.14E−15	4.44E−15	1.74E−15
	SCABC	**8.88E−16**	**8.88E−16**	**8.88E−16**	**8.88E−16**	**0.00E+00**
*F*14	ABC	2.68E−04	7.15E−03	7.62E−03	2.26E−02	5.66E−03
	SCA	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**
	SCABC	**0.00E+00**	1.23E−07	1.42E−07	5.46E−07	1.19E−07
*F*15	ABC	1.53E−09	3.85E−08	**5.93E−08**	**1.92E−07**	**5.23E−08**
	SCA	8.65E−03	4.22E−02	3.81E−02	6.81E−02	1.69E−02
	SCABC	**1.77E−31**	**1.16E−25**	5.62E−01	1.06E+01	1.98E+00
*F*16	ABC	2.41E−08	2.00E−07	**2.40E−07**	**7.30E−07**	**1.80E−07**
	SCA	6.95E−02	1.39E−01	1.49E−01	2.86E−01	6.87E−02
	SCABC	**1.48E−29**	**1.31E−25**	2.33E−02	1.96E−01	4.94E−02
*F*17	ABC	2.59E−04	7.07E−04	7.07E−04	1.30E−03	2.55E−04
	SCA	**1.10E−89**	**1.27E−83**	**4.25E−59**	**1.27E−57**	**2.33E−58**
	SCABC	1.05E−15	5.44E−15	5.59E−15	1.07E−14	2.31E−15
*F*18	ABC	9.21E−11	8.71E−10	1.17E−09	5.51E−09	1.28E−09
	SCA	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**
	SCABC	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**
*F*19	ABC	4.90E+00	1.18E+01	1.21E+01	2.30E+01	4.50E+00
	SCA	2.62E−90	3.13E−84	1.01E−77	3.03E−76	5.54E−77
	SCABC	**3.28E−252**	**7.49E−248**	**2.00E−245**	**5.83E−244**	**0.00E+00**
*F*20	ABC	1.42E−04	2.27E−03	3.83E−03	1.64E−02	3.96E−03
	SCA	4.57E−146	1.19E−136	9.16E−129	1.47E−127	3.38E−128
	SCABC	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**
*F*21	ABC	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**
	SCA	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**
	SCABC	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**
*F*22	ABC	5.00E−01	7.06E−01	7.34E−01	1.00E+00	1.16E−01
	SCA	**9.99E−02**	**9.99E−02**	9.99E−02	**9.99E−02**	4.05E−09
	SCABC	**9.99E−02**	**9.99E−02**	**9.99E−02**	**9.99E−02**	**4.49E−17**
*F*23	ABC	7.82E−02	2.28E−01	2.09E−01	3.12E−01	4.64E−02
	SCA	**9.72E−03**	**9.72E−03**	**9.72E−03**	**9.72E−03**	**2.50E−09**
	SCABC	**9.72E−03**	**9.72E−03**	1.06E−02	3.72E−02	5.02E−03

4.1.2 Comparison of results in terms of exploration strength

The test problems from *F*11 to *F*23 in the benchmark set are multimodal problems where several local optima are present. In the test problems *F*11, *F*13, *F*19 and *F*20, the proposed SCABC algorithm provides better results in terms of all the statistics as compared to the classical ABC and classical SCA for 10- and 30-dimensional problems. In the problem *F*18, the SCABC and SCA provide the optima of the problem and are better than the classical ABC algorithm for both 10 and 30 dimensions. In the problems *F*15, *F*16, *F*22 and *F*23, the proposed SCABC algorithm is either performed better or very competitive than the ABC and SCA. In the problem *F*12, the SCA and ABC both are better than the SCABC algorithm. The objective function in the problem *F*12 is also known as Rastrigin function. This function has large number of local optima, and therefore, the high diversity is needed. Since the SCA and ABC show high diversity during its search mechanism, it can be observed from the results that the SCA and ABC have outperformed on this problem. The SCABC tries to achieve a stable stage of exploration and exploitation, and therefore, a leading guidance for the employed bees is added into the search mechanism. This leading guidance reduces the high diversity, and therefore, the performance

Table 3 Comparison of results obtained by ABC, SCA and proposed SCABC algorithms on classical benchmark problems with dimension 30

Test function	Algorithm	Best	Median	Average	Worst	STD
*F*1	ABC	6.97E−07	4.59E−06	8.28E−06	2.40E−05	7.35E−06
	SCA	3.59E−33	3.53E−27	3.43E−20	1.00E−18	1.83E−19
	SCABC	**1.95E−281**	**1.04E−277**	**2.25E−274**	**4.11E−273**	**0.00E+00**
*F*2	ABC	4.38E−08	6.41E−07	8.30E−07	3.50E−06	7.57E−07
	SCA	1.92E−34	1.72E−28	2.29E−23	5.15E−22	9.77E−23
	SCABC	**1.52E−283**	**1.09E−277**	**1.94E−276**	**3.35E−275**	**0.00E+00**
*F*3	ABC	1.43E−04	4.90E−04	5.54E−04	1.33E−03	2.91E−04
	SCA	7.47E−33	4.73E−27	1.01E−24	1.19E−23	2.90E−24
	SCABC	**3.76E−206**	**1.08E−202**	**8.46E−201**	**1.65E−199**	**0.00E+00**
*F*4	ABC	1.13E+04	1.57E+04	1.56E+04	1.95E+04	2.00E+03
	SCA	3.26E−03	1.25E+01	1.84E+02	2.93E+03	5.56E+02
	SCABC	**6.22E−56**	**3.97E−47**	**2.44E−41**	**7.01E−40**	**1.28E−40**
*F*5	ABC	3.34E+01	4.28E+01	4.17E+01	4.89E+01	4.07E+00
	SCA	4.19E−04	4.56E−02	1.69E−01	1.00E+00	2.54E−01
	SCABC	**1.47E−51**	**1.24E−46**	**4.43E−02**	**1.33E+00**	**2.43E−01**
*F*6	ABC	**1.29E+01**	3.19E+01	3.30E+01	5.80E+01	1.02E+01
	SCA	2.63E+01	2.73E+01	**2.73E+01**	**2.81E+01**	**5.20E−01**
	SCABC	2.04E+01	**2.28E+01**	3.39E+01	1.52E+02	3.42E+01
*F*7	ABC	5.31E−07	6.18E−06	7.55E−06	2.65E−05	6.73E−06
	SCA	3.24E+00	3.65E+00	3.70E+00	4.52E+00	3.03E−01
	SCABC	**5.39E−25**	**2.03E−24**	**2.89E−24**	**7.70E−24**	**2.10E−24**
*F*8	ABC	8.48E−21	1.90E−18	1.42E−17	1.70E−16	3.55E−17
	SCA	4.07E−45	4.15E−34	2.91E−26	7.76E−25	1.42E−25
	SCABC	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**
*F*9	ABC	8.39E−02	2.12E−01	2.06E−01	2.51E−01	3.67E−02
	SCA	**2.34E−04**	**1.93E−03**	2.95E−03	1.10E−02	2.67E−03
	SCABC	1.50E−03	2.66E−03	**2.68E−03**	**3.92E−03**	**5.93E−04**
*F*10	ABC	6.04E−17	2.76E−14	8.29E−14	5.15E−13	1.35E−13
	SCA	4.54E−75	3.97E−56	1.67E−38	5.01E−37	9.15E−38
	SCABC	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**
*F*11	ABC	− 1.19E+04	− 1.18E+04	− 1.18E+04	− 1.15E+04	1.05E+02
	SCA	− 4.75E+03	− 4.37E+03	− 4.35E+03	− 3.84E+03	2.47E+02
	SCABC	**− 1.26E+04**	**− 1.26E+04**	**− 1.26E+04**	**− 1.26E+04**	**2.89E−01**
*F*12	ABC	2.39E+00	5.57E+00	5.51E+00	7.86E+00	**1.33E+00**
	SCA	**0.00E+00**	**0.00E+00**	**2.45E−01**	**7.36E+00**	1.34E+00
	SCABC	2.29E+01	3.63E+01	3.61E+01	4.78E+01	6.72E+00
*F*13	ABC	1.68E−02	5.20E−02	5.95E−02	1.41E−01	3.28E−02
	SCA	7.99E−15	1.96E+01	1.33E+01	2.02E+01	8.79E+00
	SCABC	**8.88E−16**	**8.88E−16**	**8.88E−16**	**8.88E−16**	**0.00E+00**
*F*14	ABC	2.67E−05	9.52E−04	1.98E−03	**1.35E−02**	**2.76E−03**
	SCA	**0.00E+00**	**0.00E+00**	**1.85E−03**	5.56E−02	1.01E−02
	SCABC	**0.00E+00**	**0.00E+00**	1.46E−02	1.08E−01	2.33E−02
*F*15	ABC	1.27E−08	3.05E−07	**4.90E−07**	**2.11E−06**	**5.16E−07**
	SCA	2.80E−01	3.77E−01	3.64E−01	4.49E−01	4.79E−02
	SCABC	**3.34E−26**	**3.63E−24**	4.84E−02	3.11E−01	7.57E−02
*F*16	ABC	2.68E−07	**8.24E−06**	**1.15E−05**	**3.85E−05**	**1.03E−05**
	SCA	1.70E+00	2.08E+00	2.05E+00	2.31E+00	1.51E−01
	SCABC	**1.31E−24**	4.39E−02	6.67E−02	2.22E−01	7.66E−02

Table 3 (continued)

Test function	Algorithm	Best	Median	Average	Worst	STD
*F*17	ABC	2.88E−03	1.64E−02	1.68E−02	4.05E−02	9.65E−03
	SCA	**3.29E−26**	**1.31E−18**	2.02E−01	6.05E+00	1.11E+00
	SCABC	8.04E−07	2.78E−06	**2.95E−06**	**8.49E−06**	**1.66E−06**
*F*18	ABC	1.17E−08	7.03E−08	7.44E−08	2.44E−07	5.71E−08
	SCA	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**
	SCABC	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**
*F*19	ABC	1.90E+02	2.46E+02	2.42E+02	2.90E+02	2.40E+01
	SCA	5.55E−07	8.76E−05	1.03E−01	2.87E+00	5.22E−01
	SCABC	**2.21E−85**	**5.00E−82**	**3.23E−80**	**4.60E−79**	**9.43E−80**
*F*20	ABC	4.64E−03	7.96E−02	1.13E−01	3.30E−01	9.84E−02
	SCA	2.44E−33	1.33E−25	2.67E−19	7.90E−18	1.44E−18
	SCABC	**3.27E−277**	**1.37E−274**	**5.33E−272**	**1.26E−270**	**0.00E+00**
*F*21	ABC	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00
	SCA	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00
	SCABC	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00
*F*22	ABC	3.00E+00	4.00E+00	3.98E+00	4.40E+00	2.88E−01
	SCA	**9.99E−02**	**9.99E−02**	**1.20E−01**	**2.00E−01**	**4.07E−02**
	SCABC	**9.99E−02**	2.00E−01	1.80E−01	3.00E−01	4.84E−02
*F*23	ABC	4.93E−01	4.97E−01	4.97E−01	4.98E−01	1.24E−03
	SCA	**9.72E−03**	**9.72E−03**	**1.43E−02**	**3.72E−02**	1.04E−02
	SCABC	**9.72E−03**	3.72E−02	3.45E−02	**3.72E−02**	**8.34E−03**

of classical ABC is better than the SCABC algorithm on the problem *F*12. In the problem *F*14, the performance of the classical ABC and SCABC algorithms is very competitive to each other. On the 10-dimensional *F*14, the classical SCA performs much better than the SCABC algorithm, but for the dimension 30, the SCABC and classical SCA provide very competitive results. The presence of high diversity in the SCA is the reason for better performance on this instance as compared to the SCABC algorithm. In *F*17, the proposed SCABC algorithm is better than the classical ABC but worse than the classical SCA because this problem has large number of local optima which requires high diversity in the applied algorithm, and in the literature, it has been observed that the SCA is rich in diversity. In the problem *F*21, all the algorithms provide the optima (0). Hence, the performance of the proposed SCABC algorithm on most of the multimodal problems demonstrates the better search efficiency of the SCABC algorithm than the classical ABC and classical SCA in terms of better exploration and local optima avoidance ability.

Thus, the overall comparison of results based on best, average, median, worst and standard deviation of the objective function values verifies that the proposed SCABC is a better optimization method than the classical ABC and classical SCA.

4.2 Statistical validity of the results

In order to demonstrate the significant difference between the proposed search strategy in SCABC and algorithms ABC and SCA, a nonparametric Wilcoxon signed-rank test is applied on the obtained results from these algorithms. A nonparametric test is selected because it can be applied without any information about the distribution of data set of results. In the nonparametric test, the measure of central tendency is median and this can be considered as better representative to represent the significant improvement in the algorithm. In this section, the Wilcoxon test is applied at 5% significance level. The obtained *p*-values along with the outcomes obtained from Wilcoxon test are presented in Tables 4 and 5 corresponding to 10- and 30-dimensional benchmark problems. In the table, '+/−/=' signs are used to denote that the proposed SCABC algorithm is better than, worse than or similar to its competitor.

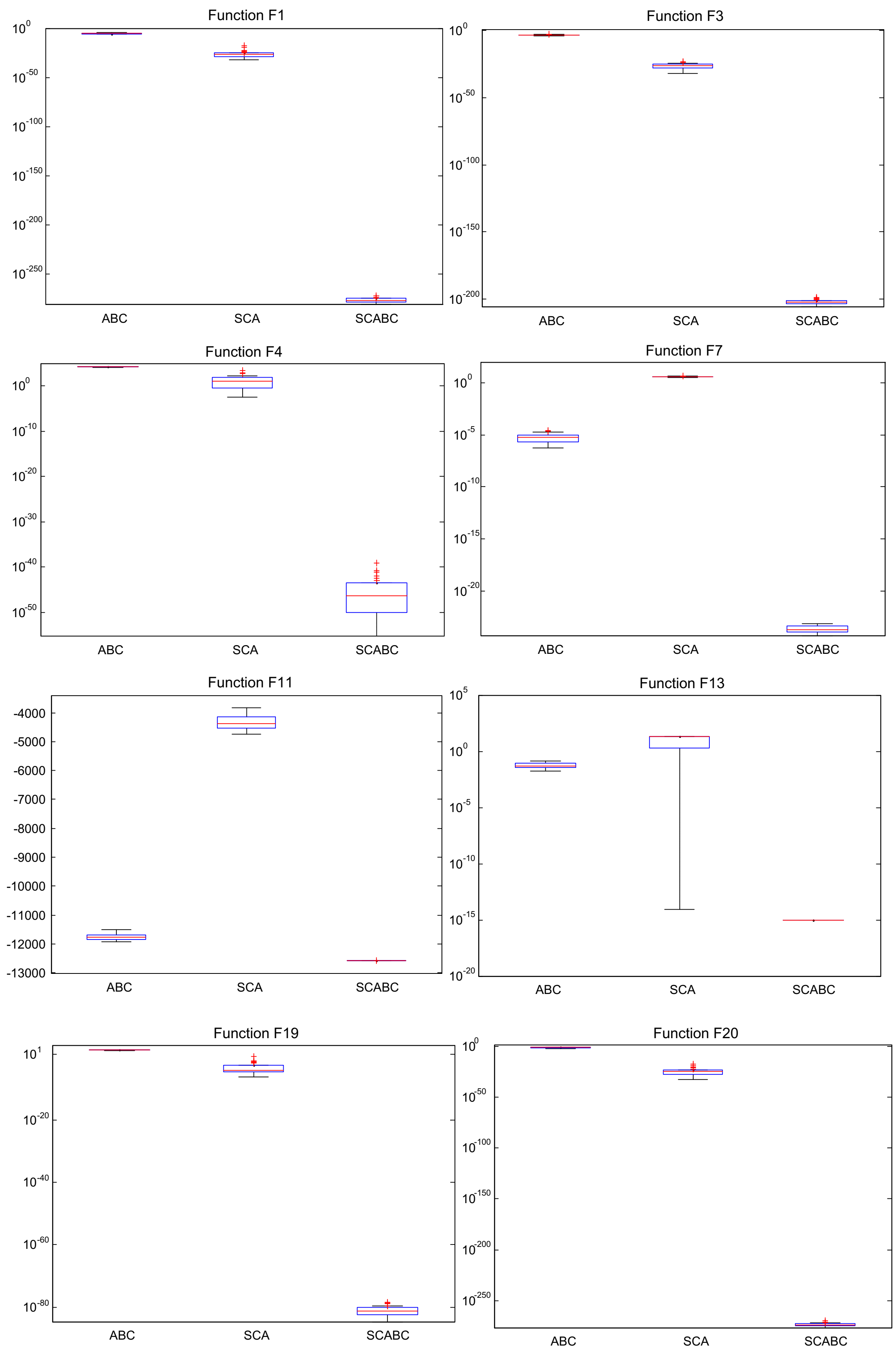


Fig. 2 Comparison based on boxplots

Table 4 Statistical outcomes obtained by applying Wilcoxon signed-rank test for 10-dimensional classical benchmark problems

Test function	SCABC versus ABC	SCABC versus SCA	Test function	SCABC versus ABC	SCABC versus SCA
*F*1	+ (1.74E−06)	+ (1.74E−06)	*F*13	+ (1.74E−06)	+ (1.31E−05)
*F*2	+ (1.74E−06)	+ (1.74E−06)	*F*14	− (2.35E−06)	− (3.79E−06)
*F*3	+ (1.74E−06)	+ (1.74E−06)	*F*15	= (6.43E−01)	= (6.43E−01)
*F*4	+ (1.74E−06)	+ (1.74E−06)	*F*16	= (3.71E−01)	+ (2.60E−06)
*F*5	+ (1.74E−06)	+ (1.74E−06)	*F*17	+ (1.74E−06)	− (1.74E−06)
*F*6	= (5.72E−02)	+ (1.48E−02)	*F*18	+ (1.74E−06)	= (1.00E+00)
*F*7	+ (1.74E−06)	+ (1.74E−06)	*F*19	+ (1.74E−06)	+ (1.74E−06)
*F*8	+ (1.74E−06)	+ (1.74E−06)	*F*20	+ (1.74E−06)	+ (1.74E−06)
*F*9	+ (1.74E−06)	− (1.74E−06)	*F*21	= (1.00E+00)	= (1.00E+00)
*F*10	+ (1.74E−06)	+ (1.74E−06)	*F*22	+ (1.74E−06)	+ (1.74E−06)
*F*11	+ (1.74E−06)	+ (1.74E−06)	*F*23	+ (1.74E−06)	− (3.11E−05)
*F*12	− (1.92E−06)	− (2.56E−06)			
			Overall outcomes		
			+/−/=	16/3/4	16/4/3

Table 5 Statistical outcomes obtained by applying Wilcoxon signed-rank test for 30-dimensional classical benchmark problems

Test function	SCABC versus ABC	SCABC versus SCA	Test function	SCABC versus ABC	SCABC versus SCA
*F*1	+ (1.74E−06)	+ (1.74E−06)	*F*13	+ (1.74E−06)	+ (1.74E−06)
*F*2	+ (1.74E−06)	+ (1.74E−06)	*F*14	= (5.71E−01)	− (4.18E−03)
*F*3	+ (1.74E−06)	+ (1.74E−06)	*F*15	= (3.82E−01)	+ (1.74E−06)
*F*4	+ (1.74E−06)	+ (1.74E−06)	*F*16	− (1.48E−03)	+ (1.74E−06)
*F*5	+ (1.74E−06)	+ (3.11E−05)	*F*17	+ (1.74E−06)	− (3.59E−04)
*F*6	+ (1.40E−02)	+ (2.77E−03)	*F*18	+ (1.74E−06)	= (1.00E+00)
*F*7	+ (1.74E−06)	+ (1.74E−06)	*F*19	+ (1.74E−06)	+ (1.74E−06)
*F*8	+ (1.74E−06)	+ (1.74E−06)	*F*20	+ (1.74E−06)	+ (1.74E−06)
*F*9	+ (1.74E−06)	= (1.48E−03)	*F*21	= (1.00E+00)	= (1.00E+00)
*F*10	+ (1.74E−06)	+ (1.74E−06)	*F*22	+ (1.74E−06)	− (4.68E−03)
*F*11	+ (1.74E−06)	+ (1.74E−06)	*F*23	+ (1.74E−06)	− (2.60E−05)
*F*12	− (1.74E−06)	− (1.74E−06)			
			Overall outcomes		
			+/−/=	20/2/1	15/5/3

4.3 Convergence behavior analysis

In this section, the convergence curves have been plotted to observe the convergence rate and search ability of algorithms to locate the optima of test problems. The convergence curves are plotted in Figs. 3 and 4 by considering the median value of objective functions obtained in 30 independent trials of algorithms. In Fig. 3, the convergence curves are shown for unimodal problems, and in Fig. 4, the convergence curves are plotted for multimodal problems. In these curves, the horizontal axis depicts the iterations and the vertical axis represents the objective function values. From the convergence curves of unimodal test problems, it can be assured that in terms of convergence rate the SCABC algorithm is better than ABC and SCA. From the convergence curves corresponding to the multimodal problems from *F*11 to *F*23, it can be observed that in most of the problems the SCABC algorithm has shown its better search efficiency as compared to ABC and SCA.

4.4 Performance index analysis

In this section, the proposed hybrid method SCABC is compared with ABC and SCA in terms of success rate,

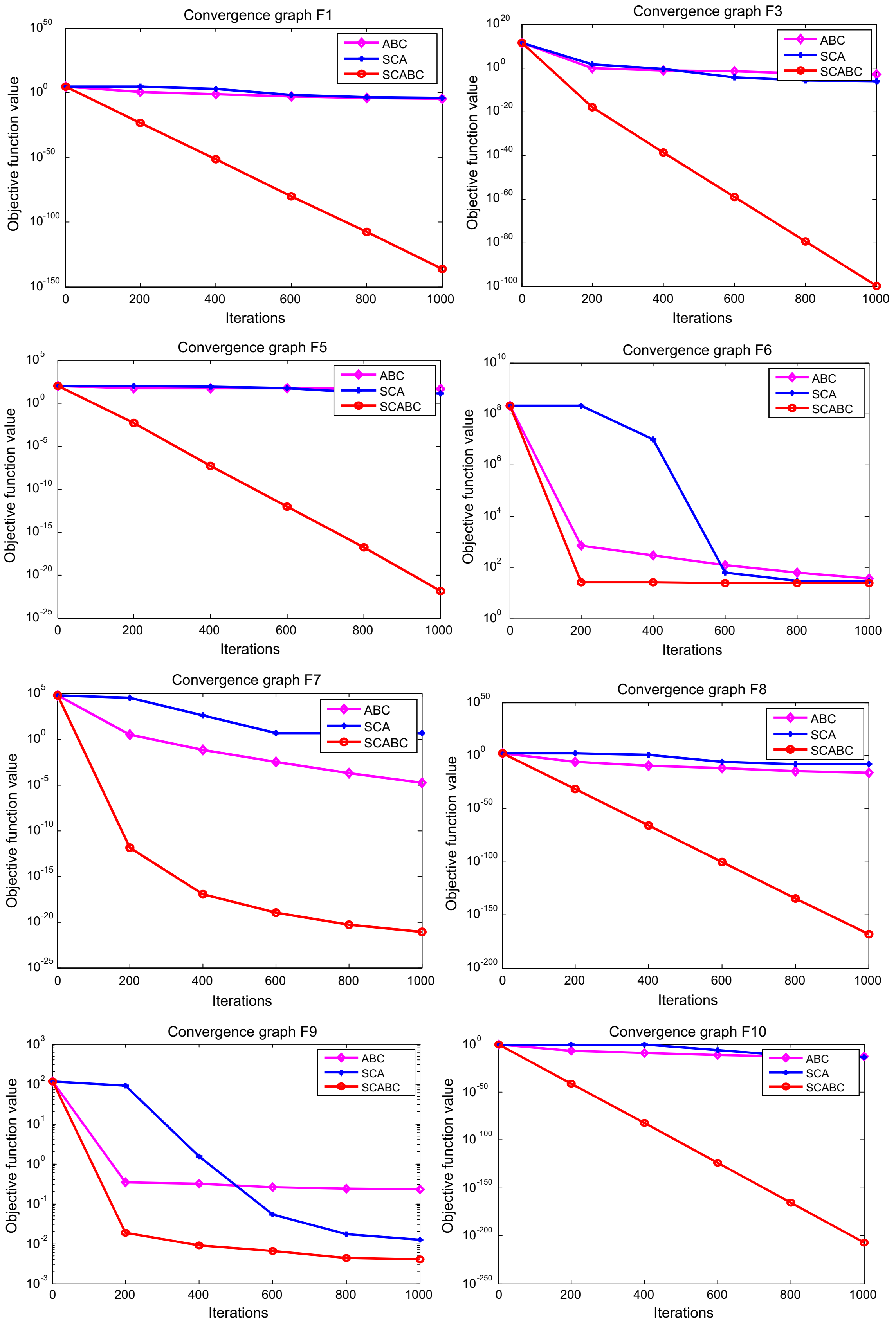


Fig. 3 Convergence curves corresponding to the unimodal problems

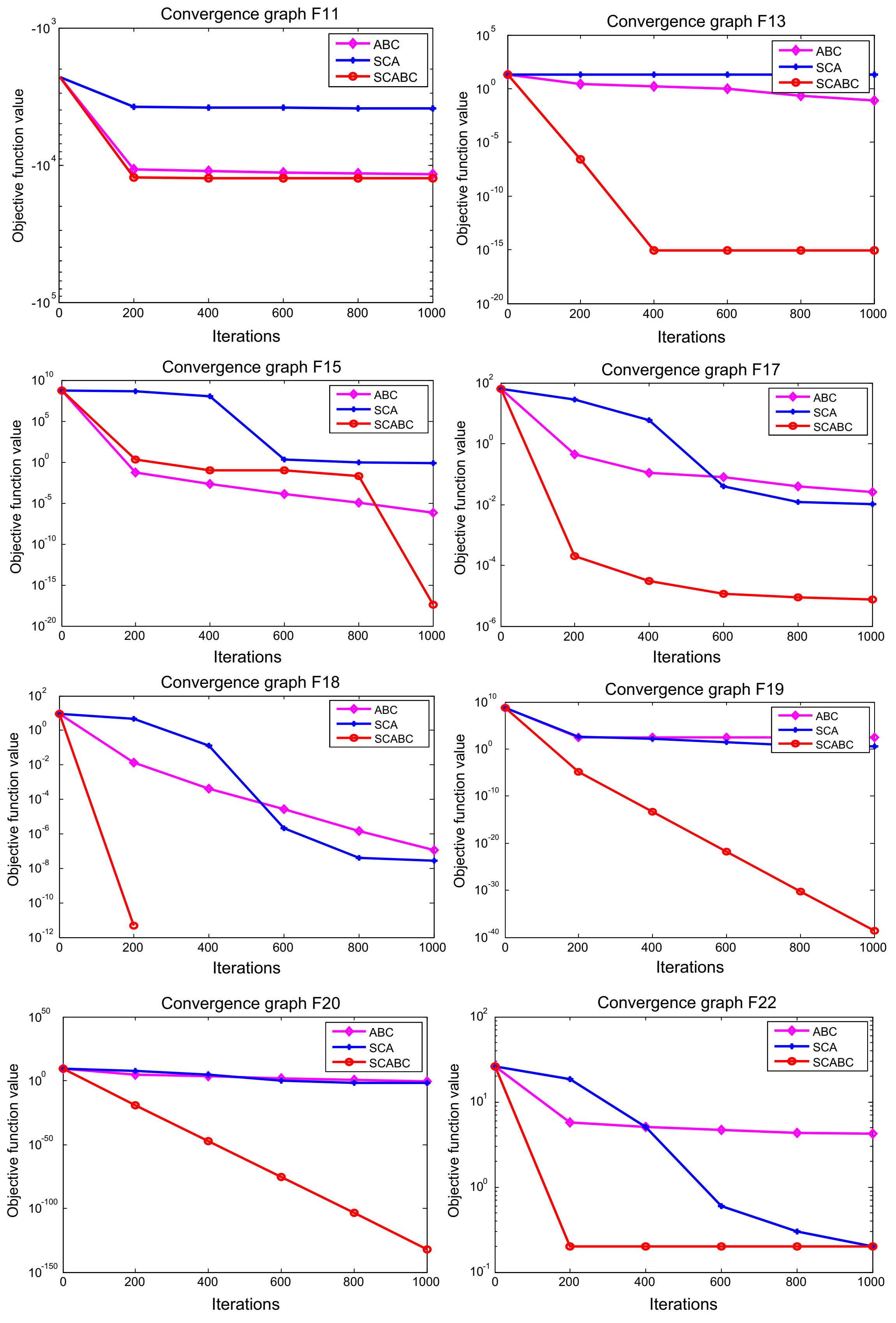
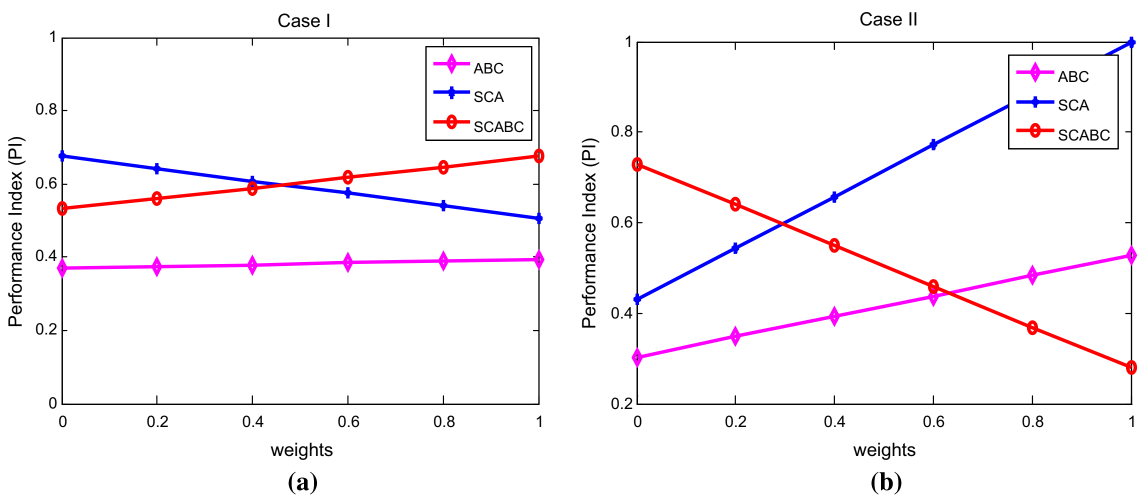


Fig. 4 Convergence curves corresponding to the multimodal problems



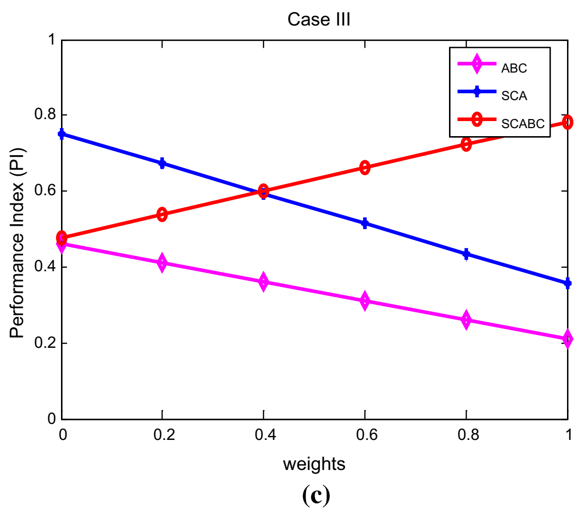


Fig. 5 Performance index (PI) for the cases I, II and III

time complexity and error value by performance index (PI) analysis metric [33]. The relative performance of an Algorithm A using PI can be calculated as follows:

$$\mathrm{PI}_{\mathrm{A}} = \frac{1}{N_{\mathrm{P}}}\sum_{i=1}^{N_{\mathrm{P}}} \mu_1\alpha_1 + \mu_2\alpha_2 + \mu_3\alpha_3 \tag{9}$$

where $\alpha_1 = \frac{\mathrm{SR}^i}{\mathrm{TR}^i}$, $\alpha_2 = \frac{\mathrm{MT}^i}{\mathrm{AT}^i}$, and $\alpha_3 = \frac{\mathrm{ME}^i}{\mathrm{AE}^i}$. SR^i: number of successful runs for the ith problem. TR^i: total number of runs conducted for the ith problem. MT^i: minimum of mean computational time taken by all the algorithms for the ith problem. AT^i: mean computational time taken by an algorithm for the ith problem. ME^i: minimum value of mean error obtained for the ith problem. AE^{P}: mean error obtained by an algorithm for the ith problem. N_{P}: total number of problems, and μ_1, μ_2 and μ_3 are nonnegative weights $(\mu_1 + \mu_2 + \mu_3 = 1)$ associated with success rate α_1, computational time term α_2, and error term α_3. The three different cases based on the weights assigned to the success, time and error terms are as follows:

Case I $\mu_1 = w, \mu_2 = \frac{(1-w)}{2}$ and $\mu_3 = \frac{(1-w)}{2}, 0 \le w \le 1$
Case II $\mu_1 = \frac{(1-w)}{2}, \mu_2 = w$ and $\mu_3 = \frac{(1-w)}{2}, 0 \le w \le 1$
Case III $\mu_1 = \frac{(1-w)}{2}, \mu_2 = \frac{(1-w)}{2}$ and $\mu_3 = w, 0 \le w \le 1$

The PI curves for all the above three cases are plotted in Fig. 5. In the case I, equal weights are provided to the error and computational time. It can be observed from Fig. 5a that PI of SCABC is higher than ABC for all the weights and higher than SCA for $w \ge 0.5$. In case II, equal weights are provided to the success and error term. From Fig. 5b, it is clear that the PI graph is decreased for SCABC with the weight w assigned to the computational time, which shows that SCABC takes more time to solve the problem as compared to SCA and ABC. In the case III, equal weights are provided to the computational time and success term. From Fig. 5c, it can be analyzed that PI of the proposed

Table 6 Comparison of various algorithms on 30 dimensional test problems

Test function	Basic PSO	mPSO	HS	SSA	MFO	FA	GABC	SCABC
*F*1	4.74E−03	1.03E−03	6.06E+04	4.27E−09	3.00E+03	1.80E−68	4.71E−21	**2.25E−274**
*F*2	6.65E−02	1.57E−02	8.50E+03	7.28E−09	7.27E+02	1.70E+00	6.76E−22	**1.94E−276**
*F*3	3.05E+01	3.28E+01	7.86E+10	4.37E−01	4.43E+01	1.47E+01	1.17E−11	**8.46E−201**
*F*4	9.72E+03	1.16E+04	7.75E+04	1.10E−07	1.94E+04	2.55E+03	1.47E+04	**2.44E−41**
*F*5	2.56E+01	2.75E+01	8.26E+01	1.19E+00	6.82E+01	8.61E+00	2.61E+01	**4.43E−02**
*F*6	1.35E+06	1.93E+06	1.99E+08	6.00E+01	1.05E+07	1.44E+02	**1.37E+00**	3.39E+01
*F*7	4.48E+03	5.55E+03	6.04E+04	4.25E−09	2.01E+03	**3.61E−32**	1.09E−20	2.89E−24
*F*8	7.26E−01	8.84E−01	9.94E+01	8.95E−25	4.21E+00	1.10E−05	6.32E−43	**0**
*F*9	9.72E−01	1.24E+00	1.01E+02	1.02E−02	2.51E+00	1.02E−01	6.43E−02	**2.68E−03**
*F*10	1.20E−04	1.56E−04	4.03E−01	2.41E−08	5.68E−115	4.78E−07	8.84E−22	**0**
*F*11	6.93E+03	6.87E+03	9.36E+03	4.75E+03	4.08E+03	5.53E+03	3.82E−04	**3.82E−04**
*F*12	2.04E+02	2.06E+02	4.05E+02	6.40E+01	1.60E+02	8.24E+01	**2.17E−07**	3.61E+01
*F*13	1.25E+01	1.32E+01	2.05E+01	1.99E+00	1.68E+01	2.04E+00	1.47E−09	**8.88E−16**
*F*14	4.37E+01	5.01E+01	5.47E+02	1.55E−02	2.71E+01	5.42E−03	**5.98E−09**	**1.46E−02**
*F*15	8.92E+03	4.46E+04	4.29E+08	1.73E+00	1.35E−01	2.71E+00	**4.70E−22**	4.84E−02
*F*16	1.01E+06	1.83E+06	8.82E+08	4.67E−03	4.29E−01	8.07E+00	**1.07E−19**	6.67E−02
*F*17	1.92E+01	2.02E+01	5.66E+01	2.95E+00	5.35E+00	2.48E+00	4.02E−05	2.95E−06
*F*18	2.57E+00	2.80E+00	8.78E+00	1.18E+00	5.90E−01	7.39E−01	**0**	**0**
*F*19	1.92E+02	1.96E+02	3.20E+08	1.48E−10	3.58E+02	1.80E+02	2.26E+02	**3.23E−80**
*F*20	9.00E+07	1.02E+08	2.26E+09	1.43E+06	8.50E+07	7.68E+06	1.48E−16	**5.33E−272**
*F*21	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
*F*22	7.28E+00	7.96E+00	2.51E+01	4.80E−01	6.54E+00	3.71E+00	1.69E+00	**1.80E−01**
*F*23	4.89E−01	4.90E−01	5.00E−01	1.53E−01	4.98E−01	4.93E−01	4.65E−01	**3.45E−02**

SCABC algorithm is higher than ABC for all the weights and higher than SCA for $w \geq 0.4$. Overall, in order to achieve comparatively more success and to provide less error, the proposed SCABC algorithm can be recommended over SCA and ABC.

4.5 Comparisons of the proposed SCABC algorithm with some other nature-inspired algorithms

This section compares the performance of the SCABC algorithm with some other nature-inspired optimization algorithms. The comparison has been performed at same parameter settings as used for SCABC algorithm on the same set of benchmark problems given in Table 1. The results obtained from various algorithms such as basic version of PSO (Basic PSO) [2]), modified PSO (mPSO) [34], harmony search (HS) [35], salp swarm algorithm (SSA) [36], moth-flame optimization (MFO) [37], firefly algorithm (FA) [38], gbest-guided ABC (GABC) [39] and proposed SCABC are presented in Table 6. In this table, the average of error values in objective fitness is presented. From the table, it can be observed that the proposed SCABC algorithm outperformed basic PSO, mPSO, HS, SSA and MFO for both the category (unimodal and multimodal) of benchmark problems. The comparison with the FA shows that the SCABC performs better than the FA on most of the problems except *F*7. Similarly, when the comparison is performed between the SCABC and GABC algorithms, it can be observed that for all the unimodal test problems (except *F*6) the SCABC is provided better results as compared to the GABC algorithm. The comparison on the multimodal problems shows the competitive performance of the SCABC algorithm with the GABC algorithm in terms of exploration strength.

Hence, an overall analysis shows that the SCABC algorithm can be preferred over all the algorithms basic PSO, mPSO, HS, SSA, MFO, FA and GABC for the unimodal problems. On the multimodal problems, the SCABC algorithm can be considered as a better optimization method as compared to the basic PSO, mPSO, HS, SSA, MFO and FA. The SCABC and GABC algorithms are very competitive to each other for multimodal problems.

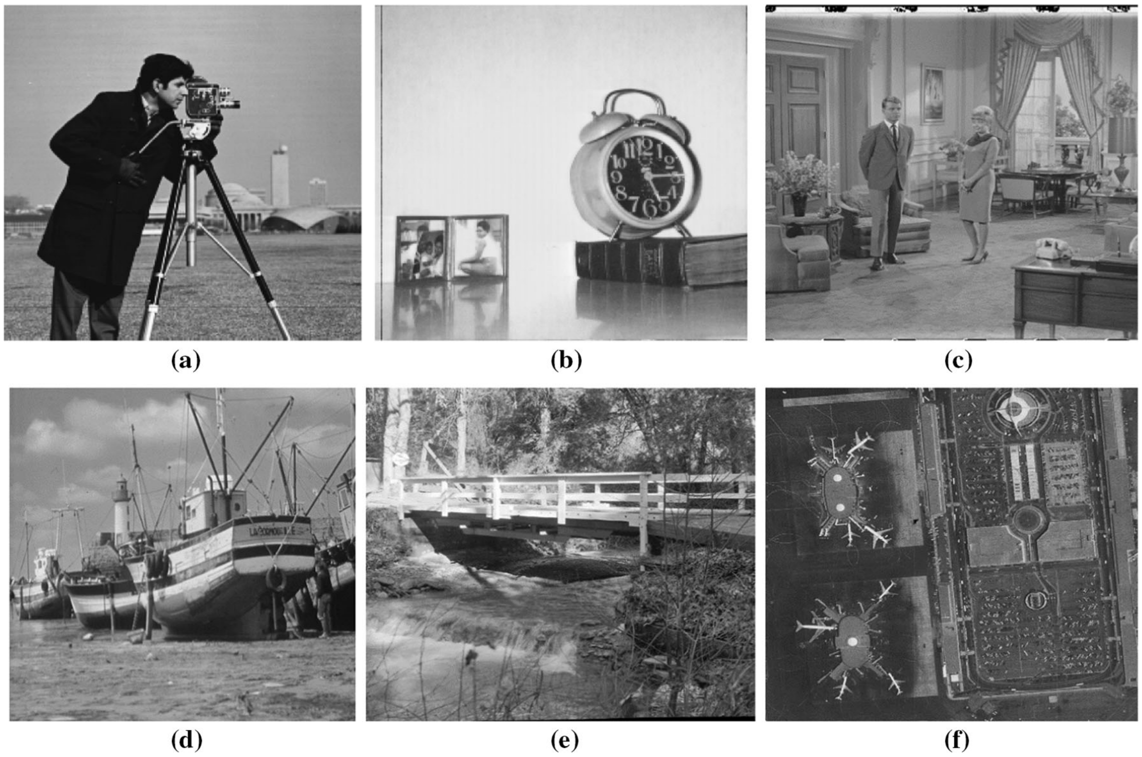


Fig. 6 Benchmark test images—**a** cameraman, **b** clock, **c** couple, **d** boat, **e** bridge and **f** airport

5 Application of SCABC algorithm in image segmentation

In the present section, the problem of multilevel gray image thresholding has been tried to solve using Kapur's entropy method [40] and proposed SCABC algorithm. Image segmentation is a process of splitting an image into various segments, where the main task is to simplify the representations of objects within the image for further processing. In multilevel thresholding, finding the optimal thresholds is very crucial for the image segmentation. Kapur's method (based on entropy criteria) is a well-known approach to find the optimal thresholds. In the present section, a brief description for the Kapur's entropy method is provided.

Consider an image defined as an 2D gray-level intensity function $f(x,y)$, where the values of $f(x,y)$ is the gray level, lies in the range $\{0,1,2,\ldots,L-1\}$. Let the number of pixels at intensity i are n_i and the total number of pixels in the image is N. The probability of occurrence for ith gray level can be obtained by: $p_i = \frac{n_i}{N}$. The Kapur's entropy method is defined as follows:

5.1 Kapur's entropy method

In order to determine the threshold values, the Kapur's method [40] maximizes the entropy of the segmented classes. The Kapur's method utilizes the concept from Shannon's entropy given in [41]. In the Kapur's method, the entropy of the image is defined with the assumption that

Table 7 Comparison of best fitness values obtained by ABC, SCA and proposed SCABC algorithm

Benchmark image	No. of thresholds	ABC	SCA	SCABC
Cameraman	**4**	18.4097	18.3371	**18.4279**
	5	21.1175	21.0455	**21.1774**
	6	23.7275	23.6769	**23.8035**
Clock	**4**	18.7340	18.6154	**18.7487**
	5	21.4865	21.4214	**21.5758**
	6	24.0789	23.9932	**24.1305**
Couple	**4**	18.4412	18.3478	**18.5118**
	5	21.1055	21.0602	**21.1622**
	6	23.5179	23.2803	**23.6089**
Boat	**4**	18.6471	18.6454	**18.6555**
	5	21.3526	21.2616	**21.3884**
	6	23.8510	23.7768	**23.9001**
Bridge	**4**	12.3363	12.3282	**12.3401**
	5	13.7242	13.6915	**13.7453**
	6	14.9891	14.9187	**15.0338**
Airport	**4**	18.2542	18.1707	**18.2662**
	5	21.0704	20.9964	**21.1084**
	6	23.6459	23.4445	**23.6580**

Table 8 Optimal threshold values obtained by ABC, SCA and proposed SCABC algorithm

Benchmark image	No. of thresholds	ABC	SCA	SCABC
Cameraman	**4**	42, 95, 144, 195	30, 89, 144, 193	44, 96, 145, 196
	5	24, 58, 102, 144, 200	28, 66, 112, 143, 195	24, 62, 98, 144, 196
	6	27, 60, 97, 129, 154, 199	31, 81, 110, 155, 192, 224	23, 63, 100, 144, 193, 220
Clock	**4**	32, 87, 147, 195	31, 98, 155, 190	32, 89, 143, 195
	5	31, 68, 118, 160, 203	30, 75, 130, 165, 198	32, 79, 119, 161, 202
	6	30, 71, 107, 143, 173, 201	32, 60, 95, 127, 175, 212	31, 67, 99, 131, 168, 205
Couple	**4**	66,112, 153, 204	74, 106, 172, 204	66, 110, 159, 204
	5	59, 91, 130, 167, 204	63, 109, 144, 178, 206	67, 101, 134, 168, 205
	6	62, 90, 122, 143, 173, 206	61, 81, 111, 144, 180, 201	57, 85, 109, 138, 171, 204
Boat	**4**	47, 89, 128, 182	48, 87, 127, 180	50, 91, 128, 181
	5	52, 99, 128, 174, 205	49, 88, 135, 175, 210	49, 91, 128, 173, 202
	6	47, 87, 124, 149, 176, 202	40, 73, 98, 136, 175, 196	43, 72, 102, 134, 175, 204
Bridge	**4**	53, 100, 153, 201	56, 103, 150, 198	53, 99, 148, 199
	5	37, 85, 122, 174, 219	49, 88, 135, 175, 210	40, 79, 123, 167, 208
	6	35, 78, 121, 153, 189, 223	36, 61, 105, 145, 177, 209	35, 73, 109, 145, 180, 219
Airport	**4**	29, 86, 147, 195	27, 84, 145, 197	29, 92, 150, 192
	5	28, 82, 124, 165, 202	28, 78, 118, 156, 197	29, 83, 122, 160, 199
	6	29, 88, 127, 154, 200, 242	30, 76, 118, 139, 175, 207	29, 70, 117, 152, 193, 242

Table 9 Comparison of mean fitness values obtained by ABC, SCA and proposed SCABC algorithm

Benchmark image	No. of thresholds	ABC	SCA	SCABC
Cameraman	**4**	18.3734	18.2298	**18.4207**
	5	21.0753	20.9071	**21.1691**
	6	23.6445	23.3932	**23.7690**
Clock	**4**	18.6761	18.5604	**18.7314**
	5	21.4183	21.1403	**21.5502**
	6	23.9408	23.7091	**24.0956**
Couple	**4**	18.3907	18.1643	**18.4616**
	5	21.0115	20.7332	**21.1305**
	6	23.4085	23.0211	**23.5616**
Boat	**4**	18.6240	18.5529	**18.6511**
	5	21.2899	21.1285	**21.3754**
	6	23.7947	23.6078	**23.8708**
Bridge	**4**	12.3096	12.2856	**12.3378**
	5	13.6998	13.6000	**13.7371**
	6	14.9381	14.7795	**15.0085**
Airport	**4**	18.2135	18.0957	**18.2464**
	5	21.0061	20.7502	**21.0972**
	6	23.5360	23.2741	**23.6211**

the image is represented by its gray-level histogram. If there are m number of thresholds $(T_1, T_2, \ldots, T_m)$ to be determined and these thresholds are dividing the image into $m+1$ classes, namely $C_0, C_1, C_2, \ldots, C_m$, then the Kapur's method does it by maximizing the objective function (fitness function):

$$f(T_1, T_2, \ldots T_m) = E_0 + E_1 + \cdots + E_m \tag{10}$$

where

$$E_0 = -\sum_{i=0}^{T_1-1} \frac{p_i}{\omega_0} \ln\left(\frac{p_i}{\omega_0}\right), \quad w_0 = \sum_{i=0}^{T_1-1} p_i$$

$$E_k = -\sum_{i=T_k}^{T_{k+1}-1} \frac{p_i}{\omega_k} \ln\left(\frac{p_i}{\omega_k}\right), \quad w_k = \sum_{i=T_k}^{T_{k+1}-1} p_i, \quad k = 1, 2, \ldots, m-1$$

$$E_m = -\sum_{i=T_m}^{L-1} \frac{p_i}{\omega_m} \ln\left(\frac{p_i}{\omega_m}\right), \quad w_m = \sum_{i=T_m}^{L-1} p_i$$

where $E_0, E_1, \ldots, E_m$ are the Kapur's entropy and $\omega_0, \omega_1, \ldots, \omega_m$ represent the class probabilities of the segmented classes $C_0, C_1, C_2, \ldots, C_m$, respectively.

5.2 Experimental setup

The present section provides a brief description of experimental setup for the proposed hybrid method SCABC. In the present study of multilevel thresholding, six gray benchmark images—cameraman, clock, couple, boat, bridge and airport—are considered. These images are picked from USC-SIPI Image Database and presented in Fig. 6.

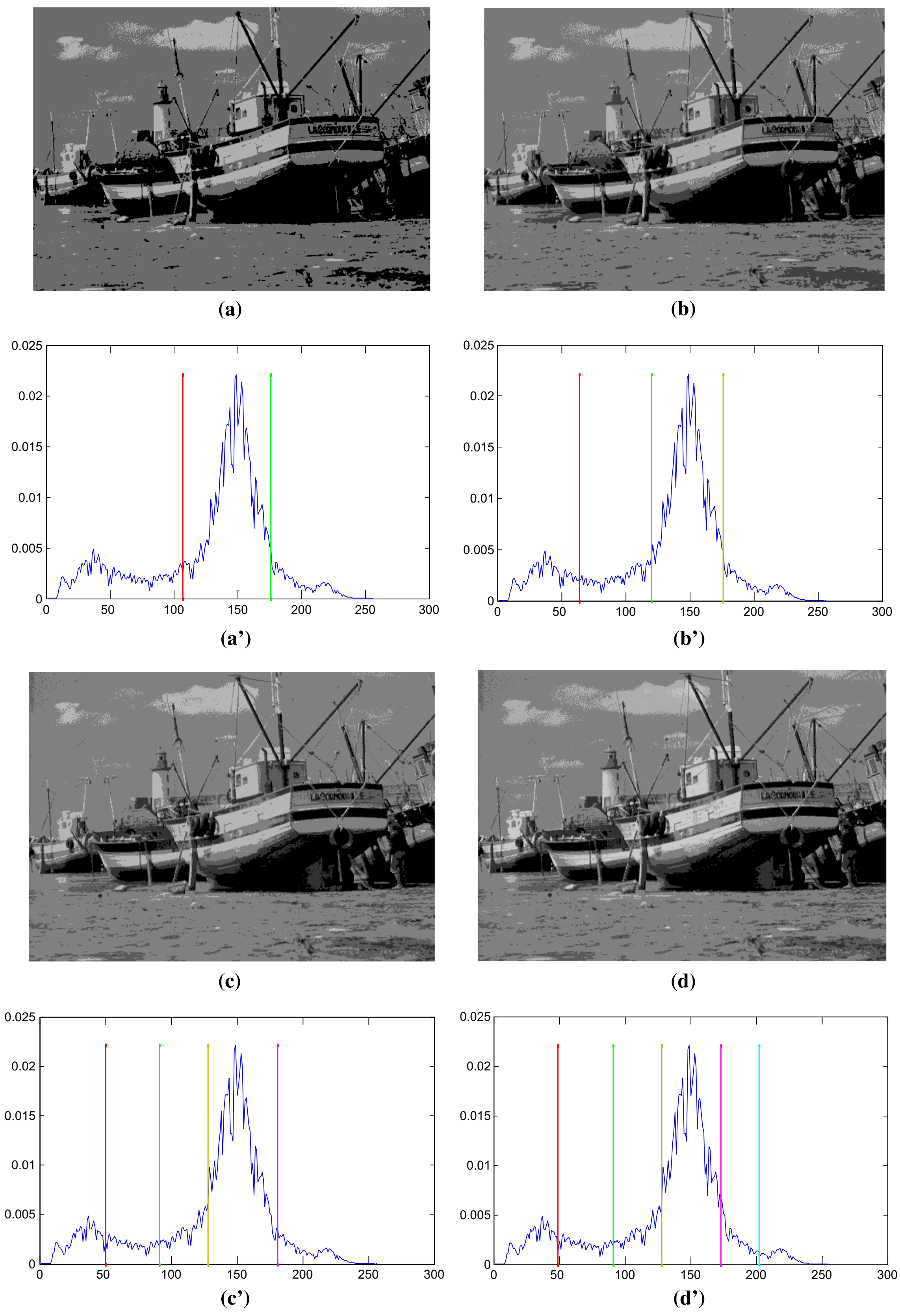


Fig. 7 Thresholded boat test image by SCABC algorithm corresponding to Kapur's method. **a–d** the segmented images of three, four, five, and six classes, respectively. **a'–d'** the thresholds for the segmented images. **a–d** The fitted histogram

Since the proposed method, SCABC is the hybridized version of SCA and ABC; therefore, for the performance comparison of SCABC is done with ABC and SCA with same parameter settings of termination criteria. In the proposed SCABC algorithm, 12 food sources/candidate solutions and 100 iterations are fixed as termination criteria of the algorithm. The same parameter setting is used for ABC and SCA. All the algorithms are implemented in MATLAB 2014a with Intel core-i5 @ 2.30 GHz.

In order to evaluate the quality of segmented images, a famous metric known as peak-signal-to-noise ratio (PSNR) is used in the paper. The PSNR metric depends directly on the image intensity and demonstrates the accuracy of the segmented image. The PSNR value can be determined as follows:

$$\mathrm{PSNR} = 10\log_{10}\left(\frac{255^2}{\mathrm{MSE}}\right) \tag{11}$$

$$\mathrm{MSE} = \frac{1}{MN}\sum_{i=1}^{M}\sum_{j=1}^{N}[I(i,j) - J(i,j)]^2 \tag{12}$$

where I and J represent the original and segmented images, respectively.

5.3 Results and analysis

This section presents the numerical results of the proposed multilevel threshold scheme SCABC. The results in our study are produced with the same parameter setting as provided in Sect. 5.2. The quality of segmented images is compared by the objective fitness given by Kapur's entropy method and by the PSNR measure.

The best objective function values obtained by implementing the SCABC, ABC and SCA algorithms using Kapur's method are presented in Table 7, and their corresponding values of thresholds are presented in Table 8. The obtained mean objective function values are reported in Table 9. For a small number of threshold values (m = 2 or 3), the objective function values are practically the same. Therefore, in the present study, results are presented corresponding to large number of thresholds (m = 4, 5 and 6). From the presented results, it can be observed that the SCABC algorithm has achieved higher value of objective function in all test images as compared to ABC and SCA. An example of segmented test image (boat) obtained by the SCABC algorithm is presented in Fig. 7. In the same figure, the fitted histogram and locations of thresholds for segmented images are also presented. The obtained mean PSNR values of segmented images are presented in Table 10 by implementing the SCABC, ABC and SCA. The table clearly indicates either competitive or better quality of segmented images obtained from the proposed hybrid method SCABC than other algorithms. Hence, the overall analysis in terms of various performance metrics signifies the better ability of proposed hybrid method called SCABC.

Table 10 Comparison of mean PSNR values obtained by ABC, SCA and proposed SCABC algorithm

Benchmark image	No. of thresholds	ABC	SCA	SCABC
Cameraman	**4**	19.9105	19.5164	**20.1313**
	5	**21.1796**	20.5699	20.6954
	6	22.0512	21.3754	**22.1603**
Clock	**4**	18.4710	18.1906	**18.5424**
	5	19.8144	19.3573	**20.3031**
	6	**21.5355**	20.5194	20.9136
Couple	**4**	18.841	18.6372	**18.9570**
	5	20.2280	19.9108	**21.0229**
	6	21.9401	21.6067	**22.2123**
Boat	**4**	19.7091	19.4111	**19.7129**
	5	20.5228	20.2665	**20.5430**
	6	21.2695	21.3011	**21.5426**
Bridge	**4**	18.8976	18.7352	**19.0573**
	5	20.0215	19.7498	**20.5256**
	6	21.2760	21.3088	**21.7939**
Airport	**4**	**18.0771**	16.3785	17.5812
	5	**19.2469**	18.2337	19.0182
	6	20.0428	20.3692	**20.7885**

6 Conclusions

In this paper, the hybrid algorithm called SCABC has been proposed for the global optimization. First, the problem of high diversity at early generations and insufficient diversity at later generations of SCA is tried to alleviate by modifying its search equations and integrating memory-based information. Second, the employed bee phase of classical ABC algorithm has been replaced with the modified search equations of SCA in order to explore and exploit the search space more properly. The proposed search strategy for employed bees can be considered more efficient as compared to original one. To verify this fact, a classical benchmark set of 23 problems has been considered in the paper. The effectiveness and efficiency of the proposed SCABC algorithm can be assured against SCA and ABC based on the performance on considered benchmarks. Various other measures such as convergence analysis, performance index analysis and statistical analysis also ensure the better convergence rate and significant improvement in the search strategy of the SCABC algorithm. Therefore, the overall analysis recommends SCABC as a better optimization method than SCA and ABC algorithms. The comparison with some other algorithms also verifies the better or competitive search ability of the SCABC algorithm. To ensure the effectiveness of the SCABC algorithm on real-life applications, the multilevel thresholding problem is considered. The performance analysis on thresholding problem based on various metrics shows the better ability of the SCABC algorithm as compared to the ABC and SCA.

In future, we will focus on the other real-life applications of proposed SCABC algorithm based on the performance on test problems. The multi-objective and binary SCABC can also be developed in the future.

Acknowledgements The first author would like to thank Ministry of Human Resources, Government of India, for funding this research. Grant No. MHR-02-41-113-429.

Compliance with ethical standards

Conflict of interest The authors declare that they have no conflict of interest.

References

1. Wolpert DH, Macready WG (1997) No free lunch theorems for optimization. IEEE Trans Evol Comput 1(1):67–82
2. Eberhart R, Kennedy J (1995) A new optimizer using particle swarm theory. In: Proceedings of the 6th international symposium on micro machine and human science, 1995. MHS'95. IEEE, pp 39–43
3. Dorigo M, Birattari M, Stutzle T (2006) Ant colony optimization. Comput Intell Mag IEEE 1:28–39
4. Karaboga D, Basturk B (2007) A powerful and efficient algorithm for numerical function optimization: artificial bee colony (ABC) algorithm. J Glob Optim 39(3):459–471
5. Yang XS, Deb S (2009) Cuckoo search via Lévy flights. In: World congress on nature and biologically inspired computing, 2009. NaBIC 2009. IEEE, pp 210–214
6. Yang XS (2010) Firefly algorithm, stochastic test functions and design optimisation. Int J Bio Inspired Comput 2:78–84
7. Gao WF, Liu SY, Huang LL (2013) A novel artificial bee colony algorithm based on modified search equation and orthogonal learning. IEEE Trans Cybern 43(3):1011–1024
8. Zhu G, Kwong S (2010) Gbest-guided artificial bee colony algorithm for numerical function optimization. Appl Math Comput 217(7):3166–3173
9. Liu J, Zhu H, Ma Q, Zhang L, Xu H (2015) An artificial bee colony algorithm with guide of global and local optima and asynchronous scaling factors for numerical optimization. Appl Soft Comput 37:608–618
10. Xiang WL, An MQ (2013) An efficient and robust artificial bee colony algorithm for numerical optimization. Comput Oper Res 40(5):1256–1265
11. Kiran MS, Hakli H, Gunduz M, Uguz H (2015) Artificial bee colony algorithm with variable search strategy for continuous optimization. Inf Sci 300:140–157
12. Yurtkuran A, Emel E (2015) An adaptive artificial bee colony algorithm for global optimization. Appl Math Comput 271:1004–1023
13. Wang H, Wu Z, Rahnamayan S, Sun H, Liu Y, Pan JS (2014) Multi-strategy ensemble artificial bee colony algorithm. Inf Sci 279:587–603
14. Nseef SK, Abdullah S, Turky A, Kendall G (2016) An adaptive multi-population artificial bee colony algorithm for dynamic optimisation problems. Knowl Based Syst 104:14–23
15. Sharma H, Bansal JC, Arya KV, Yang XS (2016) Lévy flight artificial bee colony algorithm. Int J Syst Sci 47(11):2652–2670
16. Zhou X, Wang H, Wang M, Wan J (2017) Enhancing the modified artificial bee colony algorithm with neighborhood search. Soft Comput 21(10):2733–2743
17. Gaidhane PJ, Nigam MJ (2018) A hybrid grey wolf optimizer and artificial bee colony algorithm for enhancing the performance of complex systems. J Comput Sci 27:284–302
18. Lu R, Hu H, Xi M, Gao H, Pun CM (2019) An improved artificial bee colony algorithm with fast strategy, and its application. Comput Electr Eng 78:79–88
19. Murugan R, Mohan MR, Rajan CCA, Sundari PD, Arunachalam S (2018) Hybridizing bat algorithm with artificial bee colony for combined heat and power economic dispatch. Appl Soft Comput 72:189–217
20. Karaboga D, Gorkemli B, Ozturk C, Karaboga N (2014) A comprehensive survey: artificial bee colony (ABC) algorithm and applications. Artif Intell Rev 42(1):21–57
21. Javidrad F, Nazari M (2017) A new hybrid particle swarm and simulated annealing stochastic optimization method. Appl Soft Comput 60:634–654
22. Jitkongchuen D (2015) A hybrid differential evolution with grey wolf optimizer for continuous global optimization. In: 7th International conference on information technology and electrical engineering (ICITEE), 2015. IEEE, pp 51–54
23. Shankar T, Shanmugavel S, Rajesh A (2016) Hybrid HSA and PSO algorithm for energy efficient cluster head selection in wireless sensor networks. Swarm Evol Comput 30:1–10
24. Tawhid MA, Ali AF (2017) A hybrid grey wolf optimizer and genetic algorithm for minimizing potential energy function. Memet Comput 9(4):347–359
25. Zhang X, Kang Q, Cheng J, Wang X (2018) A novel hybrid algorithm based on biogeography-based optimization and grey wolf optimizer. Appl Soft Comput 67:197–214
26. Aydilek İB (2018) A hybrid firefly and particle swarm optimization algorithm for computationally expensive numerical problems. Appl Soft Comput 66:232–249
27. Mirjalili S (2016) SCA: a sine cosine algorithm for solving optimization problems. Knowl Based Syst 96:120–133
28. Nenavath H, Jatoth RK, Das S (2018) A synergy of the sine-cosine algorithm and particle swarm optimizer for improved global optimization and object tracking. Swarm Evol Comput 43:1–30
29. Gao W, Liu S, Huang L (2012) A global best artificial bee colony algorithm for global optimization. J Comput Appl Math 236(11):2741–2753
30. Long W, Jiao J, Liang X, Tang M (2018) An exploration-enhanced grey wolf optimizer to solve high-dimensional numerical optimization. Eng Appl Artif Intell 68:63–80
31. Gao WF, Liu SY, Huang LL (2014) Enhancing artificial bee colony algorithm using more information-based search equations. Inf Sci 270:112–133
32. Mirjalili S, Mirjalili SM, Lewis A (2014) Grey wolf optimizer. Adv Eng Softw 69:46–61
33. Deep K, Thakur M (2007) A new mutation operator for real coded genetic algorithms. Appl Math Comput 193(1):211–230

34. Shi Y, Eberhart R (1998) A modified particle swarm optimizer. In: The 1998 IEEE international conference on evolutionary computation proceedings, 1998. IEEE world congress on computational intelligence. IEEE, pp 69–73
35. Geem ZW, Kim JH, Loganathan GV (2001) A new heuristic optimization algorithm: harmony search. Simulation 76(2):60–68
36. Mirjalili S, Gandomi AH, Mirjalili SZ, Saremi S, Faris H, Mirjalili SM (2017) Salp swarm algorithm: a bio-inspired optimizer for engineering design problems. Adv Eng Softw 114:163–191
37. Mirjalili S (2015) Moth-flame optimization algorithm: a novel nature-inspired heuristic paradigm. Knowl Based Syst 89:228–249
38. Yang XS (2010) Firefly algorithm, stochastic test functions and design optimisation. arXiv preprint arXiv:1003.1409
39. Zhu G, Kwong S (2010) Gbest-guided artificial bee colony algorithm for numerical function optimization. Appl Math Comput 217(7):3166–3173
40. Kapur JN, Sahoo PK, Wong AK (1985) A new method for gray-level picture thresholding using the entropy of the histogram. Comput Vis Graph Image Process 29(3):273–285
41. Shannon C (1948) A mathematical theory of communication. Bell Syst Tech J 27:379–423