Expert Systems With Applications 119 (2019) 210–230

Contents lists available at ScienceDirect

Expert Systems With Applications

journal homepage: www.elsevier.com/locate/eswa

A hybrid self-adaptive sine cosine algorithm with opposition based learning

Shubham Gupta*, Kusum Deep

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

ARTICLE INFO

Article history:
Received 5 July 2018
Revised 15 September 2018
Accepted 31 October 2018
Available online 1 November 2018

Keywords:
Population based algorithms
Sine Cosine algorithm (SCA)
Opposition based learning
Self-adaptation
Benchmark test problems
Engineering application problems

ABSTRACT

Real-world optimization problems demand an efficient meta-heuristic algorithm which maintains the diversity of solutions and properly exploits the search space of the problem to find the global optimal solution. Sine Cosine Algorithm (SCA) is a recently developed population-based meta-heuristic algorithm for solving global optimization problems. SCA uses the characteristics of sine and cosine trigonometric functions to update the solutions. But, like other population-based optimization algorithms, SCA also suffers the problem of low diversity, stagnation in local optima and skipping of true solutions. Therefore, in the present work, an attempt has been made towards the eradication of these issues, by proposing a modified version of SCA. The proposed algorithm is named as modified Sine Cosine Algorithm (m-SCA). In m-SCA, the opposite population is generated using opposite numbers based on perturbation rate to jump out from the local optima. Secondly, in the search equations of SCA self-adaptive component is added to exploit all the promising search regions which are pre-visited. To evaluate the effectiveness in solving the global optimization problems, m-SCA has been tested on two sets of benchmark problems – classical set of 23 well-known benchmark problems and standard IEEE CEC 2014 benchmark test problems. In the paper, the performance of proposed algorithm m-SCA is also tested on five engineering optimization problems. The conducted statistical, convergence and average distance analysis demonstrate the efficacy of the proposed algorithm to determine the efficient solution of real-life global optimization problems.

1. Introduction

The term optimization refers to finding the best solution of the problem under the available options. Optimization is used in many fields like science, engineering, business, management etc. Generally, the optimization problems are formulated into a mathematical structure to find their solutions. To find the solution of these optimization problems various deterministic techniques are developed in the literature. Although the deterministic optimization techniques provide the optima of the problems but they have certain limitations like continuity, differentiability, convexity, unimodality etc. But many real-world optimization problems do not have these special mathematical features. Therefore to deal with such cases, researchers have designed optimization techniques which do not use the information of the problem and considered the optimization problem as a black box. Generally, these techniques use the random numbers in the search process and therefore these techniques are known as stochastic or probabilistic techniques. The integration of random numbers provides a random search in the search space. In these algorithms, a guided search is also involves which provides the movement around the best solutions obtained so far. Based on the number of solutions these algorithms are divided into two categories – Single solution based algorithms and multiple solutions based algorithms. In single solution based algorithms, the search process is progressed by a single solution. Hill Climbing (Hoffmann & Nebel, 2001), Simulated Annealing (Van Laarhoven & Aarts, 1987) are some single solution based algorithms. In multiple solution based algorithms, search process is followed by a set of solutions to locate the optima of the problem. Genetic algorithm (GA) (Holland, 1992), Particle Swarm Optimization (PSO) (Kennedy, 2011), Firefly Algorithm (Yang, 2010), Artificial Bee Colony (ABC) algorithm (Karaboga & Basturk, 2007), Moth-flame optimization (MFO) (Mirjalili, 2015a), Harmony search (Geem, Kim, & Loganathan, 2001), algorithm are some examples of population-based algorithms. The advantage of using set of solutions instead of considering a single solution is to increase the diversity of solutions in a search space so that large area of search space can be explored.

Over the last few decades, the population-based optimization algorithms have received increased attention due to their ability to solve real-life optimization problems. These algorithms are

* Corresponding author.
E-mail addresses: sgupta@ma.iitr.ac.in (S. Gupta), kusumfma@iitr.ac.in (K. Deep).

https://doi.org/10.1016/j.eswa.2018.10.050
0957-4174/

extremely popular among researchers because of their simplicity, flexibility and the ability to skip local optima. These techniques are developed from the inspiration of nature because nature has been considered as most powerful strength and is the origin of everything present in the universe. Therefore, these optimization algorithms (techniques) are known as Nature Inspired Optimization Algorithms. In the nature inspired optimization Algorithms, the exploration and exploitation which are conflicting operators (Črepinšek, Liu, & Mernik, 2013) work to search for optima of the problem. In the phase of exploration, new search regions of a feasible space are discovered and in exploitation phase, the potential of solutions around the previously discovered search regions is analyzed. Therefore an optimization algorithm should be capable of addressing and balancing these two important operators to estimate the global optima of the problem.

The Nature Inspired Optimization Algorithms have shown their potential in various application problems (Kar, 2016; Mavrovouniotis, Li, & Yang, 2017). Some recent applications of these algorithms are - In Woźniak and Połap (2018a) bio-inspired methods have been used for respiratory disease detection from medical images. In Singh, Kumar, Balyan, and Singh (2017), adaptive swarm intelligence optimized dark image enhancement approach is proposed for remotely sensed satellite images. In Djenouri, Belhadi, and Belkebir (2018) a new algorithm Bees swarm optimization is proposed for document information retrieval. In Ertenlice and Kalayci (2018), the authors present a survey of Swarm Intelligence based algorithms for portfolio optimization. In Woźniak and Połap (2018b) hybrid models of neural network and heuristic techniques are used for fruit peel defects detection. In Kumawat, Nanda, and Maddila (2018) Whale Optimization Algorithm has been used for positioning LED Panel for Uniform Illuminance in Indoor VLC System. For multilevel thresholding in image segmentation various nature inspired algorithms are used (El Aziz, Ewees, & Hassanien, 2017; Khairuzzaman & Chaudhury, 2017).

The nature inspired techniques can be classified into three categories - (*i*) Evolutionary Optimization Algorithms, (*ii*) Swarm Intelligence based algorithms and (*iii*) Physical phenomenon based optimization algorithms.

In Evolutionary techniques, the algorithms are developed from the inspiration of theory of evolution in nature. In these algorithms, previous population dies in each generation. Genetic Algorithms (GA) (Holland, 1992), Differential Evolution (DE) (Storn & Price, 1997) are some examples of Evolutionary Algorithms. Physical phenomenon based algorithms are designed by mimicking the physical rules. Gravitational Search Algorithm (GSA) (Rashedi, Nezamabadi-Pour, & Saryazdi, 2009), Black Hole (BH) algorithm (Hatamlou, 2013), Central Force Optimization (CFO) (Formato, 2007), Gravitational Local Search Algorithm (GLSA) (Webster & Bernhard, 2003), Ray Optimization (RO) algorithm (Kaveh & Khayatazad, 2012) are some examples of physics based algorithms.

Swarm Intelligence based algorithms mimics the intelligent, collective and social behavior of different creatures like birds, ants, whale, bees, cuckoo, wolves etc. to determine the global optima. Some most popular and efficient algorithms that are being used widely are - Particle Swarm Optimization (PSO) (Kennedy, 2011), Grey Wolf Optimizer (GWO) (Mirjalili, Mirjalili, & Lewis, 2014), Ant Lion Optimization (ACO) (Mirjalili, 2015b), Spider Monkey Optimization (SMO) (Bansal, Sharma, Jadon, & Clerc, 2014), Whale Optimization Algorithm (WOA) (Mirjalili & Lewis, 2016), Ant Colony Optimization (ACO) (Dorigo & Birattari, 2011), Artificial Bee Colony (ABC) (Karaboga & Basturk, 2007) algorithm, Salp Swarm Algorithm (SSA) (Mirjalili et al., 2017), Polar Bear Optimization Algorithm (Połap, 2017), Grasshopper optimization algorithm (GOA) (Saremi, Mirjalili, & Lewis, 2017). In all these algorithms rather than killing the previous population of solutions, the previous best positions are saved in memory and in each generation, search agents share the information regarding the feasible and optimal direction of solutions within the search space. Nowadays swarm intelligence based algorithms are used on a large scale to solve the real-world optimization problems.

From some past recent years many new algorithms have been developed, this fact is related to the No Free Lunch (NFL) theorem (Wolpert & Macready, 1997). The NFL theorem states that an ideal optimization algorithm can't be designed which can solve all the optimization problems. In other words, if an optimization algorithm performs very well on some set of optimization problems then there exist some other set of optimization problems on which this particular algorithm will not perform well i.e. on a set of all optimization problems, the performance of each optimization algorithm is equivalent.

Recently, another population-based meta-heuristic algorithm called Sine Cosine Algorithm (SCA) (Mirjalili, 2016) was proposed by Mirjalili for global optimization problems, where the characteristics of sine and cosine functions are used. In Mirjalili (2016) the competitive performance of SCA is presented experimentally with other meta-heuristic search algorithms like PSO, GA, GSA etc.

The advantageous of SCA in exploring and exploiting the search space efficiently motivate the researchers to use it for various real-life applications problems. For example in Hafez, Zawbaa, Emary, and Hassanien (2016), SCA has been utilized for feature selection problem. In Sindhu, Ngadiran, Yacob, Zahri, and Hariharan (2017), an elitist strategy is incorporated in SCA with improved search equations to solve the feature selection problem. In Kumar and Kumar (2017), for data clustering, SCA has been used. In Ismael, Aleem, and Abdelaziz (2017), to find the optimal size of conductors in a distribution system SCA is used. In Sahlol, Ewees, Hemdan, and Hassanien (2016), feedforward neural network has been trained using SCA. Li, Fang, and Liu (2018) have used SCA for optimizing the parameters of support vector regression. In Nenavath, Jatoth, and Das (2018), SCA has been hybridized with PSO to solve the object tracking problem.

From the experimental analysis of SCA, it is observed that SCA falls into the local solutions during the search process and suffers deficiency of exploitation. These problems have been analyzed from the performance of SCA on unimodal and multimodal benchmark test problems which are used to evaluate the exploitation and exploration (and/or local optima avoidance) potential of problem's search space. Therefore, in Elaziz, Oliva, and Xiong (2017), an improved SCA based on opposition based learning is proposed which tries to prevents the population from local optima, but from the provided results it can be analyzed that this algorithm also suffers from the problem of stagnation and premature convergence. In Reddy, Panwar, Panigrahi, and Kumar (2017) a binary version of SCA is proposed to solve the problems having binary search space.

Therefore, the present work aims to increase the diversity of solutions and local exploitation of search space in SCA by proposing a modified version of classical Sine Cosine Algorithm. The proposed algorithm is referred to as modified Sine Cosine Algorithm (m-SCA). In the proposed algorithm m-SCA, two strategies – opposition based learning (OBL) and self-adaptation of individual solutions are adopted to enhance the diversity of solution and to prevent the skipping of true solutions. The parameters used in this strategy (OBL and self-adaptation) are optimized to maintain a proper balance between exploration and exploitation mechanism in the algorithm. In the paper, to evaluate the performance of the proposed algorithm classical benchmark test set and the standard test set IEEE CEC 2014 have been taken. In the paper, four well-known engineering optimization problems are also used to evaluate the capability of proposed algorithm in obtaining the solution of real-life problems. Comparison with other state-of-the art

Algorithm 1
Classical Sine Cosine Algorithm (SCA).

1. Generate the initial, uniformly distributed random population of solutions
2. Evaluate the fitness of each individual solution
3. Initialize the algorithm parameters T (maximum number of iterations as termination criteria) and coefficient A.
4. Select the best solution x_{best} from the population of solutions
5. Initialize the iteration number $t = 0$
6. while $t < T$
7. **for** each individual solution
8. Update the state with the help of Eqs. (3) and (4)
9. Evaluate the fitness of updated solution
10. Update the best solution x_{best}
11. **end of for**
12. update the coefficient A
13. $t = t + 1$
14. end of whle
15. Return the best solution x_{best}

algorithms proves the efficacy and reliability of the proposed algorithm.

The remaining of the paper is organized as follows – Section 2 provides an overview of Sine Cosine Algorithm. In Section 3, some basic concepts are introduced. Section 4 presents a modified version of classical Sine Cosine Algorithm. In Section 5, the experimental setup has been established and analysis of the results on classical set of benchmark problems and standard IEEE CEC 2014 benchmark problems is presented. Sections 6 evaluates the proposed algorithms on five engineering optimization problems. Finally, Section 7 concludes the findings of the paper.

2. Overview of sine cosine algorithm (SCA)

The Sine Cosine Algorithm (SCA) is a recently developed meta-heuristic algorithm based on the mathematical features of sine and cosine functions. This algorithm was developed by Mirjalili in 2015 (Mirjalili, 2016). Similar to other population-based meta-heuristic algorithms, SCA also starts with a randomly distributed set of solutions. Then each individual solution is updated with the help of following equations –

$$x_{i,t+1} = x_{i,t} + A \sin(b) |C x_{best} - x_{i,t}| \quad (1)$$

$$x_{i,t+1} = x_{i,t} + A \cos(b) |C x_{best} - x_{i,t}| \quad (2)$$

The above two equations are used in SCA in a following manner

$$x_{i,t+1} = \begin{cases} x_{i,t} + A \sin(b) |C x_{best} - x_{i,t}| & if\ rand < 0.5 \\ x_{i,t} + A \cos(b) |C x_{best} - x_{i,t}| & otherwise \end{cases} \quad (3)$$

where $x_{i,t}$ and $x_{i,t+1}$ represents the ith individual solution at iteration t and $(t+1)$ respectively. x_{best} is the fittest solution within the set of solutions, *rand* is a uniformly distributed random number in the interval (0, 1). The parameter A is a random vector which is liable to decide the area of the search space around the current solution. This region of search space may lie inside x_{best} and $x_{i,t}$ or outside them. This parameter also helps in exploration and exploitation of a search space along with a proper balance between them. In the first half of the maximum number of iterations, coefficient A dedicated to the exploration and the second half of maximum number of iterations is devoted to the exploitation of an available search space. Mathematically the vector A can be defined as follows –

$$A = 2 - 2\left(\frac{t}{T}\right) \quad (4)$$

where T represents the maximum number of iterations which is predefined as a termination criteria for SCA. b is a vector that is used to decide the direction of moment of a current solution which can be either towards the x_{best} or outside x_{best}. The vector C provides a weight to x_{best} which emphasizes the exploration ($C > 1$) and exploitation ($C < 1$). The vector C also helps in avoiding the premature convergence at the end of iterations. The vector *rand* helps in transition from sine to cosine functions and vice versa. The range of sine and cosine functions with the parameter A is presented in Fig. 1. The step-wise description of SCA algorithm is provided in Algorithm 1.

3. Preliminaries

3.1. *Opposite number* (Tizhoosh, 2005)

Consider a point $x \in [a, b]$ and $a, b \in R$. Then opposite point of x denoted by x_{op} can be obtained as –

$$x_{op} = a + b - x$$

The definition of opposite numbers can be defined for higher dimension (Tizhoosh, 2005; Ventresca & Tizhoosh, 2008) as shown below.

3.2. *Opposite vector in d-space* (Tizhoosh, 2005)

In $d-$dimensional space the opposite point $x_{op} = (x_{op}{}^1, x_{op}{}^2, \ldots, x_{op}{}^d)$ of a point $x = (x^1, x^2, \ldots, x^d) \in R^d$ can be calculated as –

$$x_{op}{}^j = a_j + b_j + x^j$$

In Rahnamayan, Tizhoosh, and Salama (2008), it has been proved that finding the unknown optimal solution to the problem with a random direction along with its opposite direction provides a higher chance as compared to pure random direction. The integration of opposite numbers in search algorithm is fruitful when the current obtained solution is very far away from the optima especially when the optima is in opposite direction of a current solution.

The concept of opposite numbers is used by many researchers to enhance the ability to learn, search and optimize the meta-heuristic algorithms. The literature on OBL and its benefits in meta-heuristics can be found in Mahdavi, Rahnamayan, and Deb (2018).

4. Proposed modified Sine cosine algorithm (m-SCA)

In this section, the modified Sine Cosine Algorithm (SCA) is proposed which enhances the search capabilities of search agents (solutions) in SCA. From the experimental analysis, it has been observed that the classical SCA suffers from the problem of stagnation in local optima and this problem causes the premature convergence. In classical SCA, each solution uses the information of

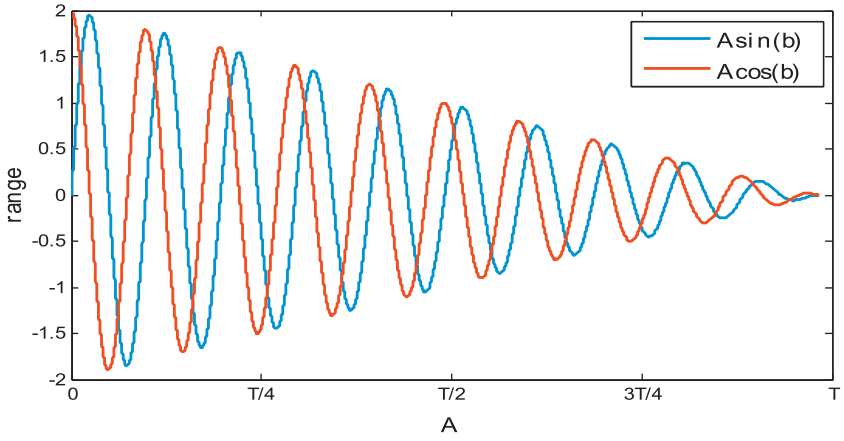

Fig. 1. The range of sine and cosine functions with parameter *A*.

best solution to update their states and if the best available solution gets trapped in local optimums then there may be a chance that the whole population of solutions will be trap in local optima due to the loss of diversity of solutions in search space. Also, in classical SCA, the problems of skipping of true solutions has been observed at the early iterations where the parameter $A > 1$. This situation mislead the search process and this misleading guidance causes the problem of stagnation in local solutions at the end of iterations where the algorithm faces the problem of insufficient diversity of solutions. To overcome these issues of classical Sine Cosine Algorithm, in the present paper, two strategies are integrated into SCA.

In the first strategy, the opposite population of solutions is generated according to the perturbation or jumping rate (J_R) by using opposite numbers as described in Section 3. In the proposed algorithm jumping rate is fixed at 0.1 to avoid the overflow of diversity because high jumping rate skips the true solutions of the problem and which misleads the search process in algorithm. Also, in this situation, the role of search equations of SCA will be deficient. This jumping rate helps to move out from local optimums and to provide new optimum direction. This new optimum direction may have a higher chance of locating global optima especially in that cases where the optima are just in opposite direction from current solution.

In the second strategy, a self-adaptive direction is integrated into the search equation of classical SCA. The proposed search equation based on the self-adaptation of solutions is given as follows –

$$v_{i,t+1} = x_{i,t+1} + S_R(x_{curr\_best} - x_{i,t}) \tag{5}$$

or

$$v_{i,t+1} = \begin{cases} x_{i,t} + \overbrace{A\sin(b)|Cx_{best} - x_{i,t}|}^{\text{Social Component}} + \overbrace{S_R(x_{curr\_best} - x_{i,t})}^{\text{Cognitive Component}} \\ x_{i,t} + \underbrace{A\cos(b)|Cx_{best} - x_{i,t}|}_{\text{Social Component}} + \underbrace{S_R(x_{curr\_best} - x_{i,t})}_{\text{Cognitive Component}} \end{cases} \quad \begin{matrix} if\ rand < 0.5 \\ otherwise \end{matrix} \tag{6}$$

where $x_{i,t+1}$ is the updated state of solution x_i at iteration $(t+1)$ using Eq. (3), x_{curr_best} is the best state obtained so far for a solution x_i, $v_{i,t+1}$ is the new updated state of solution x_i in proposed m-SCA. S_R is the perturbation or jumping rate. The self-adaptation based vector $(x_{curr\_best} - x_{i,t})$ helps in local search and to use the information of previously obtained best solution that are stored in the memory i.e. this term is used to exploit all the promising regions around the previously obtained best solutions. In Eq. (6) the second term on right hand side refers to the social component as it provides the direction towards the fittest solution of population. The third term refers to the cognitive component of the population of solutions as it provides the direction toward the individual best memory. Thus the social and cognitive component sustain the explorative and exploitative behavior together in the algorithm.

Thus the phase of opposition based learning and self-adaptive learning provides an enhanced global and local search which helps in increasing appropriate diversity and avoids the skipping of true solutions. The step-wise description of the proposed modified Sine Cosine Algorithm (m-SCA) is presented in Algorithm 2 and the flow of search process is presented in Fig. 2. A detailed analysis of enhanced diversity of solutions and exploitation of search space has been done in experimental section.

5. Experimental discussion

The proposed modified Sine Cosine Algorithm (m-SCA) can be considered as a standard improved version of classical Sine Cosine Algorithm (SCA). In the proposed m-SCA, accuracy in obtaining the solution of a global optimization problem is increased by enhancing the local and global search ability of solutions. To ensure the efficiency of the proposed algorithm on real-life optimization problems, it should be analyzed and tested first on some benchmark test problems.

Therefore, in this section, two different benchmark test set, classical benchmark of 23 problems and standard IEEE CEC 2014 have been considered. The details and analysis of proposed m-SCA on these test problems are as follows –

5.1. Experimental analysis on benchmark test set 1

In this section, a set of 23 well-known benchmark test problems is considered. This problem set contains seven unimodal test functions and sixteen multi-modal test functions. The problems are

Algorithm 2
Modified Sine Cosine Algorithm (m-SCA).

1. Generate the initial, uniformly distributed random population of solutions (P)
2. Evaluate the fitness of each individual solution
3. Initialize the algorithm parameters – N (size of population of solutions), T (maximum number of iterations as termination criteria), A, perturbation or jumping rate J_R and self-adaptation rate S_R
4. Select the best solution x_{best} from the population of solutions
5. Initialize the iteration number $t = 0$
6. while $t < T$
7. **if** rand $< J_R$
8. Calculate the opposite population of solutions (OP_S)
9. Evaluate the fitness of each solution of OP_S
10. Select the N best solutions from the set $P \cup OP_S$
11. Update the best solution x_{best}
12. ***else***
13. **for** each individual solution
14. Update the state with the help of Eqs. (4) to (6)
15. Evaluate the fitness of updated solution
16. Update the best solution x_{best}
17. **end of for**
18. **end of if**
19. update the coefficient a
20. $t = t + 1$
21. end of while
22. Return the best solution x_{best}.

denoted by letter F and are numbers as F1, F2,..., F23. The problems are provided in detail in Table 1. In this problem set, 13 problems (F1-F13) are scalable while 10 problems (F14-F23) are non-scalable. These problems are used by many researchers to evaluate their algorithms (Mirjalili, 2015a,b, 2016; Mirjalili & Lewis 2016; Mirjalili, Mirjalili, & Hatamlou, 2016; Salimi, 2015). For scalable problems dimension is fixed as 30. In the Table 1, the classification of these test problems is presented.

To compare the results obtained from proposed m-SCA with classical SCA, a population size of solutions is fixed as 30 and the termination criteria is taken as 500 iterations. Therefore the total number of function evaluations which are used in algorithms are $500 \times 30 = 15{,}000$, as in each iteration each solution is evaluated once. The details of all the parameters used in m-SCA are shown in Table 2. The obtained results on the considered test problems are presented in Table 3. In this table, various statistical measures like best, mean, median, worst and standard deviation of objective function value obtained in 30 runs are depicted.

The results provided in Table 3 clearly demonstrates the better performance of proposed algorithm m-SCA compared to classical SCA. In problems F1 to F7 only one optimum point is present and these problems are known as unimodal test problems. Unimodal test problems evaluate the strength of exploitation or local search. The problems from F8 to F23 are multi-modals test problems and in these problems more than one extreme point is present. These multi-modal test problems are used to appraise the exploration or global search and local optima avoidance ability of meta-heuristic optimization algorithms.

In Unimodal test problems, m-SCA outperforms classical SCA in all the test statistics. In all the multi-modal test problems (F8-F23), the proposed m-SCA algorithm performs better in terms of best objective function value. The average functional value obtained by m-SCA is better than classical SCA in all the problems except F9. The obtained worst objective function value is better in m-SCA than classical SCA for all the test functions except F9. Moreover, by analyzing the standard deviation value from the Table 3 it can be concluded that the modified Sine Cosine Algorithm is more reliable as compared to classical Sine Cosine Algorithm as in classical SCA the results are more deviated.

To analyze the enhanced diversity (exploration) of solutions and exploitation capability, average distance between the solutions in each generation is plotted in Figs. 3 and 4. The Euclidean distance $\|.\|$ between two solutions $X = (x_1, x_2, \ldots, x_d)$ and $Y = (y_1, y_2, \ldots, y_d)$ can be calculated as follows –

$$||X - Y||_2 = \sqrt{\sum_{j=1}^{d} \left(x_j - y_j\right)^2}$$

From the figures, it can be analyzed that the distance between solutions in m-SCA is larger as compared to classical SCA. This ensures the diversity of solution or exploration of promising search regions. Since the distance graph is more peregrinated in m-SCA which shows the exploitation of search regions which are already discovered and this shows the advantage of integrating self-adaptation in search equation. Exploration of a search space is also maintained by the oppositional population of solutions. Therefore, these graphs demonstrate the advantage of self-adaptation mechanism and opposition based learning that is incorporated in m-SCA.

5.2. Statistical Analysis between classical SCA and m-SCA

The statistical comparison of the obtained results is necessary to analyze the significant improvement. Therefore in this section, Wilcoxon signed rank test is used between the classical SCA and proposed m-SCA. The Wilcoxon test has been chosen for statistical comparison because the median value is more important to characterize the efficiency of algorithm and this test can be used without knowing the distribution of the dataset. In Table 4, the statistical comparison between classical SCA and proposed modified SCA (m-SCA) is done at 5% level of significance. In the table, the acquired $p-$values are also presented corresponding to all test problems. To represent that the proposed m-SCA is statistically significantly better than classical SCA, $'+'$ sign has been used and $'-'$ sign is used to represent that classical SCA is significantly better than proposed m-SCA. If both the algorithms are statistically same then, $'='$ sign is used.

From the Table 4, it can be observed that out of 23 benchmark test problems, in 22 test problems the proposed m-SCA significantly outperforms classical SCA while only in one problem the classical SCA is significantly better than m-SCA.

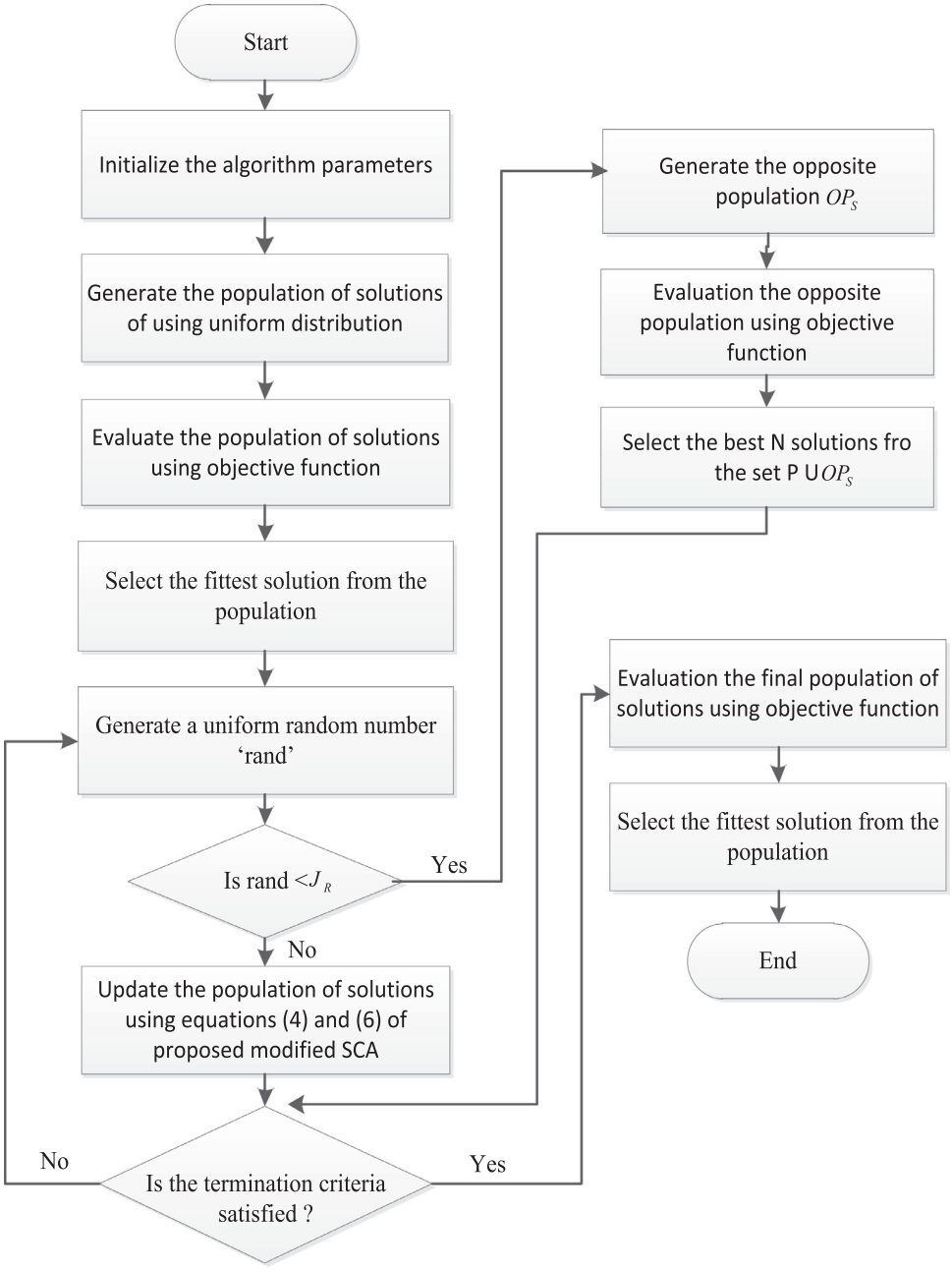

Fig. 2. Framework of modified Sine Cosine Algorithm (m-SCA).

5.3. Comparison of complexity between classical SCA and proposed m-SCA

Computational complexity of meta-heuristic algorithm is very crucial to determine the run time of algorithm. The complexity of the algorithm primarily depends on the structure of the algorithm. Therefore, the complexity of the proposed m-SCA depends on its population size of solutions, dimension of the decision variable and maximum number of iterations. Thus the complexity of the proposed algorithm in terms of big-O notation can be calculated from the pseudo code as follows:

O(m-SCA) = O(position update in opposition phase) + O(positions update through search equations of m-SCA) + O(selection of destination point)

- O(m-SCA) = $O(T \times (N \times D + N))) = O(TND + TN))$
- Similarly, O(SCA) = $O(TND + TN))$

where, T represents the maximum number of iterations, N is the number of solutions in the population and D represents the dimension of the decision variable. Therefore the complexity of the proposed m-SCA and classical SCA is same in terms of worst time complexity.

Thus, complexity wise both the algorithms are same. The CPU execution time to obtain the solution of test problems is presented in Table 5. In this table, the average execution time obtained in 30 runs is reported. From this table, it can be easily observed that m-SCA takes slightly higher time than classical SCA to obtain the solution of test problems.

5.4. Performance Index (PI) analysis

To compare the algorithms classical SCA and proposed m-SCA by giving weights to the error value, success rate and CPU comput-

Table 1
Classification of benchmark problems.

Test problems	Dimension	Range of search space	F_{min}
Unimodal benchmark problems			
$F1(x) = \sum_{i=1}^{d} x_i^2$	30	[−100, 100]	0
$F2(x) = \sum_{i=1}^{d} \lvert x_i \rvert + \prod_{i=1}^{d} \lvert x_i \rvert$	30	[−10, 10]	0
$F3(x) = \sum_{i=1}^{d} (\sum_{j-1}^{i} x_i)^2$	30	[−100, 100]	0
$F4(x) = max_i\{\lvert x_i \rvert, 1 \le i \le d\}$	30	[−100, 100]	0
$F5(x) = \sum_{i=1}^{d-1} [100\ (x_{i+1} - x_i^2)^2 + (x_i - 1)^2]$	30	[−30, 30]	0
$F6(x) = \sum_{i=1}^{d} ([x_i + 0.5])^2$	30	[−100, 100]	0
$F7(x) = \sum_{i=1}^{d} i \cdot x_i^4 + rand[0, 1)$	30	[−1.28, 1.28]	0
Multimodal benchmark problems			
$F8(x) = \sum_{i=1}^{d} -x_i sin(\sqrt{\lvert x_i \rvert})$	30	[−500, 500]	$-418.9829 \times d$
$F9(x) = \sum_{i=1}^{d} [x_i^2 - 10\ cos(2\pi x_i) + 10]$	30	[−5.12, 5.12]	0
$F10(x) = \sum_{i=1}^{d} -20\ exp(-0.2\sqrt{\frac{1}{d}\sum_{i=1}^{d} x_i^2}\ ) -$ $exp(\frac{1}{d}\sum_{i=1}^{d} cos(2\pi x_i)) + 20 + e$	30	[−32, 32]	0
$F11(x) = \frac{1}{4} \times 10^{-3} \sum_{i=1}^{d} x_i^2 - \prod_{i=1}^{d} cos(x_i/\sqrt{i}) + 1)$	30	[−600, 600]	0
$F12(x) =$ $\frac{\pi}{d}\{10\ sin(\pi\ y_1) + \sum_{i=1}^{d-1} (y_i - 1)^2[1 + 10sin^2(\ \pi y_{i+1})] + (y_n - 1)^2\} +$ $\sum_{i=1}^{d} u(x_i, 10,\ 100,\ 4)$ $y_i = \frac{x_i+5}{4}$ $u(x_i, a,\ k,\ m) = \begin{cases} k(x_i - a)^m & if\ x_i > a \\ k(-x_i - a)^m & if\ x_i < -a \\ 0 & otherwise \end{cases}$	30	[−50, 50]	0
$F13(x) = 0.1 \times$ $\{sin^2(3\ \pi x_1) + \sum_{i=1}^{d} (x_i - 1)^2[1 + sin^2(1 + 3\ \pi x_i)] + (x_d - 1)^2[1 + sin^2(2\ \pi x_n)]\} +$ $\sum_{i=1}^{d} u(x_i, 5,\ 100,\ 4)$	30	[−50, 50]	0
Fixed-dimension multimodal benchmark problems			
$F14(x) = [0.002 + \sum_{j=1}^{25} \frac{1}{j + \sum_{i=1}^{2} (x_i - x_{ij})^6}]^{-1}$	2	[−65, 65]	1
$F15(x) = \sum_{i=1}^{11} [a_i - \frac{x_1(b_i^2 + b_i\ x_2)}{b_i^2 + b_i\ x_3 + x_4}]^2$	4	[−5, 5]	0.0003
$F16(x) =$ $4x_1^2 - 2.1\ x_1^4 + 0.33\ x_1^6\ 10 + x_1x_2 - 4\ x_2^2 + 4x_2^4$	2	[−5, 5]	−1.0316
$F17(x) = 10 + 10 \times (1 - \frac{0.125}{\pi})cos\ (x_1) +$ $(x_2 - \frac{5.1}{4\ \pi^2}x_1^2 + \frac{5}{\pi}x_1 - 6)^2$	2	[−5, 5]	0.398
$F18(x) =$ $[1 + (1 + x_1 + x_2)^2(19 - 14\ x_1 + 3\ x_1^2 - 14\ x_2 + 6\ x_1\ x_2 + 3\ x_2^2)] \times$ $[30 + (2\ x_1 - 3\ x_2\ )^2 \times (18 - 32\ x_1 + 12x_1^2 + 48\ x_2 - 36\ x_1\ x_2 + 27\ x_2^2\ )]$	2	[−2, 2]	3
$F19(x) = -\sum_{i=1}^{4} c_i\ exp(-\sum_{j=1}^{3} a_{ij}(x_j - p_{ij})^2)$	3	[1,3]	−3.86
$F20(x) = -\sum_{i=1}^{4} c_i\ exp(-\sum_{j=1}^{6} a_{ij}(x_j - p_{ij})^2)$	6	[0,1]	−3.32
$F21(x) = -\sum_{i=1}^{5} [(x - a_i)(x - a_i)^T + c_i]^{-1}$	4	[0,10]	−10.1532
$F22(x) = -\sum_{i=1}^{7} [(x - a_i)(x - a_i)^T + c_i]^{-1}$	4	[0,10]	−10.4028
$F23(x) = -\sum_{i=1}^{10} [(x - a_i)(x - a_i)^T + c_i]^{-1}$	4	[0, 10]	−10.5363

ing time to obtain the solution of test problems, Performance Index is calculated (Deep & Thakur, 2007). The relative performance of an algorithm using PI can be calculated as follows –

$$PI = \frac{1}{n_p}\sum_{p=1}^{n_p} w_1\alpha_p + w_2\beta_p + w_3\delta_p$$

where n_p is the total number of problems, and

$$\alpha_p = \frac{ME^p}{AE^p} \quad \beta_p = \frac{SR^p}{TR^p}\ \delta_p = \frac{MT^p}{AT^p}$$

where,

ME^p – minimum of average error obtained by all the algorithms to obtain a solution of the problem p

AE^p – average error obtained by an algorithm to obtain a solution of the problem p

SR^p – number of successful runs while solving the problem p

TR^p – total number of runs for the problem p

MT^p – minimum of average execution time used by all the algorithms to obtain a solution of the problem p

Table 2
The detail of the parameter used in proposed m-SCA.

S. no.	Parameter	Parameter value
1	Population size of solutions	30
2	Maximum number of iterations	500
3	Maximum number of function evaluations	$30 \times 500 = 15000$
4	Perturbation or jumping rate	0.1
5	Self-adaptation rate	*rand*

AT^p – average execution time used by an algorithm to obtain a solution of the problem p,

w_1, w_2 and w_3 are the non-negative weights associated with error term α_p, success rate β_p and percentage of computational time δ_p such that $w_1 + w_2 + w_3 = 1$. In our study, three different cases of weights are considered:

Case I : $w_1 = w,\ w_2 = \frac{(1-w)}{2}$ and $w_3 = \frac{(1-w)}{2}$

Case II : $w_1 = \frac{(1-w)}{2},\ w_2 = w$ and $w_3 = \frac{(1-w)}{2}$

Case III : $w_1 = \frac{(1-w)}{2},\ w_2 = \frac{(1-w)}{2}$ and $w_3 = w$ where $0 \leq w \leq 1$.

The PI graphs corresponding to each case are plotted in Fig. 5. In these figures, weights w_1, w_2 and w_3 are depicted on horizontal axis and the corresponding calculated PI is shown on vertical axis.

In the case I, the execution time and success rate are given equal weights. It can be observed from the Fig. 5(a) that PI of the proposed algorithm m-SCA is higher than classical SCA. In case II, the error term and execution time are equally weighted while in Case III, the error term and success rate are equally weighted. From the Fig. 5(b) and (c) it can be analyzed that PI of m-SCA is higher than classical SCA except in case III, when higher preference ($w > 0.8$) is given to the execution time. Overall when the user's requirement are less error, less execution time and successful runs simultaneously then in that case proposed m-SCA outperforms classical SCA.

5.5. Comparison with other algorithms

The meta-heuristic optimization algorithms depend on some specific operators. Also, each of these algorithms is dependent on parameters and their performance can be increased or decreased by tuning the algorithm parameters. Therefore a comparison between different algorithms is a very crucial part. Thus in

Table 3
Best, Mean, Median, Worst and Standard Deviation (STD) of objective function values obtained in 30 runs on classical test problems from classical Sine Cosine Algorithm (SCA) and modified Sine Cosine Algorithm (m-SCA).

Test function	Algorithm	Best	Mean	Median	Worst	STD
F1	**SCA**	5.86E−03	1.87E+01	3.42E+00	2.33E+02	4.37E+01
	m-SCA	**1.36E−13**	**5.70E−03**	**8.34E−06**	**1.45E−01**	**2.63E−02**
F2	**SCA**	4.50E−04	1.79E−02	1.11E−02	1.36E−01	2.64E−02
	m-SCA	**5.03E−12**	**9.11E−04**	**3.90E−05**	**8.09E−03**	**1.90E−03**
F3	**SCA**	4.19E+02	8.84E+03	7.73E+03	2.21E+04	5.74E+03
	m-SCA	**1.43E+01**	**8.48E+02**	**7.31E+02**	**2.03E+03**	**5.49E+02**
F4	**SCA**	7.42E+00	3.49E+01	3.57E+01	5.50E+01	1.11E+01
	m-SCA	**5.53E−02**	**7.07E−01**	**4.65E−01**	**1.72E+00**	**5.37E−01**
F5	**SCA**	134.2994	45107.0386	8554.7258	581007.4457	108717.5745
	m-SCA	**28.4614**	**29.5658**	**28.9778**	**40.9241**	**2.4320**
F6	**SCA**	3.92E+00	2.20E+01	8.73E+00	1.74E+02	3.59E+01
	m-SCA	**3.19E−01**	**1.24E+00**	**1.19E+00**	**3.01E+00**	**5.12E−01**
F7	**SCA**	1.18E−02	1.57E−01	6.81E−02	1.35E+00	2.85E−01
	m-SCA	**8.35E−03**	**1.95E−02**	**1.79E−02**	**3.68E−02**	**6.87E−03**
F8	**SCA**	−4825.0251	−3847.7716	−3774.3958	−3307.0503	371.1573
	m-SCA	**−5024.2070**	**−4265.8691**	**−4259.2271**	**−3732.3237**	**288.8681**
F9	**SCA**	1.66E−03	**2.91E+01**	**2.38E+01**	**1.05E+02**	**2.86E+01**
	m-SCA	**1.63E−04**	7.81E+01	9.44E+01	1.57E+02	5.21E+01
F10	**SCA**	1.36E−02	1.83E+01	2.02E+01	2.03E+01	5.45E+00
	m-SCA	**6.88E−07**	**3.36E−03**	**4.51E−04**	**3.76E−02**	**8.06E−03**
F11	**SCA**	1.24E−02	9.38E−01	1.04E+00	3.10E+00	5.99E−01
	m-SCA	**8.75E−09**	**3.84E−02**	**2.65E−03**	**2.86E−01**	**7.15E−02**
F12	**SCA**	8.58E−01	3.93E+02	6.26E+00	1.08E+04	1.97E+03
	m-SCA	**3.57E−02**	**1.45E−01**	**1.41E−01**	**3.34E−01**	**8.19E−02**
F13	**SCA**	4.08E+00	8.79E+04	8.66E+01	1.18E+06	2.84E+05
	m-SCA	**7.59E−01**	**1.41E+00**	**1.40E+00**	**2.29E+00**	**3.94E−01**
F14	**SCA**	**0.9980**	1.6924	1.0034	2.9821	0.9387
	m-SCA	**0.9980**	**1.0311**	**0.9980**	**1.9920**	**0.1815**
F15	**SCA**	0.00034	0.00106	0.00093	0.00159	0.00039
	m-SCA	**0.00033**	**0.00051**	**0.00051**	**0.00070**	**0.00010**
F16	**SCA**	**−1.0316**	**−1.0316**	**−1.0316**	−1.0313	0.0001
	m-SCA	**−1.0316**	**−1.0316**	**−1.0316**	**−1.0316**	**0**
F17	**SCA**	**0.3979**	0.3991	0.3987	0.4018	0.0011
	m-SCA	**0.3979**	**0.3979**	**0.3979**	**0.3980**	**0**
F18	**SCA**	**3.0000**	**3.0000**	**3.0000**	**3.0002**	**0.0001**
	m-SCA	**3.0000**	**3.0000**	**3.0000**	**3.0002**	**0.0001**
F19	**SCA**	−3.8605	−3.8539	−3.8543	−3.8475	0.0025
	m-SCA	**−3.8628**	**−3.8626**	**−3.8627**	**−3.8621**	**0.0002**
F20	**SCA**	−3.2903	−2.8312	−2.9897	−1.4362	0.3628
	m-SCA	**−3.3214**	**−3.3166**	**−3.3180**	**−3.2977**	**0.0051**
F21	**SCA**	−5.5623	−2.6782	−2.5830	−0.4973	1.8582
	m-SCA	**−10.1322**	**−9.9485**	**−10.0100**	**−9.2290**	**0.1998**
F22	**SCA**	−6.3513	−3.1537	−3.3017	−0.5239	1.8821
	m-SCA	**−10.3770**	**−10.1947**	**−10.2706**	**−9.4788**	**0.2101**
F23	**SCA**	−7.9732	−3.7367	−4.3534	−0.9422	2.0363
	m-SCA	**−10.5252**	**−10.3605**	**−10.3663**	**−10.0842**	**0.1134**

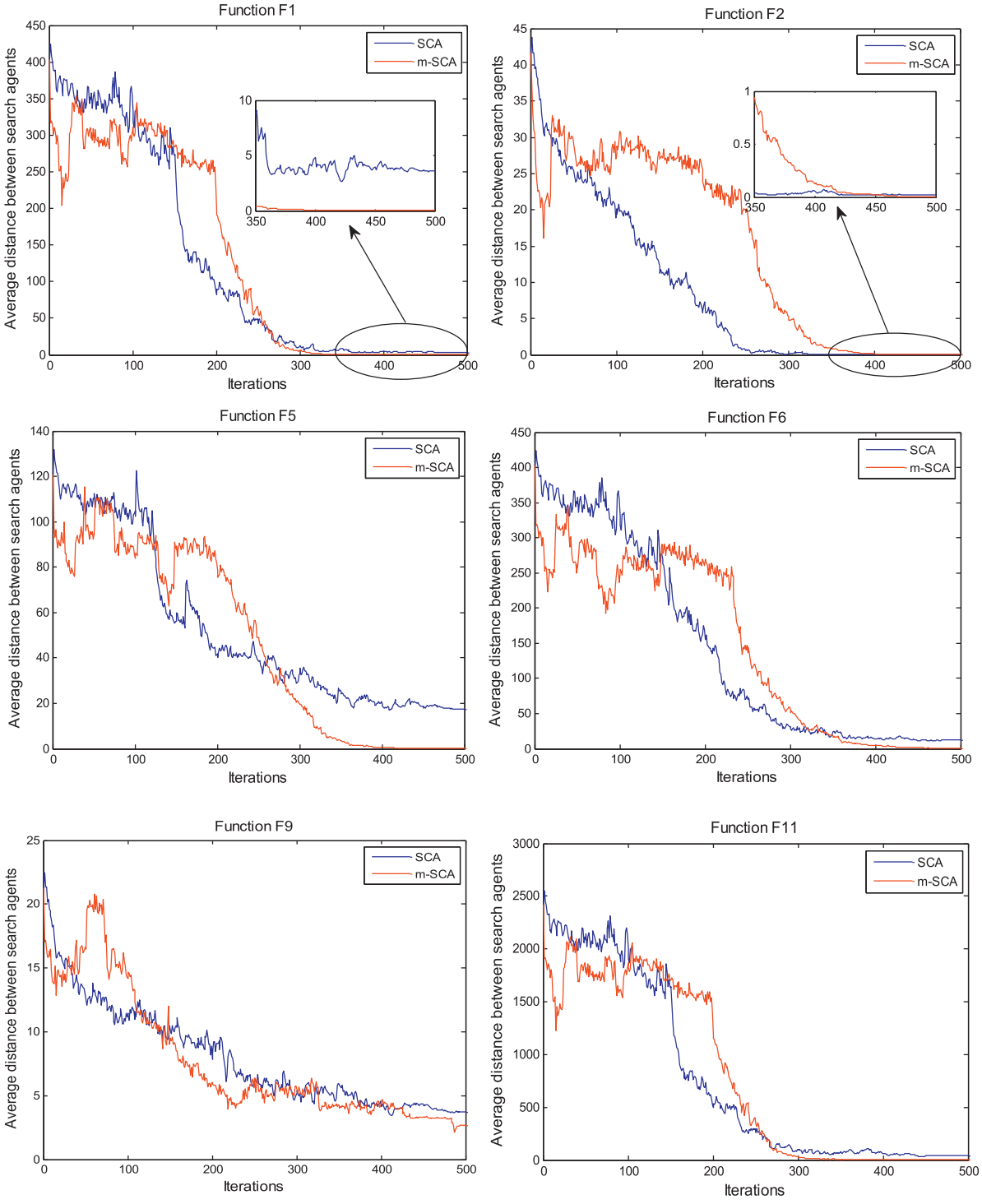

Fig. 3. Average distance between solution search agents in each iteration of classical SCA and proposed m-SCA algorithms.

Table 6 proposed m-SCA is compared with some state-of-the-art algorithms based on the average objective function value. In the table, the results have been reported for classical SCA, basic PSO (Eberhart & Kennedy, 1995), modified versions of PSO – m1PSO (Shi & Eberhart, 1998) m2PSO (Yang, Gao, Liu, & Song, 2015), Salp Swarm Algorithm (SSA) (Mirjalili et al., 2017), Moth-flame Optimization (MFO) algorithm (Mirjalili, 2015a), Harmony Search (HS) (Geem et al., 2001) and Artificial Bee Colony (ABC) (Karaboga & Basturk, 2007). In all the algorithms 500 iterations and 30 search agents have been considered for a fair comparison. From Table 6 it can be analyzed that the proposed algorithm performs better in most of the test problems. In fixed dimensional multimodal problems, the proposed algorithms outperforms other algorithms.

5.6. Convergence Analysis of m-SCA

In this section, the convergence graphs between proposed algorithm m-SCA, classical SCA and some other state-of-the-art algorithms as which are used for comparison in Section 5.5 are plotted in Figs. 6–8. In the convergence graphs the curves are plot-

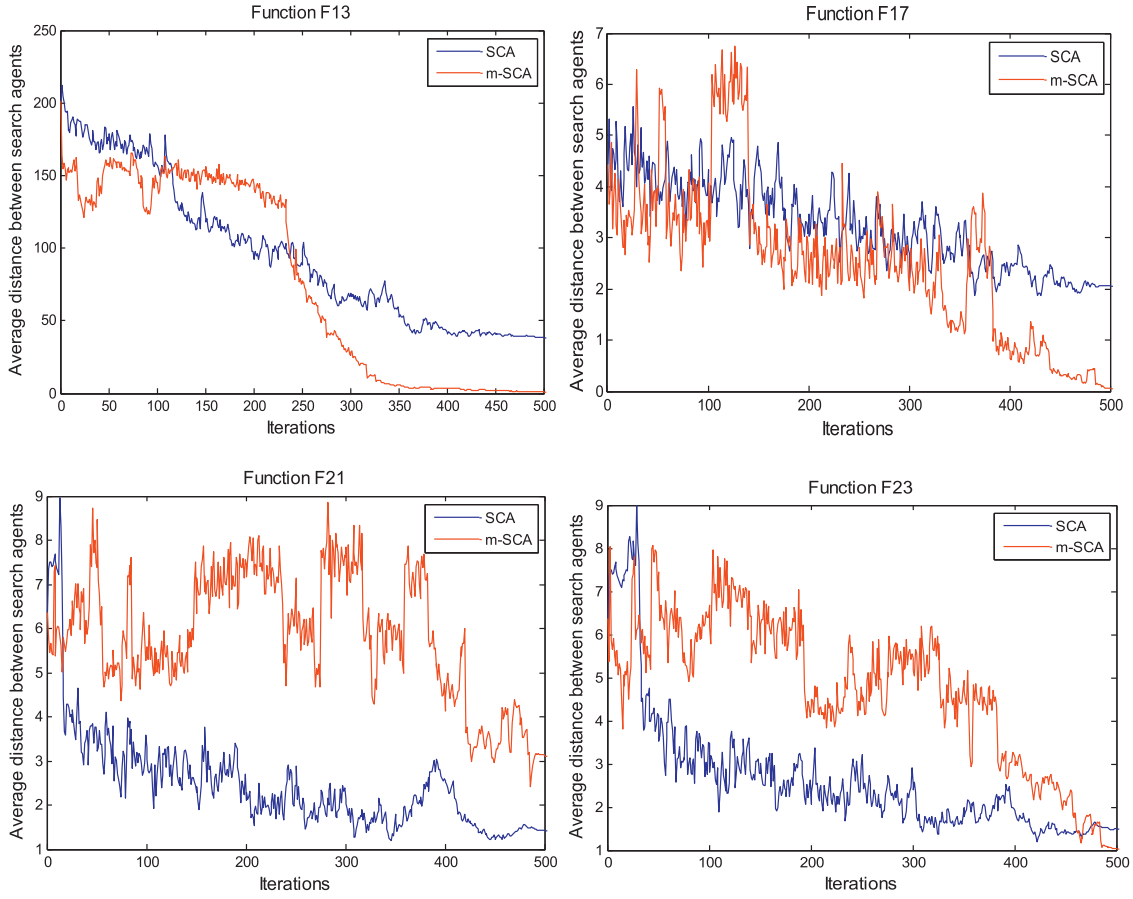

Fig. 4. Average distance between solution search agents in each iteration of classical SCA and proposed m-SCA algorithms.

Table 4
Statistical decision based on Wilcoxon Signed rank test at 5% level of significance on benchmark test set 1.

Test function	p – value	Decision	Test function	p – value	Decision
F1	1.734E−06	+	**F13**	1.734E−06	+
F2	1.639E−05	+	**F14**	5.026E−05	+
F3	2.127E−06	+	**F15**	8.466E−06	+
F4	1.734E−06	+	**F16**	1.562E−02	+
F5	1.734E−06	+	**F17**	3.790E−06	+
F6	1.734E−06	+	**F18**	5.076E−01	+
F7	7.691E−06	+	**F19**	1.712E−06	+
F8	1.742E−04	+	**F20**	1.734E−06	+
F9	4.195E−04	–	**F21**	1.734E−06	+
F10	1.734E−06	+	**F22**	1.734E−06	+
F11	2.879E−06	+	**F23**	1.734E−06	+
F12	1.734E−06	+			

Table 5
Execution time (in seconds) required to obtain the solution of test problems from classical SCA and proposed m-SCA for benchmark test set 1.

Test function	Execution time (in seconds)		Test function	Execution time (in seconds)	
	SCA	m-SCA		SCA	m-SCA
F1	0.1132	0.1225	**F13**	0.3816	0.3893
F2	0.1277	0.1336	**F14**	0.8366	0.8258
F3	0.7162	0.6991	**F15**	0.1062	0.1117
F4	0.1435	0.1489	**F16**	0.0590	0.0642
F5	0.1638	0.1657	**F17**	0.0539	0.0578
F6	0.1453	0.1545	**F18**	0.0534	0.0567
F7	0.2199	0.2254	**F19**	0.1476	0.1518
F8	0.1732	0.1836	**F20**	0.1539	0.1651
F9	0.1658	0.1807	**F21**	0.1810	0.1888
F10	0.1869	0.1908	**F22**	0.2224	0.2279
F11	0.1928	0.2033	**F23**	0.2820	0.2907
F12	0.4149	0.4230			

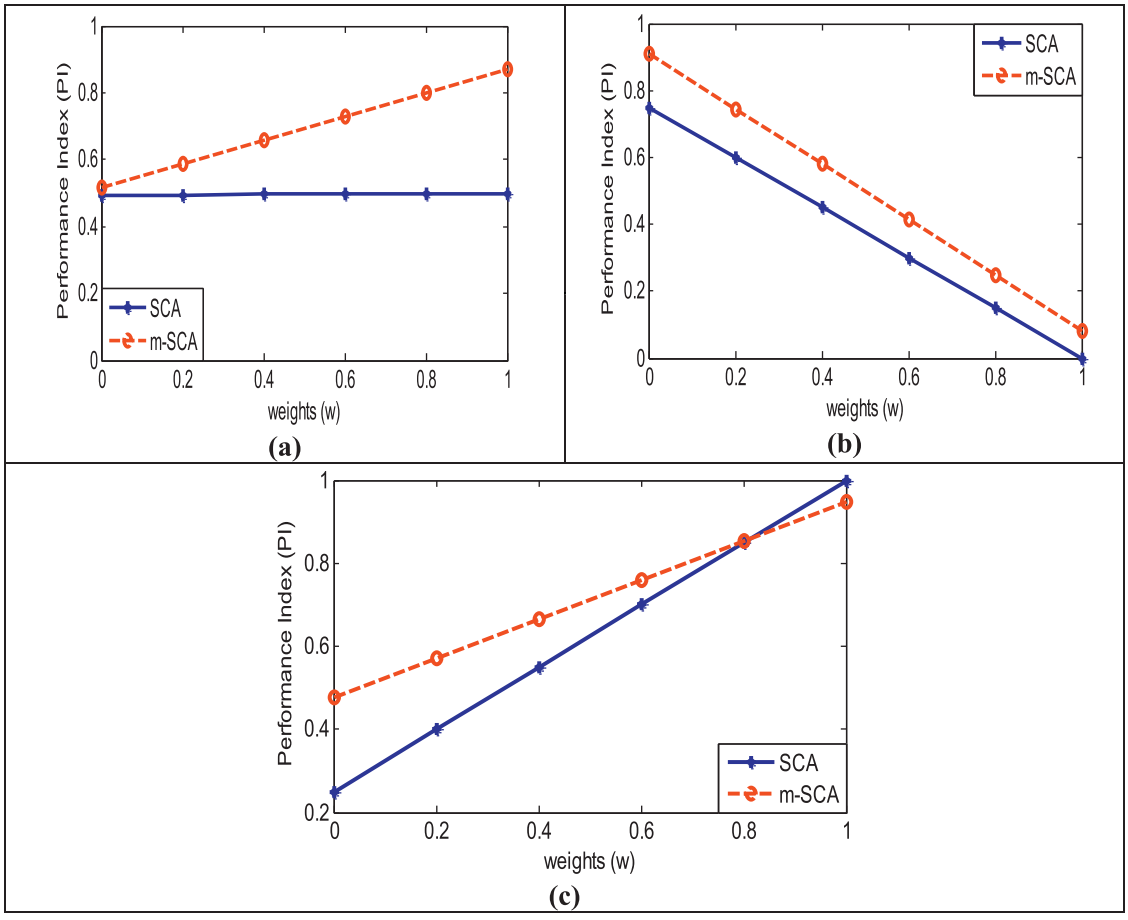

Fig. 5. Performance Index for benchmark test problems (a) for Case I, (b) for case II and (c) for case III.

Table 6
Performance comparison of proposed m-SCA with other meta-heuristic algorithms.

Test problem	SCA	basic PSO	m1PSO	m2PSO	SSA	MFO	HS	ABC	m-SCA
F1	1.87E+01	6.13E+03	7.38E+03	1.26E−05	2.09E−07	3.35E+03	6.63E+04	8.06E−05	5.70E−03
F2	1.79E−02	3.55E+01	4.00E+01	1.50E−04	2.13E+00	4.02E+01	1.61E+12	5.42E−03	9.11E−04
F3	8.84E+03	1.53E+04	1.77E+04	7.45E+02	1.58E+03	2.02E+04	1.04E+05	1.73E+04	8.48E+02
F4	3.49E+01	3.12E+01	3.38E+01	5.93E+00	1.17E+01	7.07E+01	8.52E+01	4.06E+01	7.07E−01
F5	4.51E+04	2.48E+06	3.50E+06	1.84E+02	2.15E+02	1.07E+07	2.42E+08	4.65E+01	2.96E+01
F6	2.20E+01	6.34E+03	7.40E+03	1.07E−05	1.90E−07	1.69E+03	6.64E+04	8.26E−05	1.24E+00
F7	1.57E−01	1.65E+00	2.09E+00	3.92E−02	1.79E−01	5.15E+00	1.14E+02	2.93E−01	1.95E−02
F8	−3.85E+03	−4.81E+03	−4.60E+03	−7.53E+03	−7.47E+03	−8.49E+03	−2.78E+03	−1.16E+04	−4.27E+03
F9	2.91E+01	2.25E+02	2.39E+02	4.31E+01	5.74E+01	1.68E+02	4.27E+02	4.73E+00	7.81E+01
F10	1.83E+01	1.36E+01	1.40E+01	2.68E−01	2.76E+00	1.68E+01	2.07E+01	1.28E−01	3.36E−03
F11	9.38E−01	5.62E+01	6.74E+01	1.06E−−02	2.15E−02	3.23E+01	5.97E+02	1.74E−02	3.84E−02
F12	3.93E+02	2.30E+05	4.38E+05	1.43E−01	7.77E+00	9.69E+00	5.44E+08	2.35E+02	1.45E−01
F13	8.79E+04	3.19E+06	5.64E+06	1.00E−02	1.66E+01	3.97E+01	1.05E+09	2.38E−04	1.41E+00
F14	1.69E+00	9.98E−01	9.98E−01	1.56E+00	1.16E+00	2.54E+00	3.93E+01	9.98E−01	1.03E+00
F15	1.06E−03	4.97E−03	2.98E−03	4.44E−03	3.49E−03	1.21E−03	5.07E−02	8.65E−04	5.14E−04
F16	−1.03E+00	−1.03E+00	−1.03E+00	−1.03E+00	−1.03E+00	−1.03E+00	−3.26E−01	−1.03E+00	−1.03E+00
F17	3.99E−01	3.98E−01	3.98E−01	3.98E−01	3.98E−01	3.98E−01	1.04E+00	3.98E−01	3.98E−01
F18	3.00E+00	3.01E+00	3.01E+00	3.00E+00	3.00E+00	3.00E+00	1.39E+01	3.00E+00	3.00E+00
F19	−3.85E+00	−3.86E+00	−3.86E+00	−3.86E+00	−3.86E+00	−3.86E+00	−3.57E+00	−3.00E−01	−3.86E+00
F20	−2.83E+00	−3.27E+00	−3.27E+00	−3.30E+00	−3.21E+00	−3.22E+00	−2.00E+00	−3.32E+00	−3.32E+00
F21	−2.68E+00	−7.06E+00	−6.23E+00	−5.32E+00	−7.56E+00	−6.82E+00	−8.92E−01	−9.77E+00	−9.95E+00
F22	−3.15E+00	−7.08E+00	−7.46E+00	−6.69E+00	−8.11E+00	−7.90E+00	−1.11E+00	−1.04E+01	−1.02E+01
F23	−3.74E+00	−7.84E+00	−8.18E+00	−7.20E+00	−8.91E+00	−7.65E+00	−1.38E+00	−1.05E+01	−1.04E+01

ted corresponding to the median value of the best objective function obtained in 30 runs. In these graphs, iterations are depicted on a horizontal axis and the objective fitness value is shown on a vertical axis. On the horizontal axis, equally space interval of iterations has been taken. From these figures, it can be observed that the convergence in modified Sine Cosine Algorithm (m-SCA) is better as compared to classical SCA. From the figure it can be observed also that the proposed algorithm shows very promising results in terms of convergence rate as compared to other algorithms.

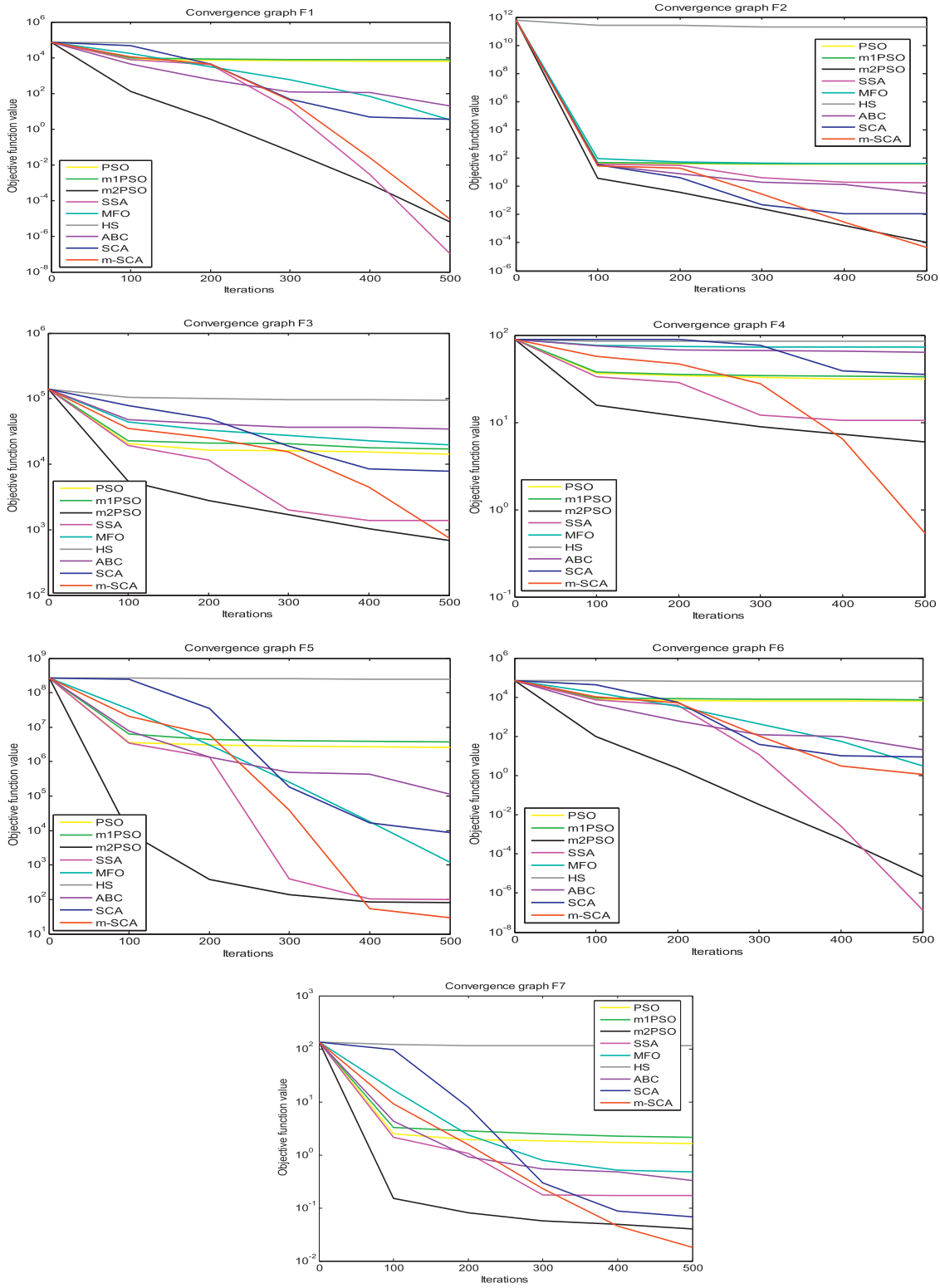

Fig. 6. Convergence graphs for unimodal problems (F1 to F7).

5.7. Experimental analysis on benchmark test set 2

In this section, standard benchmark test suite IEEE CEC 2014 (Liang, Qu, & Suganthan, 2013) has been considered to evaluate the performance of proposed algorithm. This test suite consists of 30 test problems of different complexity levels – Unimodal (F1-F3), Simple multimodal (F4-F16), hybrid (F17-F22) and composite functions (F23-F30). The test problems with dimension 10 and 30 has been considered and the search space is fixed as $[-100, 100]$. As per the guidelines of CEC 2014, the termination criteria is set up $(10^4 \times D)$ number of function evaluations, where D is the dimension of the problem. For all the test problems the population size of solutions is taken as 30 which is same as in original paper (Mirjalili, 2016).

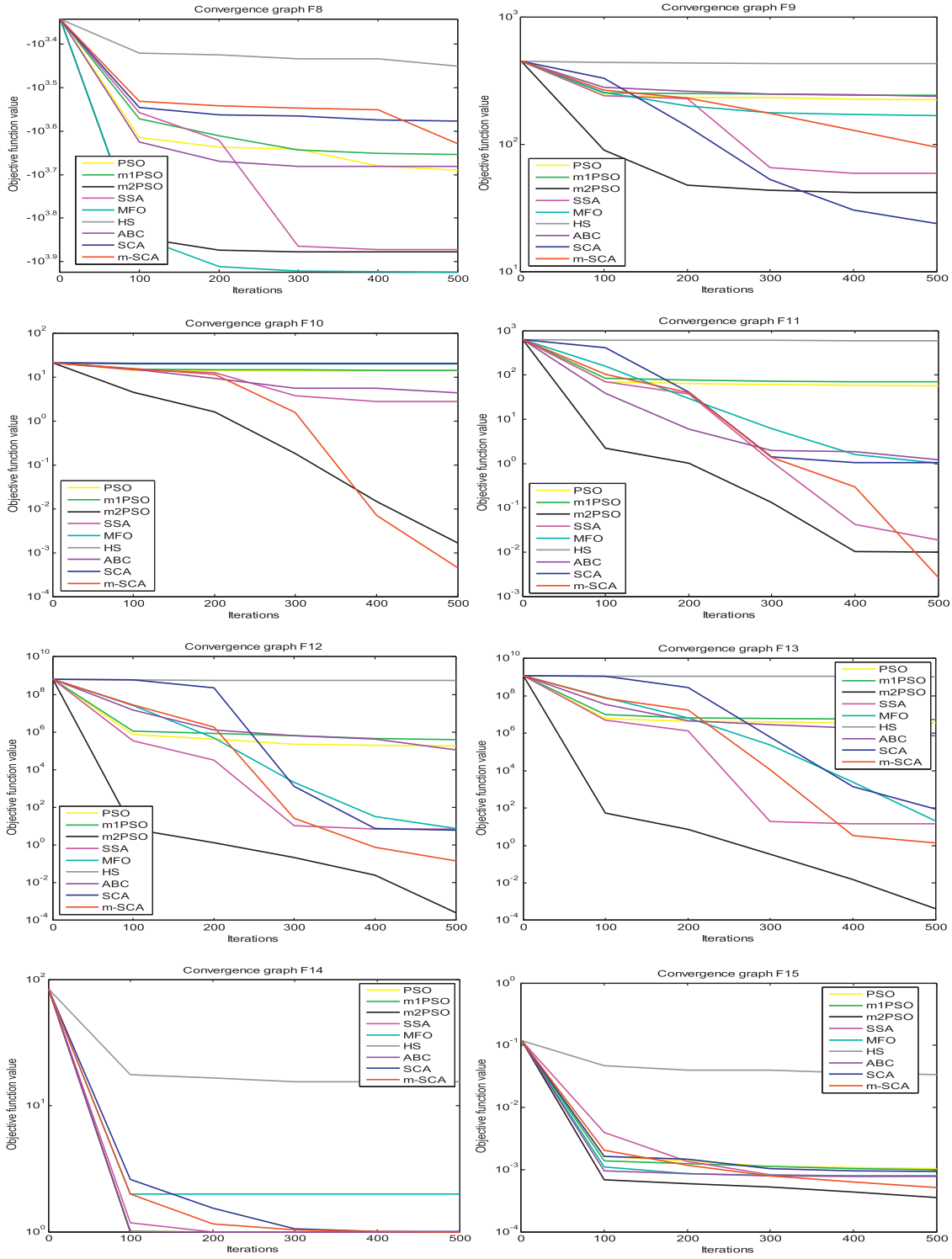

Fig. 7. Convergence graphs for multimodal problems.

The details of the results with different statistics – minimum (Best), average (Mean), Median, maximum (Worst) and standard deviation (STD) of the absolution error is presented in Tables 7 and 8 corresponding to dimensions 10 and 30. From the table it can be analyzed that on unimodal test problems the proposed m-SCA outperforms classical SCA in all the statistics. Thus in terms of exploiting the promising search regions, proposed m-SCA is better than classical SCA as the unimodal test problems evaluate the exploitation capability of search algorithms. On multimodal test problems (F4-F16), proposed m-SCA provides better results as compared to classical SCA in all measures like minimum, average, median and maximum value of the absolute error in fitness value and since multimodal test problems examine the exploration strength of meta-heuristic search algorithms. Thus the proposed m-SCA has enhanced the exploration capability of search agents to discover more promising regions of search space.

As the hybrid and composite problems are defined with the hybridization and composition of different unimodal and multimodal

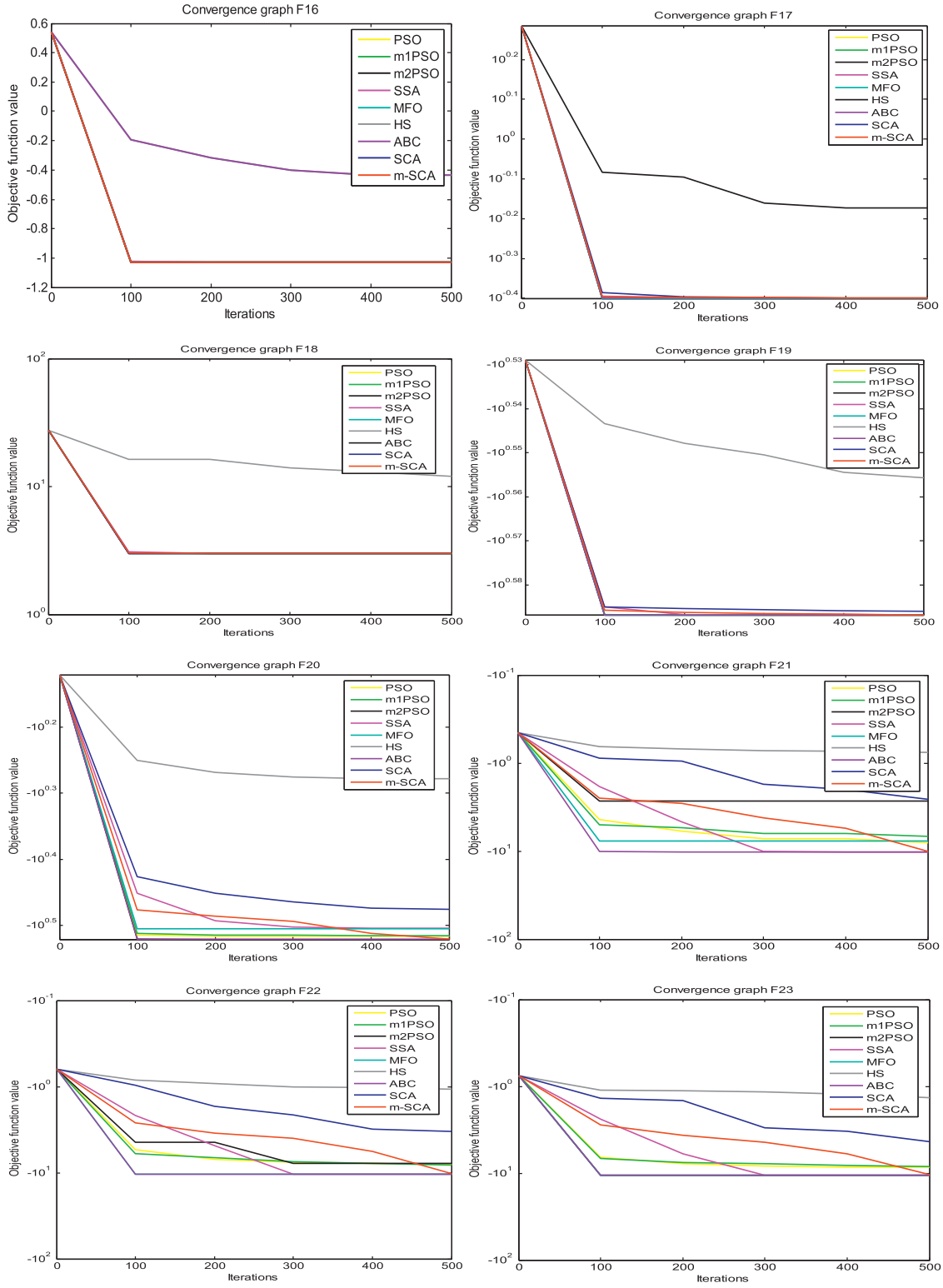

Fig. 8. Convergence graphs for multimodal problems.

problems. Therefore, these problems are used for the evaluation of local optima avoidance ability and measures the amount of balance between exploration and exploitation. Proposed m-SCA outperforms all these test problems in all the statistics but for the problems F23 and F25 the results in m-SCA are more deviated from mean as compared to classical SCA.

Therefore, the results on test problems of CEC 2014, ensure the better performance of proposed m-SCA as compared to classical SCA in all the test problems corresponding to all the statistics except the standard deviation in problem F5, F23 and F25. The significance of improvement in proposed m-SCA can be analyzed statistically and is provided in Tables 9 and 10 corresponding to the dimension 10 and 30 respectively. The statistical test

Table 7
Comparison of results between classical SCA and proposed m-SCA on IEEE CEC 2014 benchmark test problems with dimension 10.

Test function	Algorithm	Best	Mean	Median	Worst	STD
F1	**SCA**	5.459E+06	2.059E+07	1.537E+07	8.095E+07	1.417E+07
	m-SCA	**3.568E+05**	**1.798E+06**	**1.656E+06**	**4.281E+06**	**1.027E+06**
F2	**SCA**	3.541E+08	3.013E+09	2.249E+09	8.897E+09	2.150E+09
	m-SCA	**5.634E+04**	**2.167E+05**	**2.036E+05**	**4.232E+05**	**8.914E+04**
F3	**SCA**	1.965E+03	2.121E+04	1.521E+04	7.258E+04	1.794E+04
	m-SCA	**6.388E+02**	**1.850E+03**	**1.384E+03**	**6.089E+03**	**1.230E+03**
F4	**SCA**	6.395E+01	3.875E+02	2.418E+02	1.548E+03	3.310E+02
	m-SCA	**1.042E+00**	**2.052E+01**	**1.834E+01**	**3.854E+01**	**1.174E+01**
F5	**SCA**	2.020E+01	2.041E+01	2.039E+01	2.065E+01	**1.032E−01**
	m-SCA	**4.916E+00**	**1.942E+01**	**2.030E+01**	**2.043E+01**	2.916E+00
F6	**SCA**	6.053E+00	9.759E+00	1.001E+01	1.146E+01	1.220E+00
	m-SCA	**6.886E−01**	**1.396E+00**	**1.278E+00**	**3.017E+00**	**5.692E−01**
F7	**SCA**	6.657E+00	6.615E+01	7.577E+01	1.188E+02	3.186E+01
	m-SCA	**4.275E−01**	**6.979E−01**	**7.083E−01**	**9.059E−01**	**9.636E−02**
F8	**SCA**	4.273E+01	7.405E+01	7.105E+01	1.089E+02	1.583E+01
	m-SCA	**4.219E+00**	**1.020E+01**	**9.684E+00**	**1.755E+01**	**2.666E+00**
F9	**SCA**	2.603E+01	7.949E+01	8.361E+01	1.181E+02	2.111E+01
	m-SCA	**6.659E+00**	**1.118E+01**	**1.131E+01**	**1.783E+01**	**2.463E+00**
F10	**SCA**	6.907E+02	1.317E+03	1.327E+03	1.762E+03	2.208E+02
	m-SCA	**9.268E+01**	**3.082E+02**	**2.933E+02**	**6.497E+02**	**1.354E+02**
F11	**SCA**	1.197E+03	1.641E+03	1.719E+03	2.021E+03	2.516E+02
	m-SCA	**1.366E+02**	**4.323E+02**	**4.409E+02**	**7.623E+02**	**1.446E+02**
F12	**SCA**	8.000E−01	1.529E+00	1.447E+00	2.648E+00	4.368E−01
	m-SCA	**2.622E−01**	**6.466E−01**	**6.171E−01**	**1.037E+00**	**1.743E−01**
F13	**SCA**	5.736E−01	2.804E+00	2.848E+00	4.811E+00	1.245E+00
	m-SCA	**1.527E−01**	**2.428E−01**	**2.362E−01**	**4.367E−01**	**5.347E−02**
F14	**SCA**	6.997E−01	1.447E+01	1.480E+01	2.767E+01	7.299E+00
	m-SCA	**1.189E−01**	**2.374E−01**	**2.352E−01**	**3.628E−01**	**6.216E−02**
F15	**SCA**	6.488E+00	1.111E+04	7.919E+03	4.674E+04	1.230E+04
	m-SCA	**1.021E+00**	**1.777E+00**	**1.732E+00**	**2.767E+00**	**3.889E−01**
F16	**SCA**	2.916E+00	3.903E+00	3.982E+00	4.368E+00	3.164E−01
	m-SCA	**1.592E+00**	**2.532E+00**	**2.578E+00**	**2.988E+00**	**2.771E−01**
F17	**SCA**	3.824E+03	3.436E+05	1.709E+05	1.884E+06	3.965E+05
	m-SCA	**8.839E+02**	**2.145E+03**	**1.818E+03**	**8.178E+03**	**1.166E+03**
F18	**SCA**	8.457E+03	2.810E+06	5.832E+05	3.874E+07	6.492E+06
	m-SCA	**2.922E+02**	**2.374E+03**	**1.701E+03**	**7.323E+03**	**1.880E+03**
F19	**SCA**	6.497E+00	1.125E+01	1.105E+01	2.267E+01	3.549E+00
	m-SCA	**1.561E+00**	**2.020E+00**	**2.004E+00**	**2.513E+00**	**1.833E−01**
F20	**SCA**	3.781E+02	1.610E+05	5.155E+04	1.102E+06	2.421E+05
	m-SCA	**9.705E+01**	**6.544E+02**	**4.096E+02**	**2.719E+03**	**5.574E+02**
F21	**SCA**	1.888E+03	1.227E+05	5.032E+04	8.773E+05	1.898E+05
	m-SCA	**5.239E+02**	**1.466E+03**	**1.299E+03**	**3.917E+03**	**6.089E+02**
F22	**SCA**	5.395E+01	2.265E+02	2.323E+02	5.035E+02	1.115E+02
	m-SCA	**2.510E+01**	**3.037E+01**	**2.955E+01**	**4.125E+01**	**3.745E+00**
F23	**SCA**	3.400E+02	4.319E+02	4.301E+02	5.361E+02	**4.622E+01**
	m-SCA	**1.349E+01**	**3.096E+02**	**3.295E+02**	**3.301E+02**	7.003E+01
F24	**SCA**	1.476E+02	2.015E+02	2.028E+02	2.374E+02	2.430E+01
	m-SCA	**1.082E+02**	**1.156E+02**	**1.152E+02**	**1.230E+02**	**2.972E+00**
F25	**SCA**	1.696E+02	2.064E+02	2.074E+02	2.220E+02	**7.826E+00**
	m-SCA	**1.271E+02**	**1.510E+02**	**1.518E+02**	**1.830E+02**	1.324E+01
F26	**SCA**	1.005E+02	1.025E+02	1.024E+02	1.038E+02	8.564E−01
	m-SCA	**1.001E+02**	**1.002E+02**	**1.002E+02**	**1.003E+02**	**3.340E−02**
F27	**SCA**	1.231E+02	4.449E+02	4.608E+02	6.110E+02	1.160E+02
	m-SCA	**2.841E+00**	**4.575E+00**	**4.462E+00**	**8.516E+00**	**1.188E+00**
F28	**SCA**	4.139E+02	5.485E+02	5.487E+02	7.389E+02	7.743E+01
	m-SCA	**2.216E+02**	**4.154E+02**	**4.349E+02**	**5.316E+02**	**7.275E+01**
F29	**SCA**	5.593E+03	1.518E+05	4.721E+04	1.832E+06	3.139E+05
	m-SCA	**2.774E+02**	**5.040E+02**	**4.597E+02**	**1.026E+03**	**1.611E+02**
F30	**SCA**	1.290E+03	8.130E+03	4.876E+03	4.709E+04	9.535E+03
	m-SCA	**1.181E+03**	**1.678E+03**	**1.679E+03**	**2.267E+03**	**2.411E+02**

Wilcoxon signed rank is used for statistical analysis of results. The obtained $p-$values by applying Wilcoxon test are also reported in the same table. From this table, it can be concluded that the proposed m-SCA outperforms classical SCA in all the test problems as the Wilcoxon signed rank test favors the proposed m-SCA.

The comparison with other state-of-the art algorithms on CEC 2014 benchmark test problems is presented in Table 11. In these Tables results are compared with classical SCA, basic PSO (Eberhart, & Kennedy, 1995), modified-PSO – m1-PSO (Shi & Eberhart, 1998), m2-PSO (Yang et al., 2015), Artificial Bee Colony (ABC) (Karaboga & Basturk, 2007), Harmony Search (HS) algorithm (Assad & Deep, 2018; Geem et al., 2001), Salp Swarm Algorithm (SSA) (Mirjalili et al., 2017) and Moth-flame Optimization (MFO) (Mirjalili, 2015a). The comparison of results confirmed the competitive performance of proposed modified sine cosine algorithm with other algorithms.

Table 8
Comparison of results between classical SCA and proposed m-SCA on IEEE CEC 2014 benchmark test problems with dimension 30.

Test function	Algorithm	Best	Mean	Median	Worst	STD
F1	**SCA**	1.586E+08	6.275E+08	6.354E+08	1.845E+09	3.005E+08
	m-SCA	**9.863E+06**	**2.258E+07**	**2.230E+07**	**4.601E+07**	**6.353E+06**
F2	**SCA**	1.780E+10	5.010E+10	4.803E+10	1.270E+11	2.402E+10
	m-SCA	**2.398E+07**	**8.109E+07**	**5.705E+07**	**2.692E+08**	**5.590E+07**
F3	**SCA**	3.155E+04	1.079E+05	1.003E+05	2.865E+05	6.483E+04
	m-SCA	**1.598E+04**	**2.587E+04**	**2.533E+04**	**3.941E+04**	**6.433E+03**
F4	**SCA**	9.362E+02	9.413E+03	7.112E+03	2.335E+04	7.025E+03
	m-SCA	**1.452E+02**	**1.834E+02**	**1.791E+02**	**2.576E+02**	**2.583E+01**
F5	**SCA**	2.077E+01	2.098E+01	2.098E+01	2.111E+01	**5.704E−02**
	m-SCA	**2.062E+01**	**2.090E+01**	**2.092E+01**	**2.099E+01**	7.070E−02
F6	**SCA**	3.219E+01	4.025E+01	4.071E+01	4.454E+01	**2.739E+00**
	m-SCA	**9.390E+00**	**1.464E+01**	**1.475E+01**	**2.209E+01**	3.352E+00
F7	**SCA**	1.383E+02	6.639E+02	7.208E+02	1.056E+03	2.626E+02
	m-SCA	**1.239E+00**	**2.007E+00**	**1.958E+00**	**3.110E+00**	**4.617E−01**
F8	**SCA**	2.393E+02	3.877E+02	4.061E+02	4.892E+02	8.038E+01
	m-SCA	**7.744E+01**	**1.096E+02**	**1.112E+02**	**1.328E+02**	**1.240E+01**
F9	**SCA**	2.519E+02	4.528E+02	5.072E+02	5.926E+02	1.106E+02
	m-SCA	**1.074E+02**	**1.308E+02**	**1.306E+02**	**1.500E+02**	**9.809E+00**
F10	**SCA**	5.716E+03	7.115E+03	7.197E+03	7.921E+03	5.040E+02
	m-SCA	**2.210E+03**	**3.580E+03**	**3.645E+03**	**4.565E+03**	**4.722E+02**
F11	**SCA**	6.438E+03	7.634E+03	7.696E+03	8.645E+03	4.992E+02
	m-SCA	**3.759E+03**	**4.934E+03**	**5.040E+03**	**5.689E+03**	**4.707E+02**
F12	**SCA**	1.911E+00	2.945E+00	2.909E+00	3.569E+00	4.072E−01
	m-SCA	**1.062E+00**	**1.756E+00**	**1.780E+00**	**2.469E+00**	**2.886E−01**
F13	**SCA**	3.239E+00	7.604E+00	8.139E+00	1.068E+01	2.014E+00
	m-SCA	**1.999E-01**	**3.856E-01**	**3.881E-01**	**4.832E-01**	**5.918E−02**
F14	**SCA**	4.346E+01	2.512E+02	2.592E+02	4.086E+02	9.466E+01
	m-SCA	**1.923E-01**	**2.649E-01**	**2.625E-01**	**3.328E-01**	**2.939E−02**
F15	**SCA**	1.711E+03	5.709E+06	6.027E+06	1.208E+07	4.367E+06
	m-SCA	**1.139E+01**	**1.512E+01**	**1.522E+01**	**1.811E+01**	**1.470E+00**
F16	**SCA**	1.267E+01	1.346E+01	1.352E+01	1.387E+01	**2.711E−01**
	m-SCA	**1.137E+01**	**1.203E+01**	**1.207E+01**	**1.265E+01**	2.893E−01
F17	**SCA**	2.864E+06	4.382E+07	4.386E+07	9.857E+07	2.598E+07
	m-SCA	**1.628E+05**	**5.993E+05**	**5.196E+05**	**1.661E+06**	**3.592E+05**
F18	**SCA**	1.042E+08	1.931E+09	1.891E+09	4.996E+09	1.039E+09
	m-SCA	**3.935E+04**	**1.611E+05**	**1.476E+05**	**4.807E+05**	**8.252E+04**
F19	**SCA**	9.053E+01	4.406E+02	4.638E+02	8.194E+02	1.760E+02
	m-SCA	**1.376E+01**	**1.934E+01**	**1.862E+01**	**4.844E+01**	**5.953E+00**
F20	**SCA**	1.539E+04	5.276E+05	2.505E+05	3.091E+06	6.879E+05
	m-SCA	**5.035E+03**	**1.217E+04**	**1.202E+04**	**1.868E+04**	**3.703E+03**
F21	**SCA**	8.689E+05	1.456E+07	1.156E+07	4.463E+07	1.042E+07
	m-SCA	**3.062E+04**	**1.096E+05**	**9.824E+04**	**3.322E+05**	**5.655E+04**
F22	**SCA**	8.854E+02	1.776E+03	1.781E+03	2.635E+03	3.934E+02
	m-SCA	**1.741E+02**	**2.570E+02**	**2.485E+02**	**3.803E+02**	**5.706E+01**
F23	**SCA**	3.786E+02	1.075E+03	1.209E+03	1.807E+03	3.988E+02
	m-SCA	**3.187E+02**	**3.211E+02**	**3.209E+02**	**3.249E+02**	**1.584E+00**
F24	**SCA**	2.019E+02	4.580E+02	5.127E+02	5.523E+02	1.107E+02
	m-SCA	**2.001E+02**	**2.001E+02**	**2.001E+02**	**2.002E+02**	**4.292E−02**
F25	**SCA**	2.210E+02	2.939E+02	2.950E+02	3.566E+02	3.228E+01
	m-SCA	**2.000E+02**	**2.010E+02**	**2.000E+02**	**2.133E+02**	**3.194E+00**
F26	**SCA**	1.031E+02	1.075E+02	1.078E+02	1.104E+02	1.622E+00
	m-SCA	**1.003E+02**	**1.004E+02**	**1.004E+02**	**1.005E+02**	**5.401E−02**
F27	**SCA**	5.604E+02	1.068E+03	1.111E+03	1.243E+03	1.826E+02
	m-SCA	**4.096E+02**	**4.328E+02**	**4.242E+02**	**4.953E+02**	**2.008E+01**
F28	**SCA**	2.271E+03	3.453E+03	3.380E+03	4.709E+03	5.709E+02
	m-SCA	**6.651E+02**	**1.054E+03**	**9.848E+02**	**1.996E+03**	**2.735E+02**
F29	**SCA**	9.935E+06	6.848E+07	7.313E+07	1.154E+08	2.427E+07
	m-SCA	**1.126E+04**	**4.443E+04**	**4.412E+04**	**8.605E+04**	**1.769E+04**
F30	**SCA**	4.129E+05	1.299E+06	1.243E+06	4.116E+06	6.792E+05
	m-SCA	**1.653E+04**	**3.484E+04**	**3.488E+04**	**5.962E+04**	**1.054E+04**

6. Applications of m-SCA

In this section, proposed modified Sine Cosine Algorithm (m-SCA) is implemented on five engineering optimization problems – two unconstrained and three constrained design problems. In constraint problems, the best solution and current best solution is obtained by using constraint violation value. For a general optimization problem

$$\textbf{\textit{Min}} \quad f(x),\ x \in R^n$$

$$\textbf{\textit{s.t.}} \quad g_i(x) \le 0 \quad i = 1, 2, \ldots m$$

$$h_j(x) = 0 \quad j = m+1, m+2, \ldots, p,$$

the constraint violation corresponding to the solution x is calculated as -

$$viol = \sum_{i=1}^{m} G_i(x) + \sum_{j=m+1}^{p} H_j(x)$$

Table 9
Statistical decision based on Wilcoxon Signed rank test at 5% level of significance on benchmark test set 2.

Test function	*p*-value	Decision	Test function	*p*-value	Decision
F1	5.145E−10	+	**F16**	5.145E−10	+
F2	5.145E−10	+	**F17**	5.145E−10	+
F3	5.460E−10	+	**F18**	5.145E−10	+
F4	5.145E−10	+	**F19**	5.145E−10	+
F5	2.660E−07	+	**F20**	5.460E−10	+
F6	5.145E−10	+	**F21**	5.460E−10	+
F7	5.145E−10	+	**F22**	5.145E−10	+
F8	5.145E−10	+	**F23**	5.145E−10	+
F9	5.145E−10	+	**F24**	5.145E−−10	+
F10	5.140E−10	+	**F25**	5.145E−10	+
F11	5.145E−10	+	**F26**	5.145E−10	+
F12	5.145E−10	+	**F27**	5.145E−10	+
F13	5.145E−10	+	**F28**	1.250E−09	+
F14	5.145E−10	+	**F29**	5.145E−10	+
F15	5.145E−10	+	**F30**	1.670E−09	+

Table 10
Statistical decision based on Wilcoxon Signed rank test at 5% level of significance on benchmark test set 2 corresponding to 30 dimensional problems.

Test function	*p*-value	Decision	Test function	*p*-value	Decision
F1	5.145E−10	+	**F16**	5.145E−10	+
F2	5.145E−10	+	**F17**	5.145E−10	+
F3	5.145E−10	+	**F18**	5.145E−10	+
F4	5.145E−10	+	**F19**	5.145E−10	+
F5	3.580E−07	+	**F20**	5.145E−10	+
F6	5.145E−10	+	**F21**	5.145E−10	+
F7	5.145E−10	+	**F22**	5.145E−10	+
F8	5.145E−10	+	**F23**	5.145E−10	+
F9	5.145E−10	+	**F24**	5.145E−10	+
F10	5.145E−10	+	**F25**	5.145E−10	+
F11	5.145E−−10	+	**F26**	5.145E−−10	+
F12	5.145E−10	+	**F27**	5.145E−10	+
F13	5.145E−10	+	**F28**	5.145E−10	+
F14	5.145E−10	+	**F29**	5.145E−10	+
F15	5.145E−10	+	**F30**	5.145E−10	+

Table 11
Performance comparison of proposed m-SCA with other meta-heuristic algorithms with 30 dimensional CEC2014 problems.

Problem	SCA	basic PSO	m1-PSO	m2-PSO	ABC	HS	SSA	MFO	m-SCA
F1	6.28E+08	7.62E+07	8.99E+07	3.88E+06	1.05E+08	1.19E+07	1.72E+06	7.73E+07	2.26E+07
F2	5.01E+10	7.74E+09	9.00E+09	2.79E+08	6.95E+04	1.33E+04	1.06E+04	1.25E+10	8.11E+07
F3	1.08E+05	2.62E+04	2.87E+04	2.82E+02	4.10E+04	7.46E+03	1.04E+03	9.86E+04	2.59E+04
F4	9.41E+03	4.31E+02	5.07E+02	1.03E+02	1.06E+02	5.27E+02	9.00E+01	9.38E+02	1.83E+02
F5	2.10E+01	2.09E+01	2.09E+01	2.06E+01	2.09E+01	5.20E+02	2.01E+01	2.03E+01	2.09E+01
F6	4.03E+01	2.73E+01	2.84E+01	1.12E+01	3.19E+01	6.14E+02	1.90E+01	2.40E+01	1.46E+01
F7	6.64E+02	7.27E+01	7.81E+01	6.73E+00	2.14E−04	7.00E+02	1.40E−02	1.06E+02	2.01E+00
F8	3.88E+02	1.86E+02	1.93E+02	4.19E+01	1.68E+02	8.00E+02	1.02E+02	1.49E+02	1.10E+02
F9	4.53E+02	2.21E+02	2.24E+02	8.68E+01	2.01E+02	9.69E+02	1.21E+02	2.14E+02	1.31E+02
F10	7.12E+03	6.14E+03	6.21E+03	9.48E+02	5.62E+03	1.00E+03	3.49E+03	3.47E+03	3.58E+03
F11	7.63E+03	6.63E+03	6.76E+03	3.00E+03	6.83E+03	3.08E+03	3.69E+03	4.13E+03	4.93E+03
F12	2.95E+00	2.52E+00	2.40E+00	2.30E−01	2.37E+00	1.20E+03	4.91E−01	4.63E−01	1.76E+00
F13	7.60E+00	1.18E+00	1.37E+00	4.99E−01	4.52E−01	1.30E+03	4.91E−01	2.01E+00	3.86E−01
F14	2.51E+02	1.93E+01	2.33E+01	1.55E+00	2.80E−01	1.40E+03	2.99E−01	2.67E+01	2.65E−01
F15	5.71E+06	6.26E+02	8.89E+02	5.63E+00	1.77E+01	1.51E+03	8.78E+00	1.94E+05	1.51E+01
F16	1.35E+01	1.24E+01	1.24E+01	1.08E+01	1.28E+01	1.61E+03	1.17E+01	1.28E+01	1.20E+01
F17	4.38E+07	2.29E+06	2.49E+06	3.20E+05	2.56E+06	2.01E+06	1.02E+05	4.31E+06	5.99E+05
F18	1.93E+09	7.36E+07	8.94E+07	1.45E+05	4.36E+02	6.62E+03	6.64E+03	8.02E+06	1.61E+05
F19	4.41E+02	4.01E+01	4.59E+01	1.06E+01	9.76E+00	1.92E+03	1.74E+01	8.91E+01	1.93E+01
F20	5.28E+05	3.15E+03	3.56E+03	4.79E+02	1.26E+04	8.80E+03	3.69E+02	7.25E+04	1.22E+04
F21	1.46E+07	6.72E+05	8.06E+05	1.27E+05	4.27E+05	5.38E+05	6.80E+04	1.85E+06	1.10E+05
F22	1.78E+03	5.81E+02	6.39E+02	4.18E+02	3.51E+02	2.71E+03	3.78E+02	7.98E+02	2.57E+02
F23	1.08E+03	3.64E+02	3.73E+02	3.17E+02	3.15E+02	2.62E+03	3.15E+02	3.86E+02	3.21E+02
F24	4.58E+02	2.75E+02	2.79E+02	2.24E+02	2.25E+02	2.63E+03	2.41E+02	2.93E+02	2.00E+02
F25	2.94E+02	2.14E+02	2.15E+02	2.07E+02	2.25E+02	2.71E+03	2.11E+02	2.13E+02	2.01E+02
F26	1.08E+02	1.33E+02	1.34E+02	1.53E+02	1.00E+02	2.74E+03	1.01E+02	1.07E+02	1.00E+02
F27	1.07E+03	7.89E+02	8.01E+02	6.09E+02	3.69E+02	3.37E+03	6.96E+02	9.31E+02	4.33E+02
F28	3.45E+03	1.60E+03	1.63E+03	1.61E+03	9.11E+02	3.88E+03	1.05E+03	1.16E+03	1.05E+03
F29	6.85E+07	4.16E+06	4.24E+06	3.36E+06	2.53E+03	4.36E+03	1.73E+06	2.55E+06	4.44E+04
F30	1.30E+06	3.85E+04	5.05E+04	1.19E+04	3.87E+03	7.34E+03	8.34E+03	2.96E+04	3.48E+04

Table 12
Comparison of results on Parameter estimation for frequency-modulated.

Algorithm	Min	Mean	Max	STD
m-SCA	3.86	15.41	20.65	3.38
SCA	12.46	22.50	25.18	3.85
G-CMA-ES	3.33	38.75	55.09	16.77
CPSOH	3.45	27.08	42.52	60.61
PSO	25.24	27.631	29.65	1.17

$$G_i(x) = \begin{cases} g_i(x) & if \quad g_i(x) > 0 \\ 0 & otherwise \end{cases}$$

$$H_j(x) = \begin{cases} |h_j(x)| & if \ |h_j(x)| - \in > 0 \\ 0 & otherwise \end{cases}$$

here $\in$ is predefined tolerance parameter which is fixed as 10^{-4} in the present paper. f is an objective function and g_i, h_j are inequality and equality constraints respectively. m and p represent the number of inequality and equality constraints respectively. Here the objective function and constraints can be linear or nonlinear.

6.1. Parameter estimation for frequency-modulated (FM) (Das & Suganthan, 2010)

The objective of this problem is to estimate the decision parameters of Frequency-modulated synthesizer. This problem is highly complex and multimodal with strong epistasis. In this problem, six decision variables are involved. The mathematical form of the problem can be stated as follows:

$$Min \quad F_1(X) = \sum_{t=1}^{100} (Y(t) - Y_0(t))^2, \ X = (a_1, \ \omega_1, a_2, \omega_2, a_3, \omega_3)$$

$$s.t. \quad -6.4 \le a_1, \ \omega_1, a_2, \omega_2, a_3, \omega_3 \le 6.35$$

where

$$Y(t) = a_1 \sin(\omega_1 \ t \ \theta + a_2 \ \sin(\omega_2 \ t \ \theta + a_3 \ \sin(\omega_3 \ t \ \theta)))$$

The expression for target sound waves is given by –

$$Y_0(t) = a_1 \sin\left(5 \ t \ \times \frac{2\pi}{100} + 1.5 \ \sin\left(4.8 \ t \ \times \frac{2\pi}{100} + 2 \ \sin\left(4.9 \ t \ \times \frac{2\pi}{100}\right)\right)\right)$$

The obtained best solutions by m-SCA and classical SCA are presented in Table 12. In the literature, this problem has been solved by various meta-heuristic search algorithms. In Table 12, the results obtained from Particle Swarm Optimization (PSO) (Kennedy, 2011), G-CMA-ES (Liang, Qin, Suganthan, & Baskar, 2006; Van den Bergh & Engelbrecht, 2004) and CPSOH (Auger & Hansen, 2005; Van den Bergh & Engelbrecht, 2004) are also reported. From the results, it can be analyzed that the proposed modified Sine Cosine Algorithm (m-SCA) is better in terms of mean, worst and standard deviation (STD) of objective function value. In terms of best objective function value, the proposed algorithm is very competitive than other algorithms. From the results, obtained from m-SCA it can be analyzed that the proposed algorithm is reliable to estimate the parameters for frequency-modulated.

6.2. Gear Train design problem

As shown in Fig. 9, Gear train design problem can be considered as discrete optimization problems as in this problem the decision parameters x_1, x_2,x_3 and x_4 represent the numbers of tooths

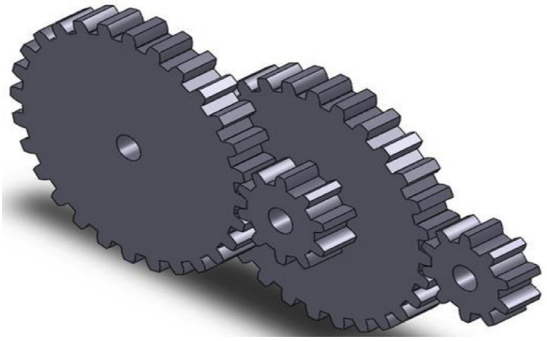

Fig. 9. Gear Train Design Problem.

for four gears of a train. The objective of gear train design problem is to find the optimal number of a tooth to minimize the gear ratio (Gandomi, 2014; Sandgren, 1990). The discrete parameters in the algorithm are handled by rounding them to the nearest integral value after updating the solution using a considered algorithm. Mathematically, the problem can be formulated as follows –

$$\textbf{\textit{Min}} \quad F_2(x) = \left(\frac{1}{6.931} - \frac{x_2 x_3}{x_1 x_4}\right)^2, \ \text{where } x = (x_1, x_2, x_3, x_4)$$

$$\textbf{\textit{s.t.}} \quad 12 \le x_i \le 60, \ \text{and } x_i \in \ Z^+ \ \forall i = 1, 2, 3, 4.$$

The obtained best solutions by proposed algorithm m-SCA and classical SCA are presented in Table 13. This problem is solved in the literature by Augmented Lagrange multiplier (ALM) (Kannan & Kramer, 1994), Genetic Algorithm (GA) (Deb & Goyal, 1996), Artificial Bee Colony (ABC) algorithm (Karaboga & Basturk, 2007), Cuckoo Search (CS) algorithm (Gandomi, Yang, & Alavi, 2013) and Mine Blast Algorithm (MBA) (Sadollah, Bahreininejad, Eskandar, & Hamdi, 2013). From the comparison of results between proposed m-SCA and other state-of-the-art algorithms it can be observed that the proposed modified Sine Cosine Algorithm (m-SCA) is better in terms of used function evaluation cost as well as best objective function value. As CS and MBA provide the same objective function value as m-SCA but with more computational cost.

6.3. Three bar truss design problem

This problem was presented in Nowcki (1974). This design problem is a non-linear fraction optimization problem. In this problem, the volume of a statically loaded three-bar truss is minimized subject to stress constraints on each truss members. In the problem only two decision parameters x_1 and x_2 are involved. Mathematically, the problem is stated as follows:

$$\textbf{\textit{Min}} \quad F_3(X) = L \times \left(2\sqrt{2} \ x_1 + x_2 \ \right)$$

$$\textbf{\textit{s.t.}} \quad \frac{\sqrt{2} \ x_1 + x_2}{2x_1x_2 + \sqrt{2}x_1^2} P \le \sigma$$

$$\frac{x_2}{2x_1x_2 + \sqrt{2}x_1^2} P \le \sigma$$

$$\frac{1}{x_1 + \sqrt{2}x_2} P \le \sigma$$

$$0.01 \le x_i \le 1. \ \text{for } i = 1, \ 2.$$

where,

$L = 100\,\text{cm}$ $P = 2\,\text{km/cm}^2$

$\sigma = 2\,\text{km/cm}^2$

Table 13
Comparison of results on gear train design problem.

Algorithm	Optimal decision variables				Objective function value ($F_{2,min}$)	Number of function evaluations
	x_1	x_2	x_3	x_4		
m-SCA	43	16	19	49	**2.7009E−12**	**1270**
SCA	55	28	17	60	1.3616E−09	1270
ALM	33	15	13	41	2.4073E−08	NA
GA	33	14	17	50	1.3616E−09	NA
ABC	43	16	19	49	2.7800E−11	40,000
CS	43	16	19	49	2.7009E−12	5000
MBA	43	16	19	49	2.7009E−−12	10,000

Table 14
Comparison of results on three bar truss design problem.

Algorithm	Decision variables		Objective function value ($F_{3,min}$)
	x_1	x_2	
m-SCA	**0.81915**	**0.36956**	**263.8972**
SCA	0.78669	0.41426	263.9348
CS	0.78867	0.40902	263.9716
Tsai (2005)	0.788	0.408	263.68*
Ray and Saini (2001)	0.795	0.395	264.30

* Represent that the obtained solution is infeasible.

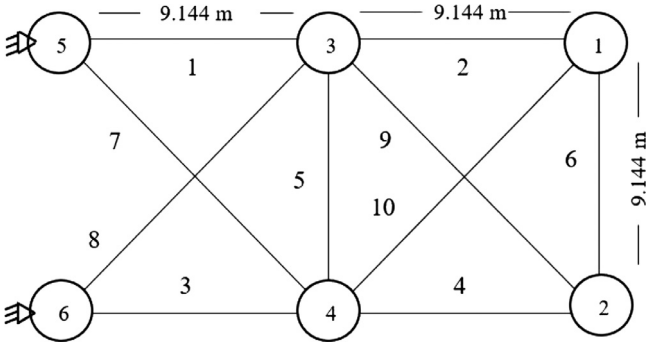

Fig. 10. Ten bar truss design.

The obtained best solutions by m-SCA is presented in Table 14. In the literature, this problem has been tried to solve by various algorithms. The solutions obtained by Tsai (2005) and Ray and Saini (2001) are also presented in the same table. This problem is solved by Cuckoo Search (Gandomi et al., 2013) algorithm also. To solve this problem 25 search agents and 15,000 function evaluation is fixed, which is same as in Gandomi et al. (2013). From the results it can be observed that proposed m-SCA outperforms other techniques to solve this problem.

6.4. Ten bar truss design problem

In this case study, similar to three-bar truss design, the mass or weight of a truss is minimized. In this problem ten decision parameters (x_1,x_2,..., x_{10}) are involved. The truss structure of this design problem is shown in Fig. 10. Mathematically, the problem can be stated as follows:

$$Min \quad F_4(x) = \sum_{j=1}^{m} l_j \rho_j x_j$$

$$s.t. \quad \eta_i \geq \eta_i^* \quad i = 1, 2, \ldots q_1.$$

$$\eta_i \geq \eta_i^* \quad i = q_1 + 1, q_1 + 2, \ldots, r.$$

$$x_j^{min} \leq x_j \leq x_j^{max} \quad j = 1, 2, \ldots, m \text{ for all bars}$$

Table 15
Design parameter for ten bar truss design problem.

Parameters	Parameter details
Decision variables	x_1,x_2,..., x_{10}
Bounds	$\eta_1 \geq 7$, $\eta_2 \geq 15$, $\eta_3 \geq 20$ Hz
Bounds for decision variable	[0.645, 50]c.m.^{2}
Density (ρ)	2770 k.g./m^3
Modulus of elasticity E	6.98×10^{10}Pa

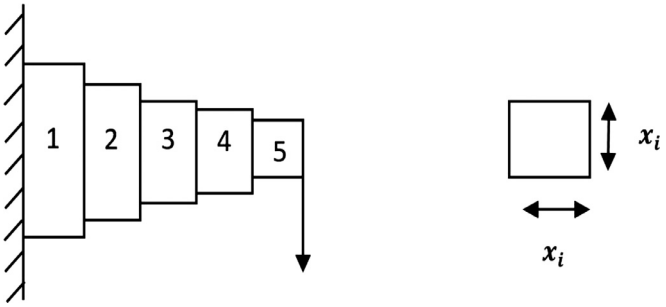

Fig. 11. Cantilever beam design.

The constraint violation has been used with the objective function to minimize the weight as in Gomes (2011). Thus the objective function for this problem is considered as:

$$Min \quad F_4(x) = \left(\sum_{j=1}^{m} l_j \rho_j x_j \right) (1 + PF)$$

where $PF = \sum |\frac{\eta_i}{\eta_i^*} - 1|$

The design parameter for this case study is provided in Table 15. To solve this problem 5000 function evaluations has been used with 20 size of population. This example was first solved by Grandhi and Venkayya (1988) using the optimality algorithm. The results obtained from the proposed algorithm are presented in the Table 16. In the same table results of various study, Grandhi (1993), Lingyun, Mei, Guangming, and Guang (2005), Sedaghati, Suleman, and Tabarrok (2002) and Wang, Zhang, and Jiang (2004) - are presented which are used to solve this problem in the literature. From the table it can be analyzed that the proposed algorithm is very competitive than other algorithms and outperforms classical SCA.

6.5. Cantilever beam design problem

This is the case study of optimizing the weight of cantilever beam with square cross section as shown in Fig. 11. The beam is supported at node 1 and a given force is acting at node 5. In this problem the decision parameters are height or width of different beam. Mathematically, the problem can be stated as follows:

$$\textbf{\textit{Min}} \quad F_5(X) = 0.0624 \times \sum_{i=1}^{5} x_i$$

$$\textbf{\textit{s.t.}} \quad \frac{61}{x_1^3} + \frac{37}{x_2^3} + \frac{19}{x_3^3} + \frac{7}{x_4^3} + \frac{1}{x_5^3} \leq 1$$

Table 16
Comparison of results on ten bar truss design problem.

Decision variable	Wang et al. (2004)	Lingyun et al. (2005)	Sedaghati et al. (2002)	Grandhi (1993)	SCA	m-SCA
x_1	32.45	42.23	38.25	36.58	32.433	32.347
x_2	16.57	18.56	9.92	24.66	16.921	12.101
x_3	32.45	38.85	38.61	36.58	38.904	44.483
x_4	16.57	11.22	18.23	24.66	19.959	10.808
x_5	2.11	4.78	4.42	4.17	9.550	9.887
x_6	4.47	4.45	4.42	2.07	22.126	7.859
x_7	22.81	21.05	20.10	27.03	20.587	8.827
x_8	22.81	20.95	24.10	23.03	32.814	34.201
x_9	17.49	10.26	13.90	10.35	24.270	23.874
x_{10}	17.49	14.34	11.45	10.35	25.513	14.781
Weight	553.80	542.75	537.01	594.00	593.268	515.2603

Table 17
Comparison of results on cantilever beam design problem.

Algorithm	Optimum decision variables					Objective function value ($F_{4,min}$)
	x_1	x_2	x_3	x_4	x_5	
m-SCA	6.0089	5.3049	4.5023	3.5077	2.1504	1.33999
SCA	6.0100	5.3000	4.4900	3.4900	2.1500	1.3400
CS	6.0089	5.3049	4.5023	3.5077	2.1504	1.33999
GCA (I)	6.0100	5.3000	4.4900	3.4900	2.1500	1.3400
GCA (II)	6.0100	5.3000	4.4900	3.4900	2.1500	1.3400
MMA	6.0100	5.3000	4.4900	3.4900	2.1500	1.3400
CONLIN	6.0100	5.3000	4.4900	3.4900	2.1500	1.3400

$0.01 \leq x_i \leq 100.\ for\ all\ i = 1, 2, \ldots, 5.$

The best fitness value of the problem obtained by m-SCA and various other techniques (Chickermane & Gea, 1996; Gandomi et al., 2013) are presented in Table 17. To solve this problem 50 search agents and 2500 searches have been used, which is same as in Gandomi et al. (2013). From the table, it can be observed that the solution obtained from m-SCA is better as compared to other techniques.

7. Conclusion and future scope

The present work proposes a modified Sine Cosine Algorithm (m-SCA) for global optimization problems. This modified version of classical SCA has been proposed by finding out the drawbacks of low diversity within the population of solutions and skipping of true solutions in classical SCA. Therefore, an opposition based learning is integrated into m-SCA to jump out from local optima during the search process. This phase of generating the opposite population of solutions is managed by jumping or perturbation rate. Moreover, in m-SCA, a self-adaptation learning of solutions is also integrated into the search equations of classical SCA. This new search equation is proposed to achieve the efficient state of solutions in SCA. In self-adaptation learning, solution stores their own best state in their memory and utilizes it to update their state during the search process. This self-adaptation learning helps in exploiting all the promising search regions around pre-obtained solutions. The proposed modified-SCA optimizer is evaluated on well-known classical set of benchmark problems and standard IEEE CEC 2014 benchmark test set. The analysis based on the average distance between solutions in each iteration, ensures that the strategies integrated into m-SCA enhanced the diversity of solutions within the population. The convergence and statistical analysis of the obtained results demonstrate the superior performance of proposed modified sine cosine algorithm (m-SCA) compared to classical SCA.

In the paper, five engineering application problems are also used to examine the performance of the proposed algorithm. The results on the engineering optimization problems also verify the significant improvement in m-SCA as compared to classical SCA. The comparison with other optimization algorithms also favors the significant improvement in m-SCA.

In future direction, the m-SCA can be implemented on complex real-life application problems. The constrained and multi-objective version of m-SCA can be developed also.

Acknowledgment

The first author gratefully acknowledges the Ministry of Human Resource and Development Ministry of Human Resource and Development (MHRD), Govt. of India, India for their financial support. Grant no. MHR-02-41-113-429.

References

Assad, A., & Deep, K. (2018). A Hybrid Harmony search and Simulated Annealing algorithm for continuous optimization. *Information Sciences, 450*, 246–266.
Auger, A., & Hansen, N. (2005, September). A restart CMA evolution strategy with increasing population size. In *Evolutionary computation, 2005. The 2005 IEEE congress on: Vol. 2* (pp. 1769–1776). IEEE.
Bansal, J. C., Sharma, H., Jadon, S. S., & Clerc, M. (2014). Spider monkey optimization algorithm for numerical optimization. *Memetic Computing, 6*(1), 31–47.
Chickermane, H., & Gea, H. C. (1996). Structural optimization using a new local approximation method. *International Journal for Numerical Methods in Engineering, 39*(5), 829–846.
Črepinšek, M., Liu, S. H., & Mernik, M. (2013). Exploration and exploitation in evolutionary algorithms: A survey. *ACM Computing Surveys (CSUR), 45*(3), 35.
Das, S., & Suganthan, P. N. (2010). *Problem definitions and evaluation criteria for CEC 2011 competition on testing evolutionary algorithms on real world optimization problems*. Kolkata: Jadavpur University, Nanyang Technological University.
Deb, K., & Goyal, M. (1996). A combined genetic adaptive search (GeneAS) for engineering design. *Computer Science and Informatics, 26*, 30–45.
Deep, K., & Thakur, M. (2007). A new crossover operator for real coded genetic algorithms. *Applied Mathematics and Computation, 188*(1), 895–911.
Djenouri, Y., Belhadi, A., & Belkebir, R. (2018). Bees swarm optimization guided by data mining techniques for document information retrieval. *Expert Systems with Applications, 94*, 126–136.
Dorigo, M., & Birattari, M. (2011). Ant colony optimization. In *Encyclopedia of machine learning* (pp. 36–39). Boston, MA: Springer.
Eberhart, R., & Kennedy, J. (1995 October). A new optimizer using particle swarm theory. In *Micro machine and human science, 1995. MHS'95, proceedings of the sixth international symposium on* (pp. 39–43). IEEE.
El Azi, M. A., Ewees, A. A., & Hassanien, A. E. (2017). Whale Optimization Algorithm and Moth-Flame Optimization for multilevel thresholding image segmentation. *Expert Systems with Applications, 83*, 242–256.
El Aziz, M. A., Oliva, D., & Xiong, S. (2017). An improved opposition-based sine cosine algorithm for global optimization. *Expert Systems with Applications, 90*, 484–500.

Ertenlice, O., & Kalayci, C. B. (2018). A survey of swarm intelligence for portfolio optimization: Algorithms and applications. *Swarm and Evolutionary Computation, 39*, 36–52.

Formato, R. A. (2007). Central force optimization. *Progress in Electromagnetics Research, 77*, 425–491.

Gandomi, A. H. (2014). Interior search algorithm (ISA): A novel approach for global optimization. *ISA Transactions, 53*(4), 1168–1183.

Gandomi, A. H., Yang, X. S., & Alavi, A. H. (2013). Cuckoo search algorithm: A metaheuristic approach to solve structural optimization problems. *Engineering with Computers, 29*(1), 17–35.

Geem, Z. W., Kim, J. H., & Loganathan, G. V. (2001). A new heuristic optimization algorithm: Harmony search. *Simulation, 76*(2), 60–68.

Gomes, H. M. (2011). Truss optimization with dynamic constraints using a particle swarm algorithm. *Expert Systems with Applications, 38*(1), 957–968.

Grandhi, R. (1993). Structural optimization with frequency constraints-a review. *AIAAJournal, 31*(12), 2296–2303.

Hafez, A. I., Zawbaa, H. M., Emary, E., & Hassanien, A. E. (2016 August). Sine cosine optimization algorithm for feature selection. In *Innovations in intelligent systems and applications (INISTA), 2016 International Symposium on* (pp. 1–5). IEEE.

Hatamlou, A. (2013). Black hole: A new heuristic optimization approach for data clustering. *Information Sciences, 222*, 175–184.

Hoffmann, J., & Nebel, B. (2001). The FF planning system: Fast plan generation through heuristic search. *Journal of Artificial Intelligence Research, 14*, 253–302.

Holland, J. H. (1992). *Adaptation in natural and artificial systems: An introductory analysis with applications to biology, control, and artificial intelligence*. MIT Press.

Ismael, S. M., Aleem, S. H. A., & Abdelaziz, A. Y. (2017 December). Optimal selection of conductors in Egyptian radial distribution systems using sine-cosine optimization algorithm. In *Power systems conference (MEPCON), 2017 nineteenth international middle east* (pp. 103–107). IEEE.

Kannan, B. K., & Kramer, S. N. (1994). An augmented Lagrange multiplier based method for mixed integer discrete continuous optimization and its applications to mechanical design. *Journal of Mechanical Design, 116*(2), 405–411.

Kar, A. K. (2016). Bio inspired computing–A review of algorithms and scope of applications. *Expert Systems with Applications, 59*, 20–32.

Karaboga, D., & Basturk, B. (2007). A powerful and efficient algorithm for numerical function optimization: Artificial bee colony (ABC) algorithm. *Journal of Global Optimization, 39*(3), 459–471.

Kaveh, A., & Khayatazad, M. (2012). A new meta-heuristic method: Ray optimization. *Computers & Structures, 112*, 283–294.

Kennedy, J. (2011). Particle swarm optimization. In *Encyclopedia of machine learning* (pp. 760–766). Boston, MA: Springer.

Khairuzzaman, A. K. M., & Chaudhury, S. (2017). Multilevel thresholding using grey wolf optimizer for image segmentation. *Expert Systems with Applications, 86*, 64–76.

Kumar, V., & Kumar, D. (2017). Data clustering using sine cosine algorithm: Data clustering using SCA. In *Handbook of research on machine learning innovations and trends* (pp. 715–726). IGI Global.

Kumawat, I. R., Nanda, S. J., & Maddila, R. K. (2018). Positioning LED panel for uniform illuminance in indoor VLC system using whale optimization. In *Optical and wireless technologies* (pp. 131–139). Singapore: Springer.

Li, S., Fang, H., & Liu, X. (2018). Parameter optimization of support vector regression based on sine cosine algorithm. *Expert Systems with Applications, 91*, 63–77.

Liang, J. J., Qin, A. K., Suganthan, P. N., & Baskar, S. (2006). Comprehensive learning particle swarm optimizer for global optimization of multimodal functions. *IEEE Transactions on Evolutionary Computation, 10*(3), 281–295.

Liang, J. J., Qu, B. Y., & Suganthan, P. N. (2013). *Problem definitions and evaluation criteria for the CEC 2014 special session and competition on single objective real–parameter numerical optimization*. Singapore: Computational Intelligence Laboratory, Zhengzhou University, Zhengzhou China and Technical Report, Nanyang Technological University.

Lingyun, W., Mei, Z., Guangming, W., & Guang, M. (2005). Truss optimization on shape and sizing with frequency constraints based on genetic algorithm. *Computational Mechanics, 35*(5), 361–368.

Mahdavi, S., Rahnamayan, S., & Deb, K. (2018). Opposition based learning: A literature review. *Swarm and Evolutionary Computation, 39*, 1–23.

Mavrovouniotis, M., Li, C., & Yang, S. (2017). A survey of swarm intelligence for dynamic optimization: Algorithms and applications. *Swarm and Evolutionary Computation, 33*, 1–17.

Mirjalili, S. (2015a). Moth-flame optimization algorithm: A novel nature-inspired heuristic paradigm. *Knowledge-Based Systems, 89*, 228–249.

Mirjalili, S. (2015b). The ant lion optimizer. *Advances in Engineering Software, 83*, 80–98.

Mirjalili, S. (2016). SCA: A sine cosine algorithm for solving optimization problems. *Knowledge-Based Systems, 96*, 120–133.

Mirjalili, S., Gandomi, A. H., Mirjalili, S. Z., Saremi, S., Faris, H., & Mirjalili, S. M. (2017). Salp Swarm Algorithm: A bio-inspired optimizer for engineering design problems. *Advances in Engineering Software, 114*, 163–191.

Mirjalili, S., & Lewis, A. (2016). The whale optimization algorithm. *Advances in Engineering Software, 95*, 51–67.

Mirjalili, S., Mirjalili, S. M., & Hatamlou, A. (2016). Multi-verse optimizer: A nature-inspired algorithm for global optimization. *Neural Computing and Applications, 27*(2), 495–513.

Mirjalili, S., Mirjalili, S. M., & Lewis, A. (2014). Grey wolf optimizer. *Advances in Engineering Software, 69*, 46–61.

Nenavath, H., Jatoth, R. K., & Das, S. (2018). A synergy of the sine-cosine algorithm and particle swarm optimizer for improved global optimization and object tracking. *Swarm and Evolutionary Computation..*

Nowcki, H. (1974). Optimization in pre-contract ship design. In Y. Fujita, K. Lind, & T. J. . Williams (Eds.). In *Computer applications in the automation of shipyard operation and ship design: Vol. 2* (pp. 327–338). North HollandNew York: Elsevier.

Połap, D. (2017). Polar Bear Optimization Algorithm: Meta-heuristic with fast population movement and dynamic birth and death mechanism. *Symmetry, 9*(10), 203.

Rahnamayan, S., Tizhoosh, H. R., & Salama, M. M. (2008). Opposition versus randomness in soft computing techniques. *Applied Soft Computing, 8*(2), 906–918.

Rashedi, E., Nezamabadi-Pour, H., & Saryazdi, S. (2009). GSA: A gravitational search algorithm. *Information Sciences, 179*(13), 2232–2248.

Ray, T., & Saini, P. (2001). Engineering design optimization using a swarm with an intelligent information sharing among individuals. *Engineering Optimization, 33*(6), 735–748.

Reddy, K. S., Panwar, L. K., Panigrahi, B. K., & Kumar, R. (2017). A New Binary Variant of Sine–Cosine Algorithm: Development and application to solve profit-based unit commitment problem. *Arabian Journal for Science and Engineering*, 1–16.

Sadollah, A., Bahreininejad, A., Eskandar, H., & Hamdi, M. (2013). Mine blast algorithm: A new population based algorithm for solving constrained engineering optimization problems. *Applied Soft Computing, 13*(5), 2592–2612.

Sahlol, A. T., Ewees, A. A., Hemdan, A. M., & Hassanien, A. E. (2016 December). Training feedforward neural networks using Sine-Cosine algorithm to improve the prediction of liver enzymes on fish farmed on nano-selenite. In *Computer engineering conference (ICENCO), 2016 12th international* (pp. 35–40). IEEE.

Salimi, H. (2015). Stochastic fractal search: A powerful metaheuristic algorithm. *Knowledge-Based Systems, 75*, 1–18.

Sandgren, E. (1990). Nonlinear integer and discrete programming in mechanical design optimization. *Journal of Mechanical Design, 112*(2), 223–229.

Saremi, S., Mirjalili, S., & Lewis, A. (2017). Grasshopper optimisation algorithm: theory and application. *Advances in Engineering Software, 105*, 30–47.

Sedaghati, R., Suleman, A., & Tabarrok, B. (2002). Structural optimization with frequency constraints using the finite element force method. *AIAA Journal, 40*(2), 382–388.

Shi, Y., & Eberhart, R. (1998 May). A modified particle swarm optimizer. In *Evolutionary computation proceedings, 1998. IEEE world congress on computational intelligence, the 1998 IEEE international conference on* (pp. 69–73). IEEE.

Sindhu, R., Ngadiran, R., Yacob, Y. M., Zahri, N. A. H., & Hariharan, M. (2017). Sine–Cosine algorithm for feature selection with elitism strategy and new updating mechanism. *Neural Computing and Applications, 28*(10), 2947–2958.

Singh, H., Kumar, A., Balyan, L. K., & Singh, G. K. (2017). Swarm intelligence optimized piecewise gamma corrected histogram equalization for dark image enhancement. *Computers & Electrical Engineering*.

Storn, R., & Price, K. (1997). Differential evolution–a simple and efficient heuristic for global optimization over continuous spaces. *Journal of Global Optimization, 11*(4), 341–359.

Tizhoosh, H. R. (2005 November). Opposition-based learning: A new scheme for machine intelligence. In *Computational intelligence for modelling, control and automation, 2005 and international conference on intelligent agents, web technologies and internet commerce, international conference on: Vol. 1* (pp. 695–701). IEEE.

Tsai, J. F. (2005). Global optimization of nonlinear fractional programming problems in engineering design. *Engineering Optimization, 37*(4), 399–409.

Van den Bergh, F., & Engelbrecht, A. P. (2004). A cooperative approach to particle swarm optimization. *IEEE Transactions on Evolutionary Computation, 8*(3), 225–239.

Van Laarhoven, P. J., & Aarts, E. H. (1987). Simulated annealing. In *Simulated annealing: Theory and applications* (pp. 7–15). Dordrecht: Springer.

Ventresca, M., & Tizhoosh, H. R. (2008). Two frameworks for Improving Gradient-Based Learning Algorithms. In *Oppositional concepts in computational intelligence* (pp. 255–284). Berlin, Heidelberg: Springer.

Wang, D., Zhang, W. H., & Jiang, J. S. (2004). Truss optimization on shape and sizing with frequency constraints. *AIAA Journal, 42*(3), 622–630.

Webster, B., & Bernhard, P. J. (2003). A local search optimization algorithm based on natural principles of gravitation. In *Proceedings of the 2003 international conference on information and knowledge engineering (IKE'03)* (pp. 255–261). USA: Las Vegas, Nevada.

Wolpert, D. H., & Macready, W. G. (1997). No free lunch theorems for optimization. *IEEE Transactions on Evolutionary Computation, 1*(1), 67–82.

Woźniak, M., & Połap, D. (2018a). Bio-inspired methods modeled for respiratory disease detection from medical images. *Swarm and Evolutionary Computation..* https://doi.org/10.1016/j.swevo.2018.01.008.

Woźniak, M., & Połap, D. (2018b). Adaptive neuro-heuristic hybrid model for fruit peel defects detection. *Neural Networks, 98*, 16–33. https://doi.org/10.1016/j.neunet.2017.10.009.

Yang, X. S. (2010). Firefly algorithm, stochastic test functions and design optimisation. *International Journal of Bio-Inspired Computation, 2*(2), 78–84.

Yang, C., Gao, W., Liu, N., & Song, C. (2015). Low-discrepancy sequence initialized particle swarm optimization algorithm with high-order nonlinear time-varying inertia weight. *Applied Soft Computing, 29*, 386–394.