Expert Systems With Applications 154 (2020) 113395

Contents lists available at ScienceDirect

Expert Systems With Applications

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

A modified Sine Cosine Algorithm with novel transition parameter and mutation operator for global optimization

Shubham Gupta [a], Kusum Deep [b,*], Seyedali Mirjalili [c], Joong Hoon Kim [d]

[a] Institute for Mega Construction, Korea University, Seoul 02841, South Korea
[b] Department of Mathematics, Indian Institute of Technology Roorkee, Roorkee, Uttarakhand 247667, India
[c] Centre for Artificial Intelligence Research and Optimisation, Torrens University Australia, 90 Bowen Terrace, Fortitude Valley, Queensland 4006, Australia
[d] School of Civil, Environmental and Architectural Engineering, Korea University, Seoul 02841, South Korea

ARTICLE INFO

Article history:
Received 14 October 2019
Revised 15 January 2020
Accepted 17 March 2020
Available online 28 March 2020

Keywords:
Optimization
Sine Cosine Algorithm
Exploration and exploitation
Multilayer perceptron
Engineering optimization problems
Algorithm
Benchmark
Grey Wolf Optimizer
Particle Swarm Optimization
Genetic Algorithm

ABSTRACT

Inspired by the mathematical characteristics of sine and cosine trigonometric functions, the Sine Cosine Algorithm (SCA) has shown competitive performance among other meta-heuristic algorithms. However, despite its sufficient global search ability, its low exploitation ability and immature balance between exploitation and exploration remain weaknesses. In order to improve Sine Cosine Algorithm (SCA), this paper presents a modified version of the SCA called MSCA. Firstly, a non-linear transition rule is introduced instead of a linear transition to provide comparatively better transition from the exploration to exploitation. Secondly, the classical search equation of the SCA is modified by introducing the leading guidance based on the elite candidate solution. When the above proposed modified search mechanism fails to provide a better solution, in addition, a mutation operator is used to generate a new position to avoid the situation of getting trapped in locally optimal solutions during the search. Thus, the MSCA effectively maximizes the advantages of proposed strategies in maintaining a comparatively better balance of exploration and exploitation as compared to the classical SCA. The validity of the MSCA is tested on a set of 33 benchmark optimization problems and employed for training multilayer perceptrons. The numerical results and comparisons among several algorithms show the enhanced search efficiency of the MSCA.

1. Introduction

Meta-heuristic optimization algorithms, which are being used widely, are designed to perform two crucial tasks which are known as exploration and exploitation (Blum & Roli, 2003; BoussaïD, Lepagnot, & Siarry, 2013). In the exploration, “macro searc” is conducted to find the most promising areas of the solution space. In the exploitation, however, an algorithm performs “micro search” to extract the useful information of the search areas which are discovered earlier and where the chance of having optimal solution is high. Exploration corresponds to global search in a provided feasible solution space to search for global optima and avoid the local solutions during the search. Exploitation corresponds to the local search with an aim of improving the available best solution. Due to the contradictory nature of these two concepts, a proper balance between exploration and exploitation should exist in any meta-heuristic algorithm to perform the well-behaved search. An immature balance leads to several problems including, but not limited to, skipping true solutions, local optima stagnation and premature convergence.

In real-life optimization problems, the global optima can be found using meta-heuristic algorithms. But, these algorithms always need to be modified because they can be trapped in local optima during the search. This fact can be supported with the No Free Lunch (NFL) theorem (Wolpert, Macready et al., 1995; 1997), which states that an ideal algorithm can not exist which will be suitable for all optimization problems. This theorem was the revolutionary development in the field of meta-heuristic algorithms, which makes this field highly active for the researchers and allows the modifications in existing algorithms to improve their performance.

Some of the well-known meta-heuristic algorithms are Genetic Algorithm (GA) (Holland John, 1975), Particle Swarm Optimization (PSO)(Kennedy, 2010), Gravitational Search Algorithm (GSA) (Rashedi, Nezamabadi-Pour, & Saryazdi, 2009), Artificial Bee

* Corresponding author.
E-mail addresses: g.shubh93@gmail.com, sgupta@ma.iitr.ac.in (S. Gupta), kusumfma@iitr.ac.in (K. Deep), ali.mirjalili@gmail.com (S. Mirjalili), jaykim@korea.ac.kr (J.H. Kim).

https://doi.org/10.1016/j.eswa.2020.113395
0957-4174/

Colony (ABC) algorithm (Karaboga & Basturk, 2007), Biogeography-based Optimization (BBO) (Simon, 2008), Cuckoo Search (CS) algorithm (Gandomi, Yang, & Alavi, 2013), Grey Wolf Optimizer (GWO) (Mirjalili, Mirjalili, & Lewis, 2014), Ant Colony Optimization (Dorigo & Birattari, 2010), Firefly Algorithm (FA) (Yang, 2010), Teaching-learning based optimization (TLBO) (Rao, Savsani, & Vakharia, 2011) and many others. Moreover, some of the recently developed meta-heuristics such as Harris Hawks Optimization (HHO) (Heidari et al., 2019), Electromagnetic Field Optimization (EFO) (Abedinpourshotorban, Shamsuddin, Beheshti, & Jawawi, 2016), Squirrel Search Algorithm (SSA) (Jain, Singh, & Rani, 2019), Lion Optimization Algorithm (LOA) (Yazdani & Jolai, 2016), Electron Radar Search Algorithm (ERSA) (Rahmanzadeh & Pishvaee, 2019), Henry Gas Solubility Optimization (HGSO) (Hashim, Houssein, Mabrouk, Al-Atabany, & Mirjalili, 2019), Electric Fish Optimization (EFO) (Yilmaz & Sen, 2019) have also shown their efficiency as a global optimizer. These algorithms are population-based, in which instead of single agent, multiple agents are used to perform the search. During the search procedure, agents communicate to each other by sharing their experience to enhance the exploration ability of the algorithm. These characteristics of meta-heuristic algorithms help in avoiding the local optima during the search process.

In the field of meta-heuristic algorithms, an algorithm called Sine Cosine Algorithm (SCA) (Mirjalili, 2016) was recently proposed by Mirjalili in 2016. Similar to other meta-heuristic algorithms, the SCA also suffers from the problem of local optima stagnation, premature convergence. Therefore, in the present paper, a modified version of the SCA, named MSCA is proposed based on employing various strategies. First, the transition parameter of the classical SCA is replaced with a non-linear function. Secondly, the search equation of the SCA is modified to provide a search based on the leading elite guidance. Third, a Gaussian mutation operator and chaotic map are introduced to prevent the situation of stagnation by provide a sudden random jump during the search process, when it is analyzed that the best obtained position fail to search an efficient direction of search in the algorithm. Computational experiments and comparisons on 33 benchmark test problems and on training the MLPs show that the proposed MSCA method outperforms classical SCA, modified variants of SCA and some other meta-heuristic algorithms.

The rest of the paper is structured as follows: Section 2 provides a brief review of the classical SCA and its literature review. Section 3 sets out the main contribution provided in this paper, which is a modified version the SCA (MSCA). Section 4 describes a computational experiment on a number of benchmark test problems to verify the efficiency of the MSCA on diverse category of optimization problems. In Section 5, the MSCA is used and compared with other optimization algorithms on the training of multilayer perceptrons. The results show that the MSCA is competitive and can be considered as a better optimizer than the classical SCA and other meta-heuristic algorithms. Finally, Section 6 concludes the work and suggests some future research directions.

2. Preliminaries

2.1. Sine Cosine Algorithm (SCA)

The Sine Cosine Algorithm (SCA) is a new meta-heuristic algorithm developed by Mirjalili (2016), which is inspired by the characteristics of trigonometric functions sine and cosine. In SCA, the solution vector is analogous to the candidate solution and the global best solution vector of the population is analogous to the destination point. The search equations which is involved in the SCA are as follows (Mirjalili, 2016):

$$x_{i,j}^{t+1} = \begin{cases} x_{i,j}^{t} - r_1 \sin(2\pi r_2)\left|2r_3 x_{D,j} - x_{i,j}^{t}\right| & r_4 > 0.5 \\ x_{i,j}^{t} - r_1 \cos(2\pi r_2)\left|2r_3 x_{D,j} - x_{i,j}^{t}\right| & r_4 \le 0.5 \end{cases} \quad (1)$$

where $x_{i,j}^{t}$ and $x_{i,j}^{t+1}$ are the positions of the candidate solution $X_i = (x_{i,1}, x_{i,2}, \ldots, x_{i,d}) \in \mathbf{R}^d$ for the jth dimension at iteration t and $(t+1)$, respectively. The random numbers r_2, r_3 and r_4 are uniformly distributed within the interval (0,1).

The vector $X_D = (x_{D,1}, x_{D,2}, \ldots, x_{D,d}) \in \mathbf{R}^d$ represents the destination point. The random number r_1 helps to provide a transition from the phase of exploration to exploitation, which can be formulated as:

$$r_1 = 2 - 2 \times \left(\frac{t}{T}\right) \quad (2)$$

where t indicates the current iteration counter and T is the maximum number of iterations fixed for the execution of an algorithm. The pseudo-code of the classical SCA is provided in Algorithm 1.

Algorithm 1 Pseudo-code of SCA.

Inputs: The population size N and maximum number of iterations T
Outputs: The position of the destination point and its fitness value
Initialize the population $X_i (i = 1, 2, \ldots, N)$ randomly and within the provided solution space
while (t<T) **do**
  Calculate the fitness value of each candidate solution
  Set X_D as position of destination point
  Update the parameter r_1 as given in Eq. (2)
  for (each candidate solution (X_i)) **do**
    Update the position using Eq. (1)
Return the destination point X_D

The classical SCA is good at identifying the new promising regions of the solution space within a reasonable time. But, it is not efficient in performing the local search around the best positions obtained so far (Gupta & Deep, 2019b). Moreover, in some cases, the candidate solutions of the SCA converge towards the local optimum. As a consequence, the algorithm fails to provide a sufficient mechanism to prevent from local optima. Therefore, few modified variants were developed for enhancing the solution accuracy and convergence rate. Elaziz, Oliva, and Xiong (2017) and Gupta and Deep (2019a) have modified the search mechanism of the SCA using opposition-based learning concept. Although, in Elaziz et al. (2017) and Gupta and Deep (2019a), an improvement has been observed but in most of the problems, the optima is not obtained by the proposed methods, which shows that the algorithm experiences the slow convergence rate and prone towards the local optima. In Gupta and Deep (2019b), crossover and personal best guidance of search are used to enhance the exploitation skills and search accuracy of the SCA. In Singh and Singh (2017), the exploitation ability of the classical SCA is enhanced by hybridizing it with the GWO algorithm. Attia, El Sehiemy, and Hasanien (2018) have combined the Levy-flight search into the SCA to enhance the local search and to avoid the local optima stagnation. In order to focus on enhancing the exploitation of solution space, SCA and PSO are hybridized (Issa et al., 2018; Nenavath, Jatoth, & Das, 2018). In this hybridization, PSO helps to provide a comparatively better direction of search as compared to the classical SCA. In Nayak, Dash, Majhi, and Wang (2018), to speed up the convergence rate and to avoid the local optima stagnation, the mutation operator is employed with random candidate solution of the population. Meshkat and Parhizgar (2017) have proposed an updated position updated

search mechanism to improve the search performance of the classical SCA. A brief literature survey on the SCA algorithm can be accessed from Mirjalili, Mirjalili, Saremi, and Mirjalili (2020).

Although, the above works have enhanced the search strategy of the algorithm, but still in some cases, these modified variants are unable to find the optima or near to the optima solution due to the insufficient ability of exploration and/or exploitation of the solution space. Therefore, as an alternative, the present paper proposes a modified version of the SCA by incorporating different strategies to enhance the level of exploration and exploitation in the algorithm.

3. Proposed new developments

Although, the SCA is very efficient to explore the new search regions of the solution space, when compared with other population-based meta-heuristic algorithms. But, in some cases, its ability to escape from local optimal are weakened, primarily due to the lack of diversity and extraction of the best available memory in terms of solution quality (Gupta & Deep, 2019a; 2019b; Nenavath et al., 2018). Based on an in-depth study of the SCA, the following modifications are introduced to improve its search performance.

3.1. Non-linear transition parameter

The SCA has a parameter r_1, which can be also called as transition parameter, plays a significant role while moving from the phase of exploration to the exploitation. Our observations of the performance of this algorithm showed that such parameter is changed linearly in the SCA, while a lot of problems requires non-linear changes in the exploratory and exploitative behaviours of an algorithm to avoid locally optimal solutions. Therefore, we proposed non-linear, time-varying control parameter. The higher values of $r_1(>1)$ facilitates the exploration, while the low values of $r_1(<1)$ facilitates to the local exploitation of the solution space. An appropriate selection of this parameter is essential to balance the global exploration and local exploitation. Since the search process of the SCA is non-linear and our aim is to spend more time on exploration phase as compared to the exploitation therefore, a non linear transition parameter r_1, which is adopted in our proposed algorithm, is given by:

$$r_1 = a \times \sin\left(\left(1 - \frac{t}{T}\right) \times \frac{\pi}{2}\right) + b \tag{3}$$

where, a and b are constants ($a = 2.0$ and $b = 0.5$).

This non-linear transition parameter focuses on exploration comparatively in more iterations as compared to the original linear parameter. The comparison of this non-linear parameter to the original linear parameter is shown in Fig. 1. Eq. (3) and Fig. 1 show that initially, the value of the proposed non-linear parameter is high which shows that it facilitates on exploration for a long time (approximately 65% of maximum number of iterations) as compared to the exploitation. The figure also shows that in the of the search process, the proposed transition parameter facilitates on exploitation only for approximately 35% iterations.

3.2. Modifying the search equation

After the use of non-linear control parameter in SCA, we observed that the accuracy of solutions obtained in the final iteration might be jeopardized due to degrading exploitation around the current elite candidate solution when using non-linear parameter. Therefore, the formulation of SCA is changed to consider the exploitation of the best solution obtained so far (the elite) and gravitate other solutions towards it. This has led to improving the exploitation and eventually the accuracy of the results. The search

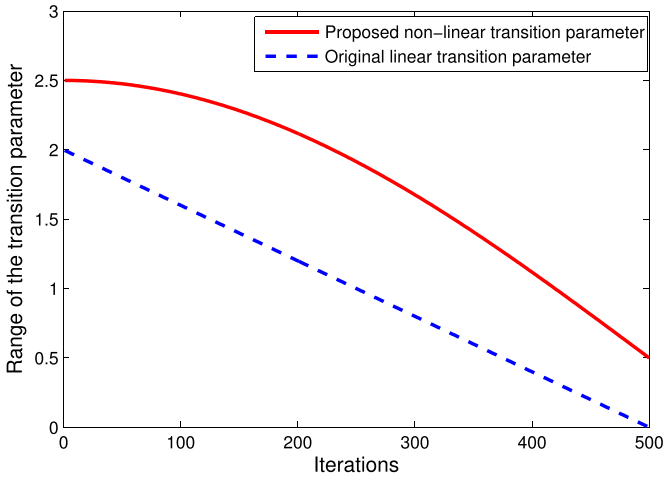

Fig. 1. Comparison of transition parameters.

equation is modified in order to provide a greedy direction of search using the leading elite guidance based search. This means that the new positions are now discovered based on the best available information about the solution space. This leading guidance helps to avoid the situation of skipping true solutions during the search. The proposed modified search equation is given by:

$$x_{i,j}^{t+1} = \begin{cases} x_{D,j} - r_1 \sin(2\pi r_2)\left|2r_3 x_{D,j} - x_{i,j}^t\right| & r_4 > 0.5 \\ x_{D,j} - r_1 \cos(2\pi r_2)\left|2r_3 x_{D,j} - x_{i,j}^t\right| & r_4 \le 0.5 \end{cases} \tag{4}$$

Although, this modified search equation helps to increase the exploitative ability of the algorithm and to extract the information available at discovered elite promising region of the search space, the exploration is maintained by the non-linear transition parameter as explained in previous section. Thus, the non-linear transition parameter and the modified position update equation tried to level up the phase of exploration and exploitation but in a balanced manner with appropriate setting of parameters.

However, these proposed strategies works well on some problems, but the algorithm still might assumed that the local optima are unavoidable in some cases. Therefore, an additional phase is added to the algorithm, when the modified search mechanism is unable to find the better position. This phase includes the mutation and generation of new solution using chaotic map (Mirjalili & Gandomi, 2017; Zhenyu, Bo, Min, & Binggang, 2006). This process of generating new solution say z_i can be understood by Eq. (5).

$$Z_i^{t+1} = \begin{cases} X_D^t \times (1+\delta) & r_5 > 0.5 \\ X_{min} + \beta \times (X_{max} - X_{min}) & r_5 \le 0.5 \end{cases} \tag{5}$$

where $X_{\min}$ and $X_{\max}$ are the lower and upper boundary limits for the candidate solution X_i, respectively. The parameter r_5 is a uniformly distributed random number from the interval (0,1). β is random number generated by the Logistic chaotic map (Zhenyu et al., 2006) which is defined as follows:

$$\beta_{k+1} = c.\beta_k \times (1 - \beta_k) \tag{6}$$

Here c is a control parameter, which is fixed to the value 4 (Zhenyu et al., 2006). The distribution of generated Logistic chaotic random numbers over 100 iterations is shown in Fig. 2

In Eq. (5), δ is a mutation operator. In our proposed approached this mutation is chosen as Gaussian mutation (Lin & Zhong, 2013). The Gaussian mutation is integrated into the MSCA with an aim of reducing the drawback of losing the diversity during the search process. The density function of Gaussian distribution is shown as follow:

$$f_{Gaussian(0,\sigma^2)}(\beta) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{\beta^2}{2\sigma^2}} \tag{7}$$

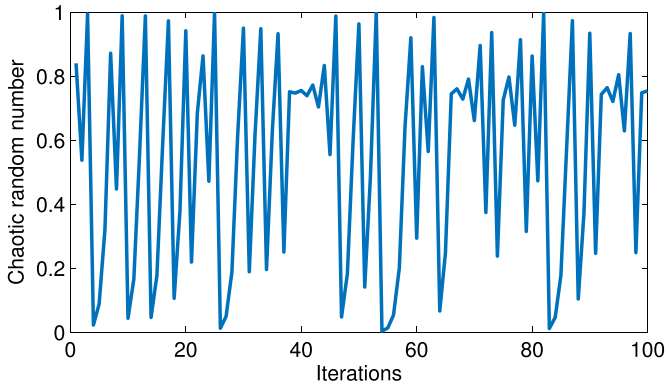

Fig. 2. Comparison of transition parameters.

where σ^2 is a variance corresponding to each candidate solution. This equation is reduced to generating a random number according to Gaussian distribution with a value of mean to 0 and standard deviation to 1. Thus, the mutation operator used to obtained a new position Z_i can be represented as:

$$Z_i^{t+1} = X_D^t \times \left(1 + G(\beta)\right) \quad if \quad r_5 > 0.5 \tag{8}$$

Thus, the implementation of the proposed MSCA based on all above proposed strategies is explained in Algorithm 2.

Algorithm 2 Pseudo-code of MSCA.

Inputs: The population size N and maximum number of iterations T
Outputs: The position of destination point and its fitness value
Initialize the population $X_i (i = 1, 2, \ldots, N)$ randomly and within the provided solution space
while (t<T) **do**
  Calculate the fitness value of each candidate solution
  Set X_D as position of the destination point (best candidate solution)
  Update the transition parameter r_1 using a new modified rule as given in eq. (3)
  for (each candidate solution (X_i)) **do**
    Obtain the new position Y_i using modified search eq. (4)
    if (fitness(Y_i) $\leq$ fitness(X_i)) **then**
      replace the position X_i by Y_i
    else
      obtain a new position Z_i using new scheme given in Eqs. (5) and (8)
      if (fitness(Z_i) $\leq$ fitness(X_i)) **then**
        replace the position X_i by Z_i
Return the destination point X_D

4. Experimental results

In order to validate the performance of the proposed MSCA, three experiments are conducted in this section.

- Experiment 1 includes the evaluation of the proposed MSCA and its comparison with classical SCA on 30-dimensional scalable and fixed-dimensional benchmark test problems. These benchmark test problems are used by several researchers to validate the performance of their algorithms (Mirjalili, 2016; Mirjalili & Lewis, 2016; Mirjalili et al., 2014). The mathematical expressions of these problems is explained in Tables 2–4. This comparison is based on several statistics such as mean, median, maximum, minimum and standard deviation of the objective functions values, which are recorded over 30 independent trials of both the algorithms classical SCA and proposed MSCA.
- Experiment 2 conducts the same experiments as performed in Experiment 1, but with the dimension size 500 of all the 23 scalable problems to observe the impact of improvements in the MSCA on the scalability of test problems.
- Experiment 3 compares the performance of the MSCA with other modified variants of the SCA which are introduced in the literature to enhance the quality of the search process in the SCA. This comparison is based on mean and standard deviation of objective function values, statistical analysis through Wilcoxon rank-sum test and convergence behaviour of algorithms. Moreover, the MSCA is also compared with some other meta-heuristic algorithms. All these comparison is performed on 30-dimensional scalable problems and on fixed-dimensional problems.

Table 1
The detailed settings of the utilized system.

Name	Setting
Hardware	
CPU	Intel Core(TM) i5 processor
Frequency	3.1GHz
RAM	4GB
Hard drive	500 GB
Software	
Operating system	Windows 10
Language	MATLAB R2014a

The experimental environment, which is used for the experiments is presented in Table 1.

4.1. Experiment 1

This section aims to compare the performance of the proposed MSCA with classical SCA to verify the robustness and reliability of the MSCA. The numerical results, which are obtained by implementing the classical SCA and MSCA on 23 scalable test problems are presented in Table 5. These results are recorded by conducting the 30 independent trials of both the algorithms with population size 30 and 500 iterations. In this table, the minimum, mean, median, maximum and standard deviation values of the objective functions are recorded. Since the test functions from F1 to F10 are unimodal functions and only one optima called global optima is present in these functions. Therefore, these test functions can be utilized to evaluate the exploitation efficiency of the candidate solutions in the algorithm. On all these unimodal test cases, the MSCA provides superior results as compared to the classical SCA in terms of all the statistics. In the problems F1 to F5, F8 and F10, the proposed MSCA is able to locate the optimal solution. The lower values of the standard deviation on all these unimodal functions show the robustness of the MSCA.

The test problems from F11 to F23 are multimodal problems and can be used to check the exploration and local optima avoidance ability of meta-heuristic algorithms, because of having massive number of optima. Out of these 13 problems, in ten problems namely, F11, F12, F14, F17-F23 the proposed MSCA has obtained the optimal solution, while the classical SCA is able to locate the optima only for F21. In the remaining problems namely, F13, F15 and F16, the proposed MSCA provides far better solution as compared to the classical SCA.

On these 13 multimodal problems, the MSCA has provided the better solutions than the classical SCA in all the statistics.

The results on the fixed-dimensional test problems are presented in Table 6. On these problems, the proposed MSCA has outperformed the classical SCA in all the test problems. In all the

Table 2
Description of unimodal benchmark functions.

Function	Dimensions	Range	f_{min}
$F1(x)=\sum_{i=1}^{n}x_i^2$	30,500	[− 100,100]	0
$F2(x)=\sum_{i=1}^{n}ix_i^2$	30,500	[− 10,10]	0
$F3(x)=\sum_{i=1}^{n}\lvert x_i\rvert+\prod_{i=1}^{n}\lvert x_i\rvert$	30,500	[− 10,10]	0
$F4(x)=\sum_{i=1}^{n}\left(\sum_{j-1}^{i}x_j\right)^2$	30,500	[− 100,100]	0
$F5(x)=\max_i\{\lvert x_i\rvert,1\le i\le n\}$	30,500	[− 100,100]	0
$F6(x)=\sum_{i=1}^{n-1}\left[100\left(x_{i+1}-x_i^2\right)^2+(x_i-1)^2\right]$	30,500	[− 30,30]	0
$F7(x)=\sum_{i=1}^{n}([x_i+0.5])^2$	30,500	[− 100,100]	0
$F8(x)=\sum_{i=1}^{n}ix_i^4$	30,500	[− 1.28,1.28]	0
$F9(x)=\sum_{i=1}^{n}ix_i^4+random[0,1)$	30,500	[− 1.28,1.28]	0
$F10(x)=\sum_{i=1}^{n}\lvert x_i\rvert^{i+1}$	30,500	[− 1,1]	0

Table 3
Description of multimodal benchmark functions.

Function	Dimensions	Range	f_{min}
$F11(x)=\sum_{i=1}^{n}-x_i\sin\left(\sqrt{\lvert x_i\rvert}\right)$	30,500	[- 500,500]	- 418.9829 $\times n$
$F12(x)=\sum_{i=1}^{n}\left[x_i^2-10\cos(2\pi x_i)+10\right]$	30,500	[- 5.12,5.12]	0
$F13(x)=-20\exp(-0.2\sqrt{\frac{1}{n}\sum_{i=1}^{n}x_i^2})-\exp\left(\frac{1}{n}\sum_{i=1}^{n}cos(2\pi x_i)\right)+20+e$	30,500	[- 32,32]	0
$F14(x)=\frac{1}{4000}\sum_{i=1}^{n}x_i^2-\prod_{i=1}^{n}\cos\left(\frac{x_i}{\sqrt{i}}\right)+1$	30,500	[- 600,600]	0
$F15(x)=\frac{\pi}{n}\left\{10\sin(\pi y_1)+\sum_{i=1}^{n-1}(y_i-1)^2\left[1+10\sin^2(\pi y_{i+1})\right]+(y_n-1)^2\right\}+$ $\sum_{i=1}^{n}u(x_i,10,100,4)y_i=1+\frac{x_i+1}{4}u(x_i,a,k,m)=\begin{cases}k(x_i-a)^m & x_i>a\\ 0-a & <x_i<a\\ k(-x_i-a)^m & x_i<-a\end{cases}$	30,500	[- 50,50]	0
$F16(x)=0.1\left\{\sin^2(3\pi x_1)+\sum_{i=1}^{n}(x_i-1)^2\left[1+\sin^2(3\pi x_i+1)\right]+(x_n-1)^2\left[1+\sin^2(2\pi x_n)\right]\right\}+$ $\sum_{i=1}^{n}u(x_i,5,100,4)$	30,500	[- 50,50]	0
$F17(x)=\sum_{i=1}^{n}\lvert x_i\sin(x_i)+0.1x_i\rvert$	30,500	[- 10,10]	0
$F18(x)=\sum_{i=1}^{n}0.1n-\left(0.1\sum_{i=1}^{n}\cos(5\pi x_i)-\sum_{i=1}^{n}x_i^2\right)$	30,500	[- 1,1]	0
$F19(x)=\sum_{i=1}^{n-1}(x_i^2+2x_{i+1}^2)^{0.25}\times[1+sin(50(x_i^2+x_{i+1}^2)^{0.1})^2$	30,500	[- 10,10]	0
$F20(x)=\sum_{i=1}^{n}(10^6)^{(i-1)/(n-1)}x_i^2$	30,500	[- 100,100]	0
$F21(x)=(-1)^{n+1}\prod_{i=1}^{n}\cos(x_i)\times\exp(-\sum_{i=1}^{n}(x_i-\pi)^2)$	30,500	[- 100,100]	0
$F22(x)=1-cos\left(2\pi\sqrt{(\sum_{i=1}^{n}x_i^2)}\right)+0.1\sqrt{(\sum_{i=1}^{n}x_i^2)}$	30,500	[- 100,100]	0
$F23(x)=0.5+\frac{sin^2\left(\sqrt{(\sum_{i=1}^{n}x_i^2)}\right)-0.5}{1+0.001\left(\sum_{i=1}^{n}x_i^2\right)^2}$	30,500	[- 100,100]	0

Table 4
Description of fixed-dimension multimodal benchmark functions.

Function	Dimensions	Range	f_{min}
$F24(x)=\left(\frac{1}{500}+\sum_{j=1}^{25}\frac{1}{j+\sum_{i=1}^{2}(x_i-a_{ij})^6}\right)^{-1}$	2	[−65, 65]	1
$F25(x)=\sum_{i=1}^{11}\left[a_i-\frac{x_1(b_i^2+b_ix_2)}{b_i^2+b_ix_3+x_4}\right]^2$	4	[−5, 5]	0.00030
$F26(x)=4x_1^2-2.1x_1^4+\frac{1}{3}x_1^6+x_1x_2-4x_2^2+4x_2^4$	2	[−5, 5]	- 1.0316
$F27(x)=\left(x_2-\frac{5.1}{4\pi^2}x_1^2+\frac{5}{\pi}x_1-6\right)^2+10\left(1-\frac{1}{8\pi}\right)\cos x_1+10$	2	[−5, 5]	0.398
$F28(x)=\left[1+(x_1+x_2+1)^2\left(19-14x_1+3x_1^2-14x_2+6x_1x_2+3x_2^2\right)\right]$ $\times\left[30+(2x_1-3x_2)^2\times\left(18-32x_1+12x_1^2+48x_2-36x_1x_2+27x_2^2\right)\right]$	2	[−2, 2]	3
$F29(x)=-\sum_{i=1}^{4}c_i\exp\left(-\sum_{j=1}^{3}a_{ij}\left(x_j-p_{ij}\right)^2\right)$	3	[1, 3]	- 3.86
$F30(x)=-\sum_{i=1}^{4}c_i\exp\left(-\sum_{j=1}^{6}a_{ij}\left(x_j-p_{ij}\right)^2\right)$	6	[0, 1]	- 3.32
$F31(x)=-\sum_{i=1}^{5}\left[(X-a_i)(X-a_i)^T+c_i\right]^{-1}$	4	[0, 10]	- 10.1532
$F32(x)=-\sum_{i=1}^{7}\left[(X-a_i)(X-a_i)^T+c_i\right]^{-1}$	4	[0, 10]	- 10.4028
$F33(x)=-\sum_{i=1}^{10}\left[(X-a_i)(X-a_i)^T+c_i\right]^{-1}$	4	[0.10]	- 10.5363

problems, either the optimal solution is obtained or the obtained solution is very close to the optimal solution. These problems have less number of local optima as compared to the multimodal problems with high dimension size and therefore, the ability of maintaining suitable balance between the exploitation and exploration can be verified in the MSCA on these test problems.

Thus, from Tables 5 and 6, it can be verified that the all the employed strategies which are discussed in Section 3 have shown their impact on improving the search mechanism of the SCA for obtaining the better solution accuracy.

In order to compare the significance of the improved results obtained by the MSCA against to the classical SCA and to verify that the obtained results are not just by a chance, a non-parametric Wilcoxon rank-sum test (Derrac, García, Molina, & Herrera, 2011) is applied at 5% significance level. The obtained statistical outcomes are presented in Tables 5 and 6 in the form of symbols '+/≈/−', to indicate that the proposed MSCA is significantly better, equal or worse than the classical SCA, respectively. By analyzing these outcomes, it can be concluded that the proposed MSCA has significantly outperform classical SCA in all the unimodal, multimodal

Table 5
Comparison of results obtained by the proposed MSCA and classical SCA on 30-dimensional benchmark set of scalable problems.

Test Function	Algorithm	min	med	mean	max	STD
F1	**SCA**	2.71E−02	2.14E+00	4.07E+00	1.86E+01	4.83E+00
	MSCA(+)	**0**	**0**	**0**	**0**	**0**
F2	**SCA**	4.25E−03	4.31E−01	1.76E+00	1.39E+01	3.13E+00
	MSCA(+)	**0**	**0**	**0**	**0**	**0**
F3	**SCA**	4.05E−05	6.61E−03	1.60E−02	1.25E−01	2.89E−02
	MSCA(+)	**0**	**0**	**0**	**0**	**0**
F4	**SCA**	8.81E+02	5.68E+03	6.75E+03	1.67E+04	3.67E+03
	MSCA(+)	**0**	**0**	**0**	**0**	**0**
F5	**SCA**	1.64E+01	3.95E+01	4.03E+01	6.72E+01	1.10E+01
	MSCA(+)	**0**	**0**	**0**	**0**	**0**
F6	**SCA**	3.53E+01	8.09E+03	5.04E+04	5.87E+05	1.18E+05
	MSCA(+)	**6.30E−07**	**8.55E−05**	**2.87E−04**	**2.75E−03**	**5.90E−04**
F7	**SCA**	4.05E+00	6.72E+00	1.10E+01	8.52E+01	1.47E+01
	MSCA(+)	**2.98E−09**	**2.06E−07**	**6.54E−07**	**4.70E−06**	**1.11E−06**
F8	**SCA**	2.81E−06	2.35E−03	8.99E−03	9.04E−02	1.94E−02
	MSCA(+)	**0**	**0**	**0**	**0**	**0**
F9	**SCA**	2.46E−02	7.34E−02	1.32E−01	4.73E−01	1.31E−01
	MSCA(+)	**6.87E−06**	**1.27E−04**	**2.07E−04**	**1.22E−03**	**2.54E−04**
F10	**SCA**	2.23E−12	6.88E−07	7.18E−05	1.29E−03	2.46E−04
	MSCA(+)	**0**	**0**	**0**	**0**	**0**
F11	**SCA**	−4.44E+03	−3.69E+03	−3.75E+03	−3.24E+03	2.96E+02
	MSCA(+)	**−1.26E+04**	**−1.26E+04**	**−1.26E+04**	**−1.26E+04**	**1.35E−03**
F12	**SCA**	1.66E−01	1.61E+01	3.02E+01	8.89E+01	2.84E+01
	MSCA(+)	**0**	**0**	**0**	**0**	**0**
F13	**SCA**	2.12E−02	2.02E+01	1.29E+01	2.03E+01	9.43E+00
	MSCA(+)	**8.88E−16**	**8.88E−16**	**8.88E−16**	**8.88E−16**	**0**
F14	**SCA**	3.81E−01	1.05E+00	1.00E+00	1.74E+00	2.96E−01
	MSCA(+)	**0**	**0**	**0**	**0**	**0**
F15	**SCA**	1.42E+00	1.36E+01	2.16E+05	1.63E+06	4.71E+05
	MSCA(+)	**1.98E−12**	**1.43E−08**	**3.27E−08**	**2.65E−07**	**5.45E−08**
F16	**SCA**	2.63E+00	7.58E+01	6.82E+04	6.42E+05	1.58E+05
	MSCA(+)	**8.47E−12**	**1.54E−07**	**1.30E−06**	**2.01E−05**	**3.73E−06**
F17	**SCA**	5.82E−03	8.46E−01	2.50E+00	2.14E+01	4.61E+00
	MSCA(+)	**0**	**0**	**0**	**0**	**0**
F18	**SCA**	1.03E−05	6.59E−04	4.72E−03	8.95E−02	1.62E−02
	MSCA(+)	**0**	**0**	**0**	**0**	**0**
F19	**SCA**	9.05E+00	3.02E+01	3.66E+01	8.20E+01	2.13E+01
	MSCA(+)	**0**	**0**	**0**	**0**	**0**
F20	**SCA**	6.44E+00	4.24E+02	2.25E+03	1.49E+04	3.86E+03
	MSCA(+)	**0**	**0**	**0**	**0**	**0**
F21	**SCA**	**0**	**0**	**0**	**0**	**0**
	MSCA(≈)	**0**	**0**	**0**	**0**	**0**
F22	**SCA**	2.00E−01	9.02E−01	1.04E+00	4.21E+00	7.47E−01
	MSCA(+)	**0**	**0**	**0**	**0**	**0**
F23	**SCA**	7.82E−02	2.30E−01	2.47E−01	4.52−01	1.03E−01
	MSCA(+)	**0**	**0**	**0**	**0**	**0**

and fixed dimensional problems except for F21. On this particular problem, both the algorithms are performed significantly same.

4.2. Experiment 2

In this section, the experiment 1 is repeated but for the 500-dimensional problems (F1-F23). The parameter setting is adopted same (i.e. 30 population size and 500 iterations) as taken in Experiment 1. The numerical results in terms of several statistics such as mean, median, minimum, maximum, and standard deviation of objective function values are recorded in Table 7. In the same table, the outcomes of the Wilcoxon rank-sum test are also shown to verify the significance of improved results.

On the unimodal test functions with the dimension of 500, the performance of the proposed MSCA is similar as for dimension 30, while the performance of the classical SCA has degraded for all unimodal problems. On the test problems F1–F5, F8 and F10, the MSCA is able to provide the optimal solution. Thus, by analyzing this behaviour of improvements, it can be suggested that the MSCA is a better optimizer than the classical SCA for solving the large dimensional unimodal optimization problems.

Similarly, the table verifies the superior performance of the MSCA against the classical SCA for solving the multimodal problems with a large dimension size. The results are better in the MSCA as compared to the classical SCA in all the statistics, which demonstrate the reliability of the proposed MSCA. On the test functions F11, F12, F14 and F17–F23, the MSCA successfully finds the optimal solution. In the remaining problems (F13, F15 and F16), the results are very impressive obtained by the MSCA and far better than the classical SCA. Thus, the experiments on large dimensional size problems verify the superior search ability of the MSCA in avoiding the local optima during the search, because in these multimodal problems, massive number of local optima exists and increases exponentially with the increase of dimension size.

4.3. Experiment 3

In this section, the performance of the MSCA is compared with the modified variants of the SCA, which are proposed in the literature for enhancing its search efficiency, and with some other meta-heuristic algorithms.

Table 6
Comparison of results obtained by the proposed MSCA and classical SCA on fixed-dimensional benchmark problems.

Test function	Algorithm	min	med	mean	max	STD
F24	**SCA**	9.98E−01	1.00E+00	1.80E+00	2.98E+00	9.84E−01
	MSCA(+)	**9.98E−01**	**9.98E−01**	**9.98E−01**	**9.98E−01**	**2.28E−10**
F25	**SCA**	3.44E−04	7.99E−04	9.49E−04	1.59E−03	3.50E−04
	MSCA(+)	**3.08E−04**	**3.24E−04**	**4.08E−04**	**1.28E−03**	**2.07E−04**
F26	**SCA**	−1.03E+00	−1.03E+00	−1.03E+00	−1.03E+00	4.38E−05
	MSCA(+)	**−1.03E+00**	**−1.03E+00**	**−1.03E+00**	**−1.03E+00**	**2.03E−06**
F27	**SCA**	3.98E−01	**3.99E−01**	**3.99E−01**	4.03E−01	1.34E−03
	MSCA(+)	**3.98E−01**	3.98E−01	3.98E−01	**3.99E−01**	**1.34E−04**
F28	**SCA**	3.00E+00	3.00E+00	3.00E+00	3.00E+00	3.74E−04
	MSCA(+)	**3.00E+00**	**3.00E+00**	**3.00E+00**	**3.00E+00**	**1.55E−05**
F29	**SCA**	−3.86E+00	−3.85E+00	−3.85E+00	−3.85E+00	3.31E−03
	MSCA(+)	**−3.86E+00**	**−3.85E+00**	**−3.86E+00**	**−3.85E+00**	**3.20E−03**
F30	**SCA**	−3.22E+00	−3.01E+00	−2.95E+00	−1.91E+00	2.79E−01
	MSCA(+)	**−3.32E+00**	**−3.13E+00**	**−3.12E+00**	**−2.64E+00**	**1.57E−01**
F31	**SCA**	−5.89E+00	−1.93E+00	−2.21E+00	−4.97E−01	1.66E+00
	MSCA(+)	**−1.02E+01**	**−1.02E+01**	**−1.02E+01**	**−1.02E+01**	**8.90E−05**
F32	**SCA**	−5.11E+00	−3.78E+00	−3.14E+00	−5.21E−01	1.55E+00
	MSCA(+)	**−1.04E+01**	**−1.04E+01**	**−1.04E+01**	**−1.04E+01**	**8.77E−05**
F33	**SCA**	−7.17E+00	−3.18E+00	−3.27E+00	−9.43E−01	1.76E+00
	MSCA(+)	**−1.05E+01**	**−1.05E+01**	**−1.05E+01**	**−1.05E+01**	**7.72E−05**

Table 7
Comparison of results obtained by the proposed MSCA and classical SCA on 500-dimensional benchmark set of scalable problems.

Test function	Algorithm	min	med	mean	max	STD
F1	**SCA**	6.79E+04	1.70E+05	1.78E+05	3.33E+05	6.94E+04
	MSCA(+)	**0**	**0**	**0**	**0**	**0**
F2	**SCA**	1.17E+05	4.49E+05	4.51E+05	7.95E+05	1.66E+05
	MSCA(+)	**0**	**0**	**0**	**0**	**0**
F3	**SCA**	2.62E+01	9.08E+01	9.26E+01	1.91E+02	4.29E+01
	MSCA(+)	**0**	**0**	**0**	**0**	**0**
F4	**SCA**	4.88E+06	6.34E+06	6.70E+06	9.70E+06	1.11E+06
	MSCA(+)	**0**	**0**	**0**	**0**	**0**
F5	**SCA**	9.82E+01	9.91E+01	9.90E+01	9.95E+01	2.93E−01
	MSCA(+)	**0**	**0**	**0**	**0**	**0**
F6	**SCA**	1.14E+09	1.82E+09	1.83E+09	2.71E+09	4.28E+08
	MSCA(+)	**1.33E−06**	**7.58E−04**	**2.04E−03**	**1.14E−02**	**3.07E−03**
F7	**SCA**	9.93E+04	1.89E+05	1.93E+05	3.16E+05	5.64E+04
	MSCA(+)	**2.01E−08**	**3.38E−06**	**5.75E−06**	**3.84E−05**	**8.10E−06**
F8	**SCA**	7.41E+03	1.36E+04	1.35E+04	2.18E+04	3.81E+03
	MSCA(+)	**0**	**0**	**0**	**0**	**0**
F9	**SCA**	8.66E+03	1.59E+04	1.58E+04	2.45E+04	3.83E+03
	MSCA(+)	**6.35E−07**	**1.08E−04**	**1.46E−04**	**6.11E−04**	**1.41E−04**
F10	**SCA**	3.91E−01	2.02E+00	1.93E+00	2.72E+00	4.89E−01
	MSCA(+)	**0**	**0**	**0**	**0**	**0**
F11	**SCA**	−1.87E+04	−1.59E+04	−1.58E+04	−1.36E+04	1.28E+03
	MSCA(+)	**−2.09E+05**	**−2.09E+05**	**−2.09E+05**	**−2.09E+05**	**2.33E−02**
F12	**SCA**	4.01E+02	1.09E+03	1.15E+03	2.86E+03	5.07E+02
	MSCA(+)	**0**	**0**	**0**	**0**	**0**
F13	**SCA**	9.47E+00	2.08E+01	1.95E+01	2.08E+01	3.48E+00
	MSCA(+)	**8.88E−16**	**8.88E−16**	**8.88E−16**	**8.88E−16**	**0**
F14	**SCA**	8.02E+02	1.80E+03	1.86E+03	3.54E+03	5.91E+02
	MSCA(+)	**0**	**0**	**0**	**0**	**0**
F15	**SCA**	3.53E+09	5.83E+09	5.63E+09	7.74E+09	1.17E+09
	MSCA(+)	**5.34E−11**	**6.89E−09**	**2.66E−08**	**3.26E−07**	**6.33E−08**
F16	**SCA**	6.65E+09	9.97E+09	1.02E+10	1.41E+10	2.21E+09
	MSCA(+)	**2.73E−11**	**1.90E−07**	**1.82E−06**	**2.13E−05**	**4.24E−06**
F17	**SCA**	4.52E+01	1.52E+02	1.54E+02	2.68E+02	5.13E+01
	MSCA(+)	**0**	**0**	**1.11E−242**	**3.32E−241**	**0**
F18	**SCA**	2.00E+01	4.42E+01	4.09E+01	6.53E+01	1.13E+01
	MSCA(+)	**0**	**0**	**0**	**0**	**0**
F19	**SCA**	4.88E+03	8.88E+04	2.01E+08	5.47E+09	9.99E+08
	MSCA(+)	**0**	**0**	**0**	**0**	**0**
F20	**SCA**	6.56E+08	3.38E+09	3.87E+09	8.49E+09	1.88E+09
	MSCA(+)	**0**	**0**	**0**	**0**	**0**
F21	**SCA**	**0**	**0**	**0**	**0**	**0**
	MSCA(≈)	**0**	**0**	**0**	**0**	**0**
F22	**SCA**	2.81E+01	4.61E+01	4.54E+01	5.50E+01	7.50E+00
	MSCA(+)	**0**	**0**	**0**	**0**	**0**
F23	**SCA**	5.00E−01	5.00E−01	5.00E−01	5.00E−01	2.12E−06
	MSCA(+)	**0**	**0**	**0**	**0**	**0**

4.3.1. Performance comparison with modified variants of the SCA

In the literature, several attempts have been conducted to enhance the search efficiency of the SCA. In this section the following versions of the SCA are implemented with the same parameter setting as considered in their original papers.

m-SCA – this modified version of the SCA is introduced in Gupta and Deep (2019a) with aim to speed up the convergence rate, reduce the diversity and to provide a suitable move which is more efficient search direction and based on the existing information of promising regions of the solution space. This algorithms utilizes the concept of opposition-based learning and integrates a cognitive component into the search equation.

ISCA – this version of the SCA (Gupta & Deep, 2019b) is developed to enhance the exploitation skills of candidate solutions using the personal best guidance of search and crossover operator. The crossover and the personal best guidance are employed in the search mechanism to balance the local and global search abilities.

modSCA – this algorithm was developed with combining the mutation operator for a randomly selected candidate solution of the population during the search procedure (Nayak et al., 2018). This mutation is introduced to increase the diversity of the algorithm.

OBSCA – this algorithm (Elaziz et al., 2017) was developed to speed up the convergence rate and for a better exploration of the solution space. To accomplish this, an opposition-based learning (OBL) concept is used for the initialization phase as well as for iterative process of the algorithm.

SCA-GWO – this algorithm (Singh & Singh, 2017) was proposed to increase the exploitation strength of the SCA using GWO. The exploration ability of the SCA is used to provide a global search and the exploitation ability of the GWO increases the search around the best solutions of the population.

SCA-PSO – this algorithm (Nenavath et al., 2018) was developed from the inspiration of exploitation ability of the PSO. In the proposed SCA-PSO, the personal best state of each candidate solution along with the global best solution are used to guide the search procedure.

The obtained results by all of these algorithms with same parameter setting (i.e. 30 population size and 500 iterations) are presented in Tables 8 and 9. It can be observed from these tables that the solutions of the MSCA are either much better or very competitive than the m-SCA, ISCA, modSCA, OBSCA, SCA-GWO and SCA-PSO.

Figs. 3 and 4 show a comparison between the proposed MSCA and classical SCA for the convergence rate of the best obtained solution during the intermediate iterations of the algorithms. These figures also compared the convergence rate of the modified variants of the SCA such as modSCA, m-SCA and ISCA. From the figures, it can be analyzed that the MSCA outperforms all other algorithms in terms of the convergence speed with a comparatively better convergence behavior towards the optimal solution.

In order to study and analyze the distributional characteristics of the experiments performed on the classical benchmark problems by the MSCA, a box plot analysis is conducted and a few of the boxplots for some of the benchmark functions are shown in Fig. 5. These boxplots demonstrate the distribution of the obtained results over 30 independent trials of algorithms. In this figure, a comparison is performed between the MSCA, classical SCA and its modified variants. The box plots can be studied based on the inter quartile range (IQR), which can be measured from the minimum and maximum point of the box. In the remaining most of the problems (F2–F5, F8, F10, F12, F14, F17–F23), which are not presented in the figure, the boxplot of the MSCA is same as shown in F1, because all the statistics such as mean, median, minimum and maximum of the obtained objective function values are same and equal to the optimal solution. Also the standard deviation is zero for those problems. Therefore, for those functions, obviously, the boxplots will show the superiority of the distribution of the results and robustness of the MSCA as compared to all other algorithms. This observation from the figure clearly indicates the better solution quality obtained by the MSCA as compared to the classical SCA and other variants of SCA.

4.3.2. Performance comparison with other meta-heuristic algorithms

This section compares the performance of the proposed MSCA with some other meta-heuristic algorithms namely, basic PSO (PSO) (Eberhart & Kennedy, 1995), Inertia weighted PSO (wPSO) (Shi & Eberhart, 1998), Grey Wolf Optimizer (GWO) (Mirjalili et al., 2014), Salp Swarm Algorithm (SSA) (Mirjalili et al., 2017), Teaching-learning Based Optimization (TLBO) (Rao et al., 2011), Covariance Matrix Adaptation Evolution Strategy (CMA-ES) (Hansen & Ostermeier, 2001), Sinusoidal Differential Evolution (SinDE) (Draa, Bouzoubia, & Boukhalfa, 2015), Improved Differential Evolution (IDE) (Hongfeng, 2019) and Firefly Algorithm (FA) (Yang, 2010). For the comparison, 30-dimensional scalable benchmark problems and fixed-dimensional benchmark problems are utilized, which are shown in Tables 2–4. The results are recorded in Tables 10 and 11, by fixing the population size as 30 and maximum iterations as 500 for each algorithm. In this table, the mean and standard deviation value of the objective functions are presented. The best values among all the results are highlighted in bold face. From the table it can be easily verified that for the unimodal problems (F1–F10) the proposed MSCA has outperformed all other algorithms (except for SSA in problem F7). The lower values of the standard deviation also verify the robustness and better efficiency of search by the MSCA. Similarly, for multimodal problems (F11 to F23), the MSCA is superior to all other meta-heuristic algorithms. It can be also seen from Table 10 that the MSCA provides the optimal solution in 17 problems out of 23 problems, this also shows that the MSCA has better search strategy to find the optima of the optimization problems. The comparison on the fixed-dimensional multimodal problems also shows that the proposed MSCA is very competitive to other meta-heuristic algorithms. In both Tables 10 and 11, the statistical outcomes (S.O.) are also shown, which are obtained by applying the Wilcoxon rank-sum test (Derrac et al., 2011) at 5% significance level. In the tables, the symbols '+/≈/−' are used to indicate that the MSCA is either significantly better, equal or worse than its competitor algorithm. These statistical outcomes verify that the proposed MSCA is significantly perform better than other meta-heuristic algorithms in most of the test cases.

Based on all the comparisons conducted in Experiment 3, the algorithms can be ranked based on their average value of objective function corresponding to each test problem. For this all the algorithms are sorted as per their objective function values and then they are ranked to observe the best performer algorithm. This visualization of rank is presented in Table 12, where it can be observed that on most of the test cases, the proposed MSCA has obtained rank 1. The average and overall rank show that the proposed MSCA has ranked one, whereas the TLBO, ISCA, SCA-GWO, GWO, CMA-ES, SinDE, SCA-PSO, OBSCA, m-SCA, IDE, wPSO, SSA, classical SCA, modSCA, FA and PSO are the next methods, respectively.

4.4. Engineering design problems

In engineering and industry, there are several cases that are tackled with mathematical models and optimization algorithms. In this section, the proposed MSCA is applied to solve four engineering design test problems: gear train design, frequency-modulated design, speed reducer design and pressure vessel design. In some of the problems, constraints are involved and therefore to handle such constraints, a simple mechanism is used. In this mechanism, the best solutions from the population are selection based on the

Table 8
Comparison of the performance with modified variants of SCA on 30-dimensional scalable problems.

	m-SCA			ISCA			modSCA			SCA-GWO			SCA-PSO			OBSCA			MSCA	
Test function	mean	STD		mean	STD		mean	STD		mean	STD		mean	STD		mean	STD		mean	STD
F1	4.58E−03	2.35E−02	+	1.36E−27	1.89E−27	+	3.73E+01	5.67E+01	+	6.29E−32	8.65E−32	+	4.79E−11	1.22E−10	+	4.79E−20	1.80E−19	+	**0**	**0**
F2	4.56E−04	1.69E−03	+	2.50E−28	5.14E−28	+	2.03E+00	5.94E+00	+	1.27E−32	2.94E−32	+	3.98E−12	5.07E−12	+	8.19E−19	4.48E−18	+	**0**	**0**
F3	6.58E−04	2.54E−03	+	1.28E−17	1.48E−17	+	2.20E−02	3.13E−02	+	8.61E−20	9.78E−20	+	3.96E−08	3.92E−08	+	8.80E−16	4.65E−15	+	**0**	**0**
F4	9.90E+02	9.91E+02	+	9.11E−01	1.34E+00	+	8.28E+03	6.25E+03	+	2.88E−04	1.08E−03	+	8.23E+01	5.56E+01	+	2.17E+00	6.30E+00	+	**0**	**0**
F5	3.87E−01	4.07E−01	+	4.61E−07	4.72E−07	+	3.59E+01	8.77E+00	+	2.27E−06	1.90E−06	+	1.16E+00	3.80E−01	+	5.21E−01	1.06E+00	+	**0**	**0**
F6	2.89E+01	7.99E−01	+	2.67E+01	4.53E−01	+	1.69E+05	4.12E+05	+	2.71E+01	8.04E−01	+	3.27E+01	2.32E+01	+	2.83E+01	3.07E−01	+	**2.87E−04**	**5.90E−04**
F7	1.19E+00	4.17E−01	+	3.41E−01	2.31E−01	+	2.35E+01	2.74E+01	+	7.01E−01	3.12E−01	+	2.77E−04	1.32E−04	+	4.63E+00	2.15E−01	+	**6.54E−07**	**1.11E−06**
F8	2.26E−10	8.70E−10	+	2.37E−47	7.73E−47	+	1.42E−02	3.15E−02	+	2.93E−55	1.26E−54	+	9.58E−19	3.76E−18	+	1.23E−23	6.76E−23	+	**0**	**0**
F9	1.50E−02	6.98E−03	+	1.56E−03	7.65E−04	+	1.29E−01	1.28E−01	+	2.71E−03	1.55E−03	+	3.13E−02	9.24E−03	+	2.63E−03	1.32E−03	+	**2.07E−04**	**2.54E−04**
F10	9.95E−18	5.15E−17	+	3.51E−88	1.90E−87	+	2.08E−04	5.53E−04	+	2.33E−105	7.09E−105	+	3.69E−36	1.52E−35	+	2.95E−21	1.62E−20	+	**0**	**0**
F11	−4.30E+03	3.40E+02	+	−6.88E+03	5.22E+02	+	−3.77E+03	3.12E+02	+	−5.77E+03	7.58E+02	+	−7.28E+03	6.87E+02	+	−3.77E+03	2.97E+02	+	**−1.26E+04**	**1.35E−03**
F12	4.72E+01	4.83E+01	+	1.73E−01	9.46E−01	≈	4.92E+01	5.64E+01	+	1.12E+01	1.10E+01	+	2.25E+01	8.12E+00	+	1.31E−05	7.19E−05	+	**0**	**0**
F13	1.83E−03	4.13E−03	+	2.23E−14	8.47E−15	+	1.46E+01	8.83E+00	+	1.55E−14	4.97E−15	+	1.35E+00	5.13E+00	+	1.40E+01	9.35E+00	+	**8.88E−16**	**0.00E+00**
F14	3.75E−02	6.05E−02	+	1.97E−13	8.65E−13	≈	9.86E−01	4.18E−01	+	6.77E−03	9.64E−03	+	1.47E−02	1.38E−02	+	9.61E−11	3.45E−10	+	**0**	**0**
F15	2.03E−01	1.95E−01	+	1.35E−02	9.58E−03	+	7.18E+04	2.91E+05	+	5.48E−02	2.88E−02	+	3.47E−03	1.89E−02	+	5.13E−01	1.44E−01	+	**3.27E−08**	**5.45E−08**
F16	1.41E+00	4.63E−01	+	3.03E−01	1.55E−01	+	1.56E+05	4.70E+05	+	7.66E−01	2.69E−01	+	9.32E−03	3.19E−02	+	2.47E+00	1.58E−01	+	**1.30E−06**	**3.73E−06**
F17	2.41E−01	1.01E+00	+	8.12E−06	4.36E−05	+	6.34E+00	1.01E+01	+	7.45E−04	1.22E−03	+	3.99E−03	6.12E−03	+	1.88E−10	5.59E−10	+	**0**	**0**
F18	6.80E−07	2.21E−06	+	**0**	**0**	≈	1.28E−03	2.32E−03	+	0.00E+00	0.00E+00	+	1.32E−13	2.15E−13	+	**0**	**0**	≈	**0**	**0**
F19	5.06E+01	1.61E+01	+	2.31E−01	2.35E−01	+	2.99E+01	1.75E+01	+	3.24E−08	6.04E−08	+	3.70E+00	1.49E+00	+	3.47E+01	2.54E+01	+	**0**	**0**
F20	1.75E−01	6.13E−01	+	5.36E−24	1.10E−23	+	1.61E+03	2.90E+03	+	3.11E−28	4.74E−28	+	1.59E−08	2.61E−08	+	3.19E−17	1.30E−16	+	**0**	**0**
F21	**0**	**0**	≈	**0**	**0**	≈	**0**	**0**	≈	**0**	**0**	≈	**0**	**0**	≈	**0**	**0**	s≈	**0**	**0**
F22	3.31E−01	1.11E−01	+	1.57E−01	5.04E−02	+	1.35E+00	6.01E−01	+	1.87E−01	4.34E−02	+	5.74E−01	6.45E−02	+	1.20E−01	4.07E−02	+	**0**	**0**
F23	7.06E−02	2.86E−02	+	2.99E−02	1.24E−02	+	2.73E−01	8.64E−02	+	4.09E−02	1.64E−02	+	1.16E−01	2.48E−02	+	1.98E−02	1.35E−02	+	**0**	**0**

Table 9
Comparison of the performance with modified variants of SCA on fixed-dimensional problem.

Test function	m-SCA			ISCA			modSCA			SCA-GWO			SCA-PSO			OBSCA			MSCA	
	mean	STD	S.O.	mean	STD	S.O.	mean	STD	S.O.	mean	STD	S.O.	mean	STD	S.O.	mean	STD	S.O.	mean	STD
F24	1.02E+00	1.02E−01	+	9.98E−01	8.63E−11	≈	1.60E+00	9.22E−01	+	4.52E+00	4.23E+00	+	1.46E+00	1.09E+00	+	3.20E+00	3.35E+00	+	**9.98E−01**	**2.28E−10**
F25	5.32E−04	**9.29E−05**	+	5.89E−04	1.79E−04	+	9.05E−04	3.27E−04	+	1.72E−03	5.07E−03	+	1.32E−03	3.61E−03	+	9.00E−04	2.04E−04	+	**4.08E−04**	2.07E−04
F26	−1.03E+00	9.85E−07	+	−1.03E+00	1.70E−08	−	−1.03E+00	6.83E−05	+	−1.03E+00	7.09E−09	≈	**−1.03E+00**	**4.65E−16**	−	−1.03E+00	7.61E−07	+	−1.03E+00	2.03E−06
F27	3.98E−01	1.08E−05	≈	3.98E−01	7.84E−07	≈	3.99E−01	1.27E−03	+	3.98E−01	2.24E−07	≈	**3.98E−01**	**3.24E−16**	−	4.00E−01	2.12E−03	+	3.98E−01	1.34E−04
F28	3.00E+00	6.58E−05	+	3.00E+00	5.58E−08	−	3.00E+00	2.15E−04	+	3.00E+00	2.13E−05	≈	**3.00E+00**	**7.61E−15**	−	3.00E+00	1.70E−03	+	3.00E+00	1.55E−05
F29	−3.86E+00	1.82E−04	≈	−3.86E+00	2.98E−06	−	−3.86E+00	3.56E−03	≈	−3.86E+00	2.55E−03	≈	**−3.86E+00**	**1.30E−13**	≈	−3.85E+00	8.51E−03	≈	−3.86E+00	3.20E−03
F30	**−3.32E+00**	**1.87E−03**	−	−3.29E+00	5.35E−02	−	−3.05E+00	8.19E−02	+	−3.27E+00	7.06E−02	−	−3.27E+00	6.03E−02	−	−3.08E+00	4.54E−02	+	−3.12E+00	1.57E−01
F31	−9.89E+00	2.41E−01	+	−9.80E+00	1.29E+00	+	−4.55E+00	1.04E+00	+	−8.98E+00	2.69E+00	+	−6.38E+00	3.07E+00	+	−9.97E+00	9.86E−02	+	**−1.02E+01**	**8.90E−05**
F32	−1.02E+01	1.53E−01	+	−1.02E+01	9.68E−01	+	−4.52E+00	1.26E+00	+	−8.93E+00	2.77E+00	+	−8.14E+00	3.10E+00	+	−1.02E+01	1.15E−01	+	**−1.04E+01**	**8.77E−05**
F33	−1.03E+01	2.33E−01	+	−1.03E+01	9.86E−01	+	−4.38E+00	1.61E+00	+	−9.88E+00	2.06E+00	+	−9.03E+00	2.84E+00	+	−1.04E+01	8.68E−02	+	**−1.05E+01**	**7.72E−05**

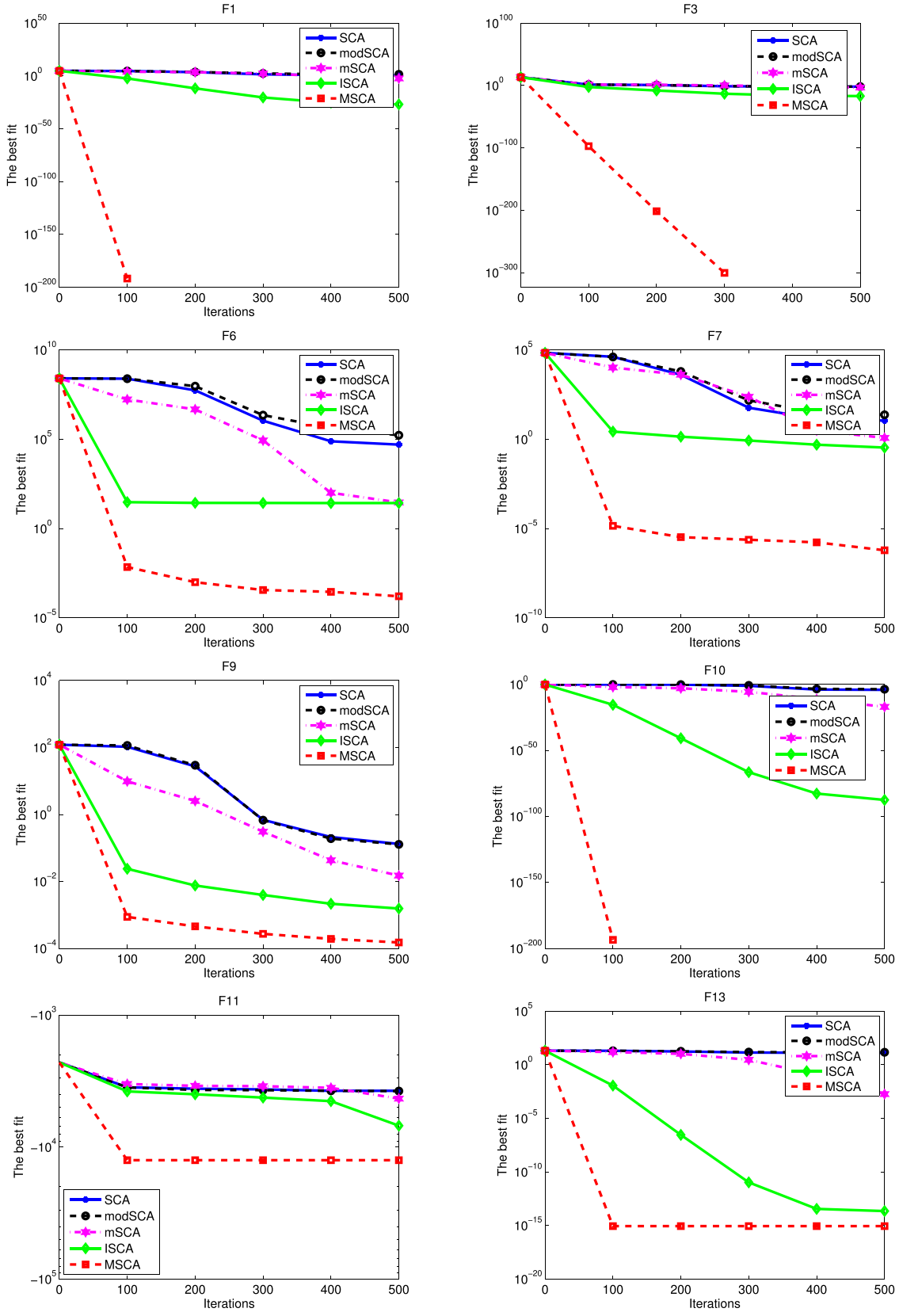

Fig. 3. Convergence curves.

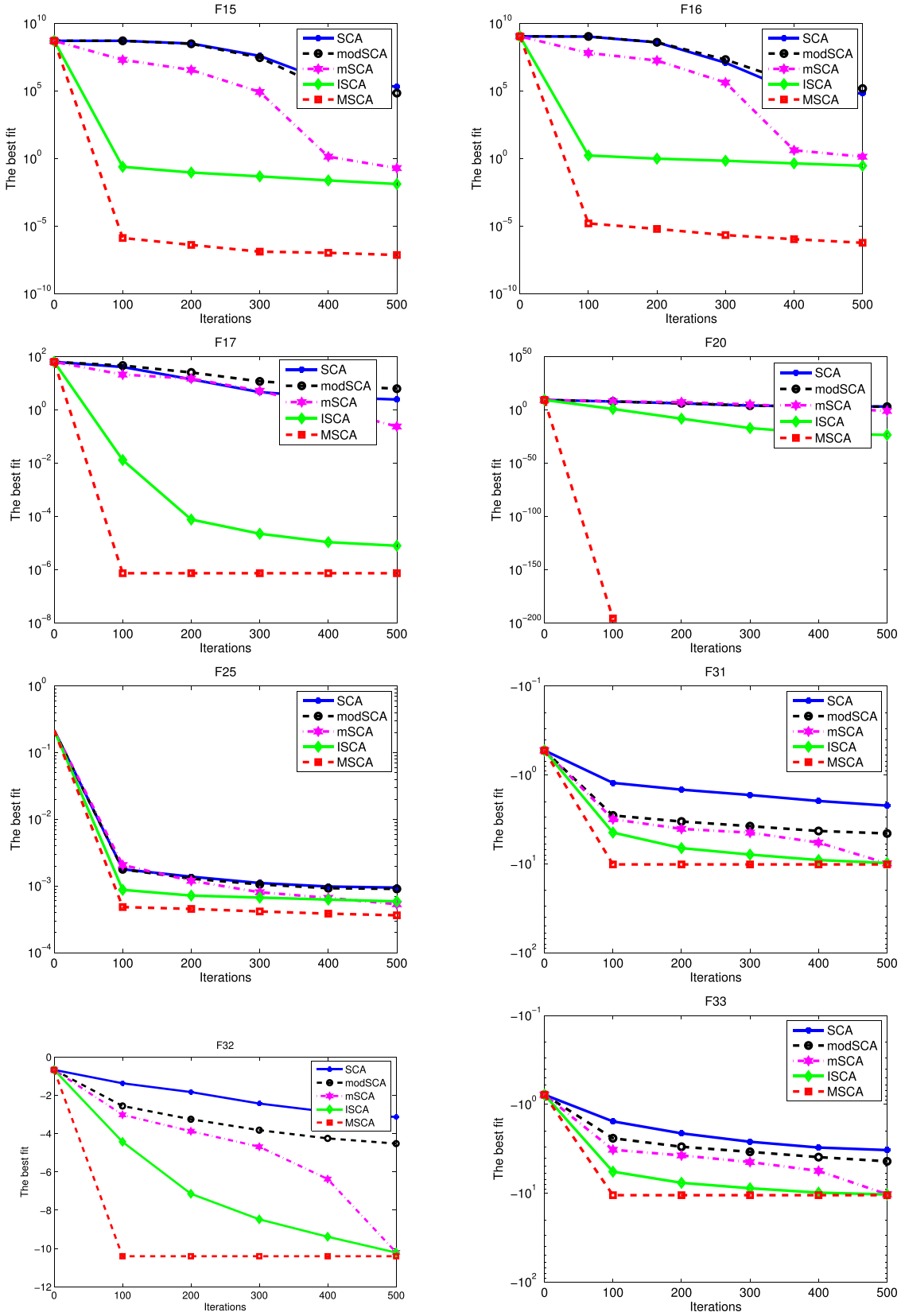

Fig. 4. Convergence curves.

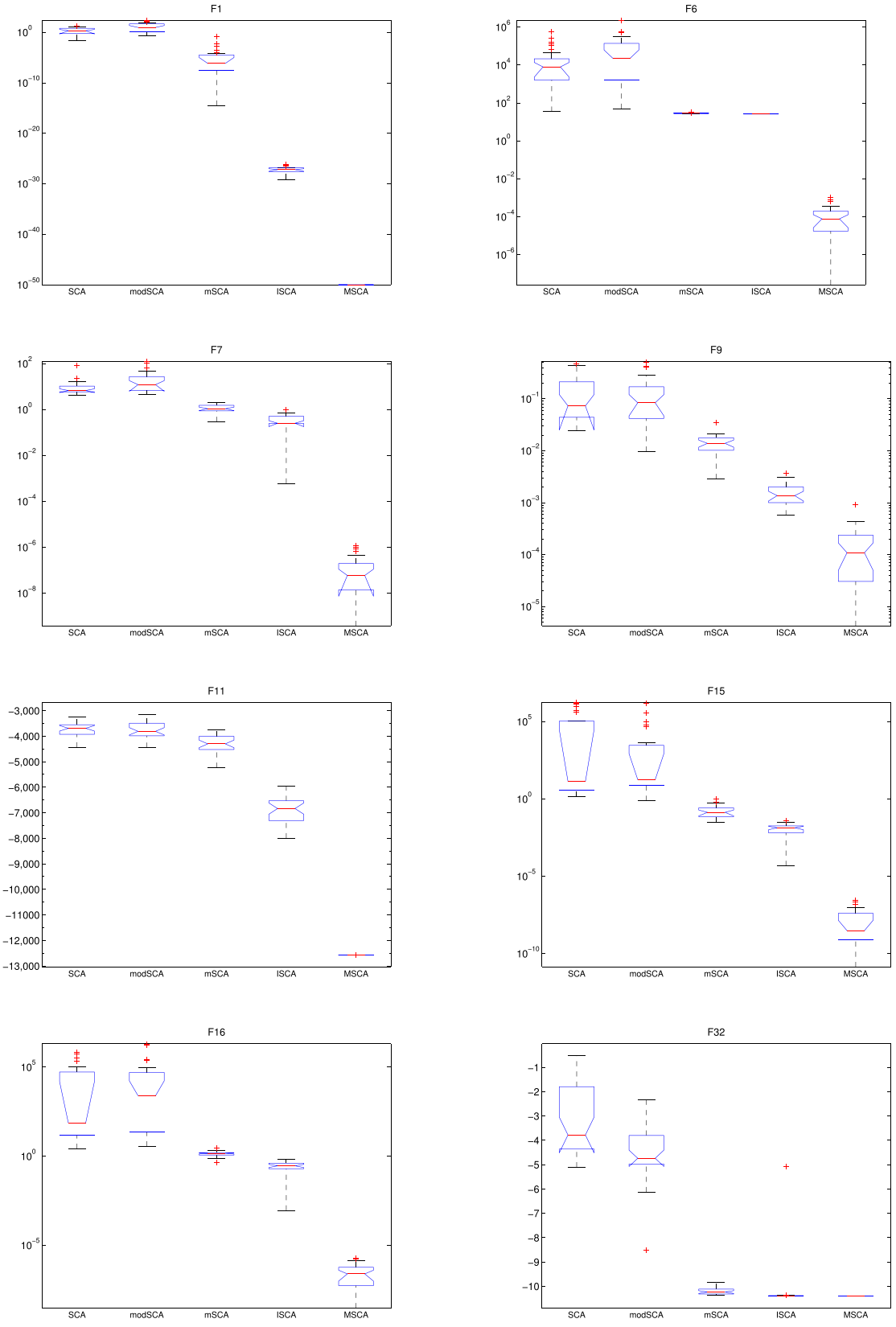

Fig. 5. Box-plots for some selected benchmark functions.

Table 10
Comparison with other meta-heuristic algorithms on 30-dimensional scalable problems.

Test function	PSO			wPSO			GWO			SSA			TLBO			CMA-ES			SinDE			IDE			FA			MSCA	
	mean	STD	S.O.	mean	STD	S.O.	mean	STD	S.O.	mean	STD	S.O.	mean	STD	S.O.	mean	STD	S.O.	mean	STD	S.O.	mean	STD	S.O.	mean	STD	S.O.		
F1	6.14E+03	6.74E+02	+	3.46E−03	5.42E−03	+	4.67E−31	1.02E−30	+	1.64E−07	1.78E−07	+	1.10E−81	3.16E−81	+	1.47E−23	1.29E−23	+	7.08E−05	2.21E−04	+	7.55E−03	3.40E−03	+	6.83E+01	2.13E+02	+	**0**	**0**
F2	9.01E+02	1.19E+02	+	2.30E−04	3.63E−04	+	1.11E−31	2.40E−31	+	2.86E+00	3.10E+00	+	5.51E−83	8.15E−83	+	4.02E−19	3.65E−19	+	4.07E−06	9.10E−06	+	8.00E−04	3.77E−04	+	1.95E+01	1.88E+01	+	**0**	**0**
F3	3.39E+01	2.88E+00	+	3.37E−01	1.83E+00	+	8.59E−19	5.23E−19	+	1.97E+00	1.55E+00	+	1.69E−41	2.13E−41	+	7.01E−11	3.73E−11	+	2.38E−04	1.51E−04	+	1.34E−02	3.29E−03	+	2.77E+01	1.40E+01	+	**0**	**0**
F4	1.46E+04	2.51E+03	+	1.69E+03	1.30E+03	+	7.96E−05	1.40E−04	+	1.37E+03	6.83E+02	+	2.72E−29	6.88E−29	+	8.71E−10	1.77E−09	+	8.03E+03	1.72E+03	+	3.95E+04	4.56E+03	+	4.36E+03	2.38E+03	+	**0**	**0**
F5	3.12E+01	1.96E+00	+	7.18E+00	1.29E+00	+	3.21E−06	2.81E−06	+	1.23E+01	3.97E+00	+	5.07E−35	4.37E−35	+	6.42E−09	3.94E−09	+	9.11E+00	3.71E+00	+	9.21E+00	1.12E+00	+	1.42E+01	3.60E+00	+	**0**	**0**
F6	2.60E+06	6.85E+05	+	3.10E+03	1.64E+04	+	2.70E+01	7.90E−01	+	1.75E+02	2.66E+02	+	2.39E+01	5.92E−01	+	1.83E+01	2.29E+00	+	7.54E+01	8.90E+01	+	2.13E+02	9.59E+01	+	3.28E+03	6.53E+03	+	**2.87E−04**	**5.90E−04**
F7	6.06E+03	8.68E+02	+	2.83E−03	4.39E−03	+	6.80E−01	3.49E−01	+	**1.59E−07**	**1.98E−07**	≈	8.18E−07	3.78E−06	+	9.73E−23	1.22E−22	−	6.24E−05	1.37E−04	+	6.04E−03	2.56E−03	+	2.93E+01	5.17E+01	+	6.54E−07	1.11E−06
F8	1.30E+00	3.29E−01	+	3.36E−09	3.66E−09	+	9.19E−55	4.36E−54	+	1.32E−10	5.27E−10	+	2.67E−161	7.28E−161	+	1.12E−40	3.43E−40	+	7.90E−09	3.31E−08	+	3.79E−09	2.82E−09	+	2.83E−03	5.85E−03	+	**0**	**0**
F9	1.73E+00	3.30E−01	+	4.81E−02	1.60E−02	+	2.63E−03	1.46E−03	+	1.96E−01	7.69E−02	+	2.48E−03	1.79E−03	+	5.06E−01	2.76E−01	+	3.39E−02	1.04E−02	+	6.29E−02	1.41E−02	+	2.76E−01	1.27E−01	+	**2.07E−04**	**2.54E−04**
F10	4.94E−04	3.12E−04	+	3.94E−19	1.74E−18	+	7.50E−102	3.84E−101	+	2.04E−06	2.10E−06	+	2.01E−212	0.00E+00	+	5.24E−09	3.68E−09	+	1.04E−12	5.67E−12	+	3.55E−17	9.12E−17	+	2.38E−06	1.43E−06	+	**0**	**0**
F11	−4.56E+03	3.63E+02	+	−8.59E+03	6.68E+02	+	−6.17E+03	8.31E+02	+	−7.51E+03	7.49E+02	+	−7.65E+03	8.11E+02	+	−1.18E+02	4.11E−14	+	−8.73E+03	4.95E+02	+	−7.89E+03	3.11E+02	+	−6.61E+03	1.24E+03	+	**−1.26E+04**	**1.35E−03**
F12	2.30E+02	1.42E+01	+	5.47E+01	1.21E+01	+	8.28E+00	6.66E+00	+	5.75E+01	1.52E+01	+	2.03E+01	1.10E+01	+	1.55E+01	3.25E+00	+	4.98E+01	1.06E+01	+	1.23E+02	1.02E+01	+	8.36E+01	2.52E+01	+	**0**	**0**
F13	1.39E+01	4.34E−01	+	4.29E−01	6.31E−01	+	1.78E−14	4.03E−15	+	2.69E+00	9.83E−01	+	4.44E−15	0.00E+00	+	2.62E−12	1.01E−12	+	1.77E−03	2.04E−03	+	2.48E−02	5.06E−03	+	6.50E+00	2.27E+00	+	**8.88E−16**	**0.00E+00**
F14	5.82E+01	6.80E+00	+	2.05E−02	1.89E−02	+	2.60E−03	6.04E−03	+	2.00E−02	1.26E−02	+	**0**	**0**	≈	**0**	**0**	≈	2.90E−03	5.94E−03	+	5.65E−02	4.80E−02	+	1.10E+00	2.14E+00	+	**0**	**0**
F15	1.64E+05	1.51E+05	+	9.97E−02	2.09E−01	+	3.87E−02	1.16E−02	+	7.96E+00	3.12E+00	+	5.28E−09	1.22E−08	+	3.46E−03	1.89E−02	+	8.64E−04	4.24E−03	+	5.13E−03	3.78E−03	+	5.85E+00	2.10E+00	+	**3.27E−08**	**5.45E−08**
F16	3.14E+06	1.21E+06	+	4.01E−02	5.38E−02	+	6.11E−01	2.05E−01	+	2.04E+01	1.41E+01	+	7.52E−02	7.22E−02	+	8.68E−20	8.11E−20	+	1.16E−03	3.34E−03	+	1.26E−02	5.79E−03	+	3.61E+01	1.69E+01	+	**1.30E−06**	**3.73E−06**
F17	2.32E+01	1.91E+00	+	2.97E−03	2.42E−03	+	9.85E−04	1.22E−03	+	3.72E+00	1.58E+00	+	2.21E−42	2.26E−42	+	7.45E−11	1.56E−10	+	5.14E−03	3.76E−03	+	1.97E+00	1.51E+00	+	3.86E+00	1.77E+00	+	**0**	**0**
F18	3.01E+00	1.94E−01	+	8.87E−02	1.26E−01	+	**0**	**0**	≈	1.16E+00	3.64E−01	+	0.00E+00	0.00E+00	≈	7.39E−02	9.31E−02	+	1.65E−07	4.15E−07	+	2.08E−05	1.27E−05	+	1.02E+00	3.46E−01	+	**0**	**0**
F19	2.15E+02	1.11E+02	+	9.30E+01	5.17E+01	+	1.61E−07	5.72E−07	+	5.34E+01	1.76E+01	+	2.49E−17	9.34E−17	+	1.24E−07	1.60E−07	+	1.51E+02	4.41E+01	+	2.89E+02	4.90E+01	+	2.32E+02	7.58E+01	+	**0**	**0**
F20	1.35E+08	2.81E+07	+	8.13E+06	2.36E+07	+	1.45E−27	2.17E−27	+	1.16E+07	7.56E+06	+	6.67E−78	1.42E−77	+	4.32E+01	2.24E+01	+	7.36E−02	1.49E−01	+	9.43E+00	3.39E+00	+	1.43E+07	1.05E+07	+	**0**	**0**
F21	**0**	**0**	≈	**0**	**0**	≈	**0**	**0**	≈	**0**	**0**	≈	**0**	**0**	≈	**0**	**0**	≈	**0**	**0**	≈	**0**	**0**	≈	**0**	**0**	≈	**0**	**0**
F22	8.69E+00	6.01E−01	+	8.81E−01	1.20E−01	+	2.03E−01	3.20E−02	+	1.90E+00	4.00E−01	+	9.99E−02	2.02E−08	+	3.70E−01	4.66E−02	+	5.13E−01	6.34E−02	+	1.24E+00	1.34E−01	+	5.63E+00	1.17E+00	+	**0**	**0**
F23	4.94E−01	1.29E−03	+	1.96E−01	3.27E−02	+	3.95E−02	1.48E−02	+	3.96E−01	4.75E−02	+	9.72E−03	3.30E−08	+	9.72E−03	1.35E−07	+	1.54E−01	3.68E−02	+	4.02E−01	2.32E−02	+	4.97E−01	1.67E−03	+	**0**	**0**

Table 11
Comparison with other meta-heuristic algorithms on fixed-dimensional problems.

Test function	PSO			wPSO			GWO			SSA			TLBO			CMA-ES			SinDE			IDE			FA			MSCA	
	mean	STD	S.O.	mean	STD	S.O.	mean	STD	S.O.	mean	STD	S.O.	mean	STD	S.O.	mean	STD	S.O.	mean	STD	S.O.	mean	STD	S.O.	mean	STD	S.O.	mean	STD
F24	9.98E−01	2.20E−04	+	**9.98E−01**	**8.25E−17**	+	3.87E+00	3.78E+00	+	1.13E+00	3.44E−01	+	1.13E+00	4.31E−01	+	1.27E+01	1.59E−13	+	1.16E+00	9.00E−01	+	9.98E−01	0.00E+00	−	2.79E+00	1.82E+00	+	9.98E−01	2.28E−10
F25	2.22E−03	4.94E−03	+	1.24E−03	3.63E−03	+	2.42E−03	6.09E−03	+	2.10E−03	4.97E−03	+	1.68E−03	5.08E−03	+	**3.07E−04**	**2.44E−19**	−	5.73E−04	1.99E−04	+	5.82E−04	1.78E−04	+	2.59E−03	4.15E−03	+	4.08E−04	2.07E−04
F26	−1.03E+00	6.18E−04	+	−1.03E+00	6.32E−16	≈	−1.03E+00	1.17E−08	≈	−1.03E+00	4.66E−14	≈	−1.03E+00	6.78E−16	≈	**−1.03E+00**	**4.52E−16**	−	−1.03E+00	6.78E−16	−	−1.03E+00	6.78E−16	−	−1.03E+00	6.15E−13	≈	−1.03E+00	2.03E−06
F27	3.98E−01	2.59E−04	+	**3.98E−01**	**0**	−	3.98E−01	4.10E−07	≈	3.98E−01	8.74E−15	≈	**3.98E−01**	**0**	−	**3.98E−01**	**0.00E+00**	−	**3.98E−01**	**0**	−	**3.98E−01**	**0**	−	3.98E−01	2.15E−13	≈	3.98E−01	1.34E−04
F28	3.00E+00	2.77E−03	+	3.00E+00	1.71E−15	−	3.00E+00	2.23E−05	+	3.00E+00	3.14E−13	−	3.00E+00	1.35E−15	−	1.38E+01	2.80E+01	+	3.00E+00	1.33E−15	−	**3.00E+00**	**1.28E−15**	−	4.59E+00	6.09E+00	+	3.00E+00	1.55E−05
F29	−3.86E+00	3.66E−04	−	−3.86E+00	2.58E−15	−	−3.86E+00	2.02E−03	s≈	−3.86E+00	1.48E−10	−	−3.86E+00	2.71E−15	−	**−3.86E+00**	**2.37E−15**	−	−3.86E+00	2.71E−15	−	−3.86E+00	2.71E−15	−	−3.86E+00	4.41E−03	+	−3.86E+00	3.20E−03
F30	−3.25E+00	6.50E−02	+	−3.28E+00	5.90E−02	−	−3.26E+00	7.74E−02	−	−3.22E+00	5.26E−02	−	−3.28E+00	5.83E−02	−	−3.25E+00	5.99E−02	−	−3.28E+00	5.64E−02	−	**−3.31E+00**	**2.71E−02**	≈	−3.31E+00	3.55E−02	−	−3.12E+00	1.57E−01
F31	−6.39E+00	2.35E+00	+	−6.57E+00	3.51E+00	+	−8.98E+00	2.44E+00	+	−8.31E+00	2.94E+00	+	−9.64E+00	1.56E+00	+	−5.06E+00	8.25E−16	+	−1.02E+01	7.01E−15	−	−9.82E+00	1.28E+00	+	−4.91E+00	3.49E+00	+	**−1.02E+01**	8.90E−05
F32	−7.23E+00	2.64E+00	+	−7.92E+00	3.40E+00	+	−1.01E+01	1.42E+00	+	−8.44E+00	3.09E+00	+	−9.83E+00	1.77E+00	+	−5.09E+00	3.48E−15	+	−9.93E+00	1.82E+00	+	**−1.04E+01**	**8.22E−10**	≈	−6.45E+00	3.56E+00	+	**−1.04E+01**	8.77E−05
F33	−7.96E+00	2.34E+00	+	−7.39E+00	3.92E+00	+	−9.84E+00	2.14E+00	+	−8.01E+00	3.46E+00	+	−1.00E+01	1.65E+00	+	−5.13E+00	2.95E−15	+	−1.03E+01	1.40E+00	+	**−1.05E+01**	**1.78E−15**	≈	−6.18E+00	3.89E+00	+	**−1.05E+01**	7.72E−05

Table 12
Ranking of all the compared algorithms based on the mean objective function values.

Algorithm	m-SCA	ISCA	modSCA	SCA-GWO	SCA-PSO	OBSCA	SCA	PSO	wPSO	GWO	SSA	TLBO	CMA-ES	SinDE	IDE	FA	MSCA
F1	12	5	15	3	8	7	14	17	11	4	9	2	6	10	13	16	1
F2	11	5	14	3	8	7	13	17	10	4	15	2	6	9	12	16	1
F3	10	5	13	3	8	6	12	17	14	4	15	2	7	9	11	16	1
F4	9	6	15	5	8	7	13	16	11	4	10	2	3	14	17	12	1
F5	7	4	16	5	9	8	17	15	10	6	13	2	3	11	12	14	1
F6	8	4	16	6	9	7	15	17	13	5	11	3	2	10	12	14	1
F7	12	9	15	11	6	13	14	17	7	10	2	4	1	5	8	16	3
F8	10	5	16	3	8	7	15	17	11	4	9	2	6	13	12	14	1
F9	7	2	12	6	8	5	13	17	10	4	14	3	16	9	11	15	1
F10	9	5	16	3	6	7	15	17	8	4	13	2	12	11	10	14	1
F11	13	8	15	11	7	14	16	12	3	10	6	5	17	2	4	9	1
F12	10	3	11	5	8	2	9	17	13	4	14	7	6	12	16	15	1
F13	8	5	17	3	11	16	14	15	10	4	12	2	6	7	9	13	1
F14	10	2	12	6	7	3	13	15	9	4	8	1	1	5	11	14	1
F15	11	7	15	9	5	12	17	16	10	8	14	1	4	3	6	13	2
F16	11	8	16	10	4	12	15	17	6	9	13	7	1	3	5	14	2
F17	11	5	16	6	9	4	13	17	8	7	14	2	3	10	12	15	1
F18	4	1	6	1	2	1	7	12	9	1	11	1	8	3	5	10	1
F19	11	6	8	3	7	9	10	15	13	5	12	2	4	14	17	16	1
F20	9	5	12	3	7	6	13	17	14	4	15	2	11	8	10	16	1
F21	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
F22	7	4	14	5	10	3	12	17	11	6	15	2	8	9	13	16	1
F23	7	4	12	6	8	3	11	15	10	5	13	2	2	9	14	16	1
F24	3	1	8	13	7	11	9	2	1	12	5	4	14	6	1	10	1
F25	3	6	8	13	11	7	9	15	10	16	14	12	1	4	5	17	2
F26	1	1	2	1	1	1	3	4	1	1	1	1	1	1	1	1	1
F27	3	1	6	1	1	7	5	4	1	1	1	1	1	1	1	1	2
F28	5	1	7	3	1	8	6	9	1	4	1	1	11	1	1	10	2
F29	2	1	7	6	1	10	9	3	1	4	1	1	1	1	1	5	8
F30	1	4	16	8	9	15	17	12	7	10	13	6	11	5	2	3	14
F31	4	6	16	8	13	3	17	12	11	9	10	7	14	1	5	15	2
F32	5	3	16	9	11	4	17	13	12	6	10	8	15	7	1	14	2
F33	5	4	16	8	10	3	17	12	13	9	11	7	15	6	1	14	2
Average rank	**7.27**	**4.15**	**12.27**	**5.67**	**6.94**	**6.94**	**12.15**	**13.30**	**8.48**	**5.73**	**9.88**	**3.24**	**6.61**	**6.67**	**7.88**	**12.27**	**1.91**
Overall rank	**9**	**3**	**14**	**4**	**8**	**8**	**13**	**15**	**11**	**5**	**12**	**2**	**6**	**7**	**10**	**14**	**1**

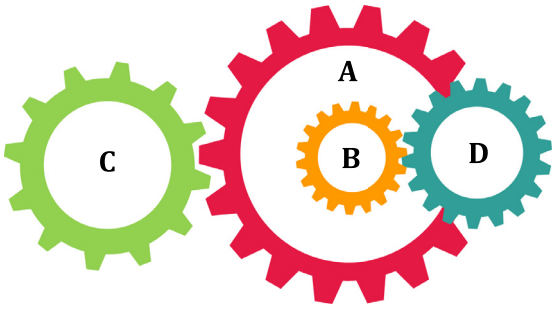

Fig. 6. Gear train (Gandomi et al., 2013).

constraint violation value Deb (1999). First, the solutions are sorted according to their constraint violation values and then the feasible solutions are sorted according to their objective function values. In the end, from this sorted list, required number of best solutions can be elected.

4.4.1. Gear train design

This case study is an unconstrained problem in nature, which was introduced by Sandgren (1990). In this problem four discrete decision parameters namely, y_1, y_2, y_3 and y_4 are involved as shown in Fig. 6. These variables represent the number of teeth on a gearwheel. The aim of this problem is to determine a minimum cost of gear ratio, defined by $\frac{y_2 y_3}{y_1 y_4}$. Mathematically, gear train design problem can be stated as follows:

$$\min F_1(y_1, y_2, y_3, y_4) = \left(\frac{1}{6.931} - \frac{y_2 y_3}{y_1 y_4}\right)^2 \tag{9}$$

where $12 \le y_1, y_2, y_3, y_4 \le 60$ and each $y_i (i = 1, 2, 3, 4)$ is an integer.

Table 13
Results comparison for gear train design problem.

Algorithm	y_1	y_2	y_3	y_4	$F_{1,min}$
MSCA	49	16	19	43	2.7009E−12
SCA	60	17	28	55	1.3616E−09
m-SCA	34	20	13	53	2.3078E−11
ISCA	46	26	12	47	9.9216E−10
modSCA	60	18	25	52	2.3576E−09
OBSCA	43	12	31	60	8.7008E−09
PSO	47	13	12	23	9.9216E−10
wPSO	57	12	37	54	8.8876E−10
GWO	23	13	12	47	9.9216E−10
SSA	57	27	17	60	3.6358E−09
TLBO	53	30	13	51	2.3078E−11
FA	43	19	16	49	2.7009E−12
SCA-PSO	46	24	13	47	9.9216E−10
SCA-GWO	51	26	15	53	2.3078E−11
CMA-ES	53	13	20	34	2.3078E−11
SinDE	30	17	14	55	1.3616E−09
IDE	47	13	24	46	9.9216E−10

In order to compare the results of the MSCA, same set of meta-heuristic algorithms i.e. classical SCA, m-SCA, ISCA, modSCA, OBSCA, PSO, wPSO, GWO, SSA, TLBO, FA, SCA-PSO, CMA-ES and IDE are implemented on gear train design problem. The similar experimental environment (i.e. 20 population size and 200 function evaluations) are adopted to obtain the results. The values of the obtained best objective function and corresponding set of decision variables is shown in Table 13. Among these results, the proposed MSCA has obtained similar objection function to the FA and superior results as compared to other algorithm, therefore, can be prefer to solve this design problem over other algorithms.

Table 14
Results comparison for FM-design problem.

Algorithm	y_1	y_2	y_3	y_4	y_5	y_6	$F_{2,min}$
MSCA	1.0000	5.0000	1.5000	−4.8000	−2.0000	4.9000	1.0262E−10
SCA	1.6110	0.0138	0.3584	0.0096	−4.2191	4.9080	1.1498E+01
m-SCA	−0.7681	−0.1153	1.0201	−0.0924	−4.3373	−4.8989	9.2462E+00
ISCA	−0.9962	−5.0019	−1.5067	−4.7991	−1.9977	4.8999	1.3493E−03
modSCA	0.4689	0.0695	−2.0419	0.0022	−4.0943	−4.8736	1.1459E+01
OBSCA	0.5363	5.0092	−1.1195	4.7112	1.5632	4.9615	1.1322E+01
PSO	−0.6602	0.0706	−4.1631	−4.9025	0.2175	0.3165	1.4083E+01
wPSO	−1.0000	−5.0000	−1.5000	−4.8000	2.0000	−4.9000	0.0000E+00
GWO	0.9452	4.9996	1.4580	−4.7990	−2.0160	4.9020	1.3429E−01
SSA	1.0000	5.0000	1.5000	−4.8000	−2.0000	4.9000	1.0262E−10
TLBO	−1.0000	−5.0000	−1.5000	−4.8000	−2.0000	4.9000	0.0000E+00
FA	0.6233	0.0105	4.3328	−4.8884	−0.0299	−2.7211	1.1495E+01
SCA-PSO	1.0000	5.0000	1.5000	−4.8000	−2.0000	4.9000	1.1632E−07
SCA-GWO	−0.9915	−4.9947	−1.4508	−4.8011	−2.0173	4.9004	4.2100E−02
CMA-ES	0.9452	4.9996	1.4580	−4.7990	−2.0160	4.9020	1.3429E−01
SinDE	1.0000	5.0000	−1.5000	4.8000	2.0000	4.9000	0.0000E+00
IDE	1.0000	5.0000	1.5000	−4.8000	2.0000	−4.9000	1.2140E−10

4.4.2. Frequency-modulated (FM) design problem

FM sound wave synthesis has an important role in several modern music systems. This problem is about to optimize the six parameters of an FM synthesizer. The task of this problem is generate a sound Eq. (10) similar to target sound Eq. (11). The mathematical expressions for estimated sound and target sound waves can be given by:

$$y(t) = a_1 \sin(\omega_1 t\theta + a_2 \sin(\omega_2 t\theta + a_3 \sin(\omega_3 t\theta)))d \tag{10}$$

$$y_0(t) = 1.0 \sin(5.0t\theta + 1.5 \sin(4.8t\theta + 2.0sin(4.9t\theta))) \tag{11}$$

respectively. The value of θ is fixed to $2\pi/100$ and the parameter range is $[-6.4, 6.35]$. The objective function for this problem can be stated as follows:

$$\min F_2(Y) = \sum_{t=0}^{100} (y(t) - y_0(t))^2 \tag{12}$$

where $Y = (a_1, \omega_1, a_2, \omega_2, a_3, \omega_3) = (y_1, y_2, y_3, y_4, y_5, y_6)$

For this problem, the results obtained by the MSCA are compared to the same set of meta-heuristic algorithms as used in previous problem i.e. classical SCA, m-SCA, ISCA, modSCA, OBSCA, PSO, wPSO, GWO, SSA, TLBO, FA, SCA-PSO, CMA-ES and IDE. The similar experimental environment (i.e. 20 population size and 10^5 function evaluations) are adopted to obtain the results. The values of the obtained best objective function and corresponding set of decision variables is shown in Table 14. Among these results, the proposed MSCA has obtained similar objection function to the SSA and outperforms all other algorithms. Hence, the proposed MSCA can be preferred to solve this problem

4.4.3. Speed reducer design

This problem is a structural design problem Gandomi et al. (2013). A line diagram of speed reducer is shown in Fig. 7. The decision variables involved in this design problem are: "face width (y_1)", "module of teeth (y_2)", "number of teeth (y_3) on pinion", "length of shafts between bearings y_4 and y_5", and "diameter of shafts (y_6 and y_7)". The objective of this problem is to determine the optimal setting of decision variables so that the weight of speed reducer can be minimized. The optimization model of this problem is expressed as follows:

$$\begin{aligned} min \quad f_3(Y) = {} & 0.7854 y_1 y_2^2 \left(3.3333 y_3^2 + 14.9334 y_3 - 43.0934\right) \\ & - 1.508 y_1 \left(y_6^2 + y_7^2\right) + 7.4770 \left(y_6^3 + y_7^3\right) \\ & + 0.7854 \left(y_4 y_6^2 + y_5 y_7^2\right) \end{aligned} \tag{13}$$

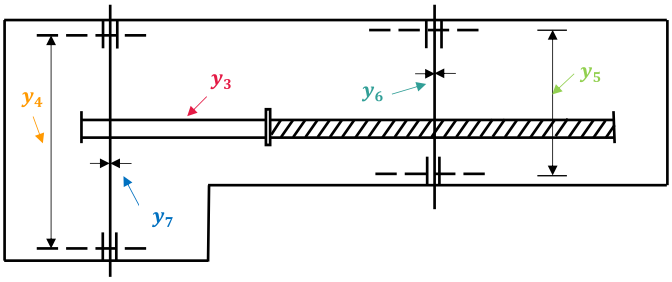

Fig. 7. Speed reducer (Gandomi et al., 2013).

where $Y = (y_1, y_2, y_3, y_4, y_5, y_6, y_7) \in \mathbb{R}^7$

$$s.t. \quad \frac{27}{y_1 y_2^2 y_3} - 1 \le 0 \tag{14}$$

$$\frac{397.5}{y_1 y_2^2 y_3^2} - 1 \le 0 \tag{15}$$

$$\frac{1.93}{y_2 y_3 y_4^3 y_6^4} - 1 \le 0 \tag{16}$$

$$\frac{1.93}{y_2 y_3 y_4^3 y_7^4} - 1 \le 0 \tag{17}$$

$$\frac{\sqrt{1.69 \times 10^6 + \left(\frac{745 y_4}{y_2 y_3}\right)^2}}{110 y_6^3} - 1 \le 0 \tag{18}$$

$$\frac{\sqrt{157.5 \times 10^6 + \left(\frac{745 y_4}{y_2 y_3}\right)^2}}{85 y_7^3} - 1 \le 0 \tag{19}$$

$$\frac{y_2 y_3}{40} - 1 \le 0 \tag{20}$$

$$\frac{y_1}{12 y_2} - 1 \le 0 \tag{21}$$

$$\frac{5 y_2}{y_1 - 1} - 1 \le 0 \tag{22}$$

The result obtained on this problem by the proposed algorithm MSCA is compared with the classical SCA and other algorithms

Table 15
Results comparison for speed reducer design design problem.

Algorithm	y_1	y_2	y_3	y_4	y_5	y_6	y_7	$F_{2,min}$
MSCA	3.5000	0.7000	17.0000	7.3000	7.8000	3.3502	5.2867	2996.3484
SCA	3.6000	0.7000	17.0000	7.3000	7.8000	3.3541	5.3144	3054.3071
m-SCA	2.9295	0.7555	19.6258	7.6285	8.2435	3.6931	5.4041	3138.9115
ISCA	3.5163	0.7000	17.0000	7.6817	7.8000	3.3510	5.2890	3007.7818
modSCA	3.6000	0.7000	17.0000	7.4742	7.8000	3.3966	5.3145	3066.8849
OBSCA	3.0879	0.7550	26.4738	7.3650	7.9577	3.4950	5.2312	3056.3122
PSO	3.6000	0.7000	17.0000	7.5814	7.9147	3.3738	5.1934	3005.5076
wPSO	3.5549	0.7025	17.0000	7.6985	8.0447	3.3862	5.2900	3018.4426
GWO	3.5004	0.7000	17.0000	7.3348	7.8805	3.3693	5.2900	3005.5830
SSA	3.5233	0.7000	17.0000	7.3009	7.8481	3.3519	5.2867	3007.0135
TLBO	3.5000	0.7000	17.0000	7.3000	7.8000	3.3502	5.2867	2996.3484
FA	3.6000	0.7000	17.0027	7.7632	7.9914	3.5669	5.3898	3013.9134
SCA-PSO	3.6000	0.7000	17.0000	7.6532	7.9801	3.3520	5.2876	3043.7059
SCA-GWO	3.5092	0.7000	17.0000	7.3257	7.8444	3.3772	5.2871	3008.3368
CMA-ES	3.5000	0.7000	17.0000	7.3000	7.8000	3.3502	5.2867	2996.3484
SinDE	3.5000	0.7000	17.0000	7.3000	7.8000	3.3502	5.2867	2996.3484
IDE	3.5000	0.7000	17.0000	7.3000	7.8000	3.3502	5.2867	2996.3484

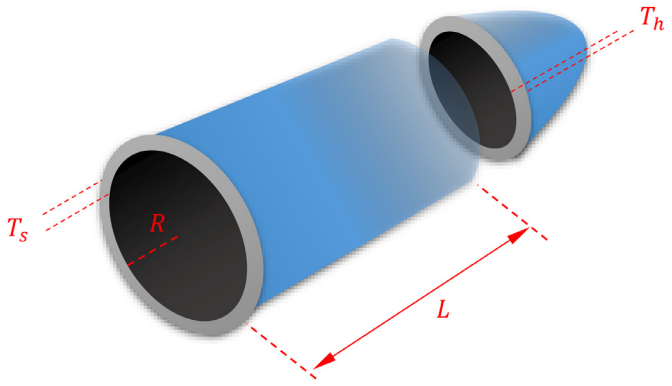

Fig. 8. Pressure vessel (Gandomi et al., 2013).

such as i.e. m-SCA, ISCA, modSCA, OBSCA, SCA-GWO, SCA-PSO, PSO, wPSO, GWO, SSA, TLBO, CMA-ES, SinDE, IDE and FA. For the comparison, identical experimental environment (i.e. same number of function evaluations as utilized by Gandomi et al. (2013) is used. The comparison of results is presented in Table 15, which indicates that the MSCA provides better results than the classical SCA, m-SCA, ISCA, modSCA, OBSCA, PSO, wPSO, GWO, SSA, FA, SCA-PSO and SCA-GWO algorithms. The obtained results by the TLBO, CMA-ES, SinDE and IDE are similar to the MSCA.

4.4.4. Pressure vessel design

The aim of this study is to obtain a minimum cost of cylindrical pressure vessel as shown in Fig. 8, which is closely related to material, structure, and welding. The ends of the pressure vessel are covered. In this problem, "the thickness of the shell (y_1)", "head (y_2)", "inner radius (y_3)" and "range of cross-section minus head (y_4)" are needed to be optimized. The description of the problem is shown mathematically as follows:

$$\min \quad f_4(Y) = 0.6224y_1y_3y_4 + 1.7781y_1^2y_3 + 3.1661y_1^2y_4 + 19.84y_1^2y_3 \tag{23}$$

where $Y = (y_1, y_2, y_3, y_4) = (T_{SH}, T_{HD}, R, L) \in \mathbb{R}^4$

$$s.t. \quad -y_1 + 0.0193y_3 \le 0 \tag{24}$$

$$-y_3 + 0.00954y_3 \le 0 \tag{25}$$

$$-\pi y_3^2 y_4 - \frac{4}{3}\pi y_3^3 + 12,96,000 \le 0 \tag{26}$$

Table 16
Results comparison for pressure vessel design problem.

Algorithm	y_1	y_2	y_3	y_4	$F_{3,min}$
MSCA	0.81250	0.43750	42.09840	176.6366	6059.7144
SCA	0.8125	0.4375	42.0605	180.8232	6151.1441
m-SCA	3.6875	0.125	30.1353	40.0806	6574.8581
ISCA	0.8125	0.4375	42.09476	176.6874	6060.2819
modSCA	0.8125	0.4375	41.54899	183.5795	6128.0763
OBSCA	3.375	1.3125	36.86738	128.8391	7088.4274
PSO	0.81250	0.18750	32.54139	145.5515	6158.7649
wPSO	0.9375	1.6875	36.86738	128.8391	6438.3219
GWO	0.81250	0.43750	42.09833	176.6403	6059.7811
SSA	0.81250	0.43750	42.09840	176.6366	6059.7144
TLBO	0.81250	0.43750	42.09845	176.6366	6059.7144
FA	0.81250	0.43750	42.09833	176.6403	6059.7811
SCA-PSO	0.81250	0.43750	42.09844	176.63660	6059.7147
SCA-GWO	0.875	0.4375	45.33548	140.2706	6090.7686
CMA-ES	0.81250	0.43750	42.09845	176.6366	6059.7144
SinDE	0.81250	0.43750	42.09845	176.6366	6059.7144
IDE	0.81250	0.43750	42.09845	176.6366	6059.7144

$$y_4 - 240 \le 0 \tag{27}$$

When the cost of pressure vessel is minimized using the proposed MSCA, same experimental setting is fixed, which is used in Gandomi et al. (2013) to obtain the results. Same set of algorithms, which are used in previous design problems, are used to compare and analyze the search efficiency of the MSCA. The numerical results are shown in Table 16. The table indicates that the MSCA provides better results than the classical SCA, m-SCA, ISCA, modSCA, OBSCA, PSO, wPSO, GWO, FA, SCA-PSO and SCA-GWO algorithms. The obtained results by the SSA, TLBO, CMA-ES, SinDE and IDE are similar to the MSCA.

5. Effect of the proposed MSCA on training multilayer perceptron

In this section, the proposed MSCA is employed for the training of multilayer perceptron (MLP). The MLP are those neural networks, where only one hidden layer and one-directional connections between their neurons are present. In these MLPs, first layer is called input layer and the last layer is known as output layer. After providing the weights, inputs and biases, the outputs of MLP can be calculated in the following steps:

Table 17
Details of the datasets.

Datasets	No. of attributes	No. of training samples	No. of test samples	Number of classes
3-bit XOR	3	8	8	2
Balloon	4	16	16	2
Iris	4	150	150	3
Cancer	9	599	100	2

1. First, the weighted sum of inputs is calculated as:

$$s_j = \sum_{i=1}^{n} w_{i,j} x_i - \theta_j, \quad j = 1, 2, \ldots, h \tag{28}$$

where n indicates the number of inputs, $w_{i,j}$ represents the connection weights from the ith input node to the jth node in the hidden layer, x_i is the ith input, θ_j is the threshold or bias of the jth node in the hidden layer.

2. The output of the each node in the hidden layer is calculated as:

$$S_j = \frac{1}{1 + e^{-s_j}} = sigmoid(s_j), \quad j = 1, 2, \ldots, h \tag{29}$$

3. In the end, the final output can be calculated based on the outputs of the nodes in the hidden layer.

$$o_k = \sum_{j=1}^{h} w_{j,k} S_j - \theta'_k, \quad k = 1, 2, \ldots, m \tag{30}$$

$$O_k = \frac{1}{1 + e^{-o_k}} = sigmoid(o_k), \quad k = 1, 2, \ldots, m \tag{31}$$

where $w_{j,k}$ indicate the connection weight from the jth hidden node to the kth output node, θ'_k is the threshold or bias of the kth output node. It can be observed from Eqs. (28)–(31) that the biases and weights are responsible for determining the final output of MLP from given inputs.

Since, the aim of training MLP is to attain highest classification accuracy for both training and testing samples. A common metric, which is used to evaluate the MLP is Mean Square Error (MSE) and is defined by Eq. (32)

$$MSE = \sum_{i=1}^{m} \left(o_i^k - d_i^k\right)^2 \tag{32}$$

where m denotes the number of outputs, o_i^k is the actual output of the ith input, when kth training sample appears in the input and d_i^k is the desired output of the ith input. In order to verify the efficacy, the MLP should adopt itself in such a way so that it can minimize the average of MSE over all the training samples. The average MSE, say F, is defined as:

$$F = \frac{\sum_{k=1}^{N} \sum_{i=1}^{m} \left(o_i^k - d_i^k\right)^2}{N} \tag{33}$$

Here N denotes the number of training samples. Thus, during the training of MLP, the task is to minimize the objective function F given by Eq. (33) with the decision variables called weights and biases.

5.1. Numerical results and discussion

This section evaluates the efficiency of the proposed MSCA for training MLP using four classification data sets namely, XOR, balloon, iris and breast cancer. These data sets are collected from the University of California at Irvine (UCI) Machine Learning Repository (Belew, McInerney, & Schraudolph, 1990). The experimental results are compared with the modified variants of SCA such as m-SCA (Gupta & Deep, 2019a), ISCA (Gupta & Deep, 2019b), modSCA (Nayak et al., 2018) and OBSCA (Elaziz et al., 2017). The range for the decision variables is fixed in the interval $[-10, 10]$. The population size of all the algorithms for the XOR and Balloon data set is fixed to 50 and for remaining data sets it is fixed to 200. The iteration are fixed to 250 for each algorithm (Mirjalili, 2015). The details of the datasets are provided in Table 17. The present work does not focus on finding the optimal number of hidden nodes and consider them equal to $2N + 1$, where N is the number of inputs in the dataset (Mirjalili, 2015).

Table 18
Experimental results for different datasets.

Dataset	Algorithm	MSE(mean ±STD)	classification rate
3-bit XOR	**MSCA**	7.5865E−05 ± 3.2801E−05	100
	SCA(+)	4.7439E−02 ± 3.4834E−02	100
	modSCA(+)	5.0637E−02 ± 3.5530E−02	75
	m-SCA(+)	5.6090E−02 ± 2.5684E−02	75
	ISCA(+)	1.7189E−04 ± 3.5767E−04	100
	OBSCA(+)	1.0126E−04 ± 9.7189E−05	100
	SCA-GWO(+)	1.3678E−02 ± 3.9190E−02	100
	SCA-PSO(+)	8.7681E−06 ± 1.2239E−05	100
Baboon	**MSCA**	1.7997E−08 ± 5.5540E−08	100
	SCA(+)	1.4267E−05 ± 2.0757E−05	100
	modSCA(+)	3.5169E−05 ± 6.4338E−05	100
	m-SCA(+)	3.6821E−06 ± 3.3734E−06	100
	ISCA(+)	1.2594E−04 ± 3.9638E−04	100
	OBSCA(+)	3.3184E−02 ± 8.2734E−03	100
	SCA-GWO(−)	7.0558E−15 ± 2.2294E−14	100
	SCA-PSO(-)	1.9443E−27 ± 6.1474E−27	100
Iris	**MSCA**	2.2900E−02 ± 3.1983E−03	91.33
	SCA(+)	1.7665E−01 ± 2.5319E−02	60
	modSCA(+)	1.5109E−01 ± 3.5413E−02	61.33
	m-SCA(+)	9.9068E−02 ± 2.5712E−02	42.67
	ISCA(+)	2.3195E−02 ± 1.1842E−03	91.33
	OBSCA(+)	3.3184E−02±8.2734E−03	91.33
	SCA-GWO(+)	2.3731E−02± 3.0877E−03	91.33
	SCA-PSO(≈)	1.8868E−02±3.4310E−03	96
Cancer	**MSCA**	1.4818E−03±6.8847E−05	100
	SCA(+)	1.0793E−02±5.3837E−03	99
	modSCA(+)	1.0758E−02±3.7134E−03	97
	m-SCA(+)	6.9875E−03±1.4268E−03	93
	ISCA(≈)	1.3722E−03± 1.3416E−04	100
	OBSCA(+)	9.7967E−03±4.8532E−03	94
	SCA-GWO(≈)	1.2809E−03±1.0541E−04	100
	SCA-PSO(+)	1.7017E−03±3.6983E−04	99

The 10 independent trials of each algorithm is performed on each of the data set and the mean and standard deviation value of the objective function F (average of MSE) is presented in Table 18. In the same table, the classification rate is also recorded corresponding to each of the applied algorithm. The outcomes obtained by the Wilcoxon rank-sum test is also mentioned in table, which clearly indicates the superiority of the proposed MSCA. Thus, by analyzing the MSE, standard deviation, classification rate and statistical outcomes, it can bee concluded that the proposed MSCA is better as a MLP trainer as compared to the classical SCA and other versions of the SCA. The reason for improved MSE is superior ability of avoiding the local optima during the search procedure by the MSCA, which is the effect of modified transition rule, mutation operator and modified search mechanism of the MSCA. The

mathematical formulation of the modified transition rule indicate that 65% of maximum iterations contribute to exploration phase. The modified search mechanism, where the new solutions are generated using chaotic map and Gaussian mutation, helps to promote the exploration that leads to finding diverse MLP structures during optimization process. Therefore, the comparative study shows that the proposed MSCA can be recommended over the classical SCA and other variants of SCA to be used in real-life optimization tasks such as training MLPs.

6. Conclusions and future research directions

In this paper, a novel variant of SCA, called MSCA was proposed by integrating three different strategies: modified transition rule, leading elite guidance of search, mutation operator with Gaussian mutation and chaotic map. The search procedure in the classical SCA showed an immature balance between the exploration and exploitation, therefore, in the MSCA, the level of both the phases was increased and tried to establish a comparatively better balance between these phases during the search. The experimental validation was performed on 33 benchmark test functions, having various category of difficulty levels. The performance of the MSCA was also examined on 500-dimensional problems to verify the impact on large dimensional problems. The experimental results obtained by the MSCA were compared with the classical SCA, modified variants of SCA and some other meta-heuristic algorithms. The comparison of results showed that the employed strategies are successful to upgrade the performance of the SCA and help to provide an optimal solution in most of the problems. To verify the applicability of the MSCA on real-life problems, the MSCA was applied to train MLP. The results and comparison demonstrated the efficacy of the MSCA on real-life problems.

In this study, the MSCA was proposed to solve single-objective optimization problems. However, many optimization problems involve multiple objectives and discrete decision variables in real-world situation. With respect to future work, the MSCA can be utilized to solve multi-objective, discrete and binary optimization problems such as vehicle scheduling problems, exciting aircraft streamline modeling problems, feature selection, optimal reactive power dispatch problem etc.

Declaration of Competing Interest

The authors declare that they do not have any financial or non-financial conflict of interests

Credit authorship contribution statement

Shubham Gupta: Conceptualization, Methodology, Software, Writing - original draft, Visualization, Writing - review & editing. **Kusum Deep:** Writing - review & editing, Visualization, Investigation, Supervision. **Seyedali Mirjalili:** Writing - review & editing, Visualization, Investigation. **Joong Hoon Kim:** Writing - review & editing, Visualization, Investigation, Supervision.

References

Abedinpourshotorban, H., Shamsuddin, S. M., Beheshti, Z., & Jawawi, D. N. (2016). Electromagnetic field optimization: A physics-inspired metaheuristic optimization algorithm. *Swarm and Evolutionary Computation, 26*, 8–22.

Attia, A.-F., El Sehiemy, R. A., & Hasanien, H. M. (2018). Optimal power flow solution in power systems using a novel sine-cosine algorithm. *International Journal of Electrical Power & Energy Systems, 99*, 331–343.

Belew, R. K., McInerney, J., & Schraudolph, N. N. (1990). *Evolving networks: Using the genetic algorithm with connectionist learning*. Citeseer.

Blum, C., & Roli, A. (2003). Metaheuristics in combinatorial optimization: Overview and conceptual comparison. *ACM Computing Surveys (CSUR), 35*(3), 268–308.

BoussaïD, I., Lepagnot, J., & Siarry, P. (2013). A survey on optimization metaheuristics. *Information Sciences, 237*, 82–117.

Deb, K. (1999). An ecient constraint handling method for genetic algorithms. In *Computer methods in applied mechanics and engineering*. Citeseer.

Derrac, J., García, S., Molina, D., & Herrera, F. (2011). A practical tutorial on the use of nonparametric statistical tests as a methodology for comparing evolutionary and swarm intelligence algorithms. *Swarm and Evolutionary Computation, 1*(1), 3–18.

Dorigo, M., & Birattari, M. (2010). *Ant colony optimization*. Springer.

Draa, A., Bouzoubia, S., & Boukhalfa, I. (2015). A sinusoidal differential evolution algorithm for numerical optimisation. *Applied Soft Computing, 27*, 99–126.

Eberhart, R., & Kennedy, J. (1995). A new optimizer using particle swarm theory. In *Mhs'95. Proceedings of the sixth international symposium on micro machine and human science* (pp. 39–43). Ieee.

Elaziz, M. A., Oliva, D., & Xiong, S. (2017). An improved opposition-based sine cosine algorithm for global optimization. *Expert Systems with Applications, 90*, 484–500.

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.

Gupta, S., & Deep, K. (2019a). A hybrid self-adaptive sine cosine algorithm with opposition based learning. *Expert Systems with Applications, 119*, 210–230.

Gupta, S., & Deep, K. (2019b). Improved sine cosine algorithm with crossover scheme for global optimization. *Knowledge-Based Systems, 165*, 374–406.

Hansen, N., & Ostermeier, A. (2001). Completely derandomized self-adaptation in evolution strategies. *Evolutionary Computation, 9*(2), 159–195.

Hashim, F. A., Houssein, E. H., Mabrouk, M. S., Al-Atabany, W., & Mirjalili, S. (2019). Henry gas solubility optimization: A novel physics-based algorithm. *Future Generation Computer Systems, 101*, 646–667.

Heidari, A. A., Mirjalili, S., Faris, H., Aljarah, I., Mafarja, M., & Chen, H. (2019). Harris hawks optimization: Algorithm and applications. *Future Generation Computer Systems, 97*, 849–872.

Holland John, H. (1975). *Adaptation in natural and artificial systems*. Ann Arbor: University of Michigan Press.

Hongfeng, Z. (2019). Dynamic economic dispatch based on improved differential evolution algorithm. *Cluster Computing, 22*(4), 8241–8248.

Issa, M., Hassanien, A. E., Oliva, D., Helmi, A., Ziedan, I., & Alzohairy, A. (2018). Asca-pso: Adaptive sine cosine optimization algorithm integrated with particle swarm for pairwise local sequence alignment. *Expert Systems with Applications, 99*, 56–70.

Jain, M., Singh, V., & Rani, A. (2019). A novel nature-inspired algorithm for optimization: Squirrel search algorithm. *Swarm and Evolutionary Computation, 44*, 148–175.

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.

Kennedy, J. (2010). Particle swarm optimization. *Encyclopedia of Machine Learning*, 760–766.

Lin, J., & Zhong, Y. (2013). *Accelerated shuffled frog-leaping algorithm with Gaussian mutation*.

Meshkat, M., & Parhizgar, M. (2017). A novel weighted update position mechanism to improve the performance of sine cosine algorithm. In *Proceedings of the 2017 fifth iranian joint congress on fuzzy and intelligent systems (cfis)* (pp. 166–171). IEEE.

Mirjalili, S. (2015). How effective is the grey wolf optimizer in training multi-layer perceptrons. *Applied Intelligence, 43*(1), 150–161.

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

Mirjalili, S., & Gandomi, A. H. (2017). Chaotic gravitational constants for the gravitational search algorithm. *Applied Soft Computing, 53*, 407–419.

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

Mirjalili, S. M., Mirjalili, S. Z., Saremi, S., & Mirjalili, S. (2020). Sine cosine algorithm: Theory, literature review, and application in designing bend photonic crystal waveguides. In *Nature-inspired optimizers* (pp. 201–217). Springer.

Nayak, D. R., Dash, R., Majhi, B., & Wang, S. (2018). Combining extreme learning machine with modified sine cosine algorithm for detection of pathological brain. *Computers & Electrical Engineering, 68*, 366–380.

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, 43*, 1–30.

Rahmanzadeh, S., & Pishvaee, M. S. (2019). Electron radar search algorithm: A novel developed meta-heuristic algorithm. *Soft Computing*, 1–23.

Rao, R. V., Savsani, V. J., & Vakharia, D. (2011). Teaching–learning-based optimization: A novel method for constrained mechanical design optimization problems. *Computer-Aided Design, 43*(3), 303–315.

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

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

Shi, Y., & Eberhart, R. C. (1998). Parameter selection in particle swarm optimization. In *Proceedings of the international conference on evolutionary programming* (pp. 591–600). Springer.

Simon, D. (2008). Biogeography-based optimization. *IEEE Transactions on Evolutionary Computation, 12*(6), 702–713.
Singh, N., & Singh, S. (2017). A novel hybrid gwo-sca approach for optimization problems. *Engineering Science and Technology, an International Journal, 20*(6), 1586–1601.
Wolpert, D. H., Macready, W. G., et al. (1995). No free lunch theorems for search. *Technical Report*. Technical Report SFI-TR-95-02-010, Santa Fe Institute.
Wolpert, D. H., Macready, W. G., et al. (1997). No free lunch theorems for optimization. *IEEE Transactions on Evolutionary Computation, 1*(1), 67–82.
Yang, X.-S. (2010). Firefly algorithm, stochastic test functions and design optimisation. arXiv preprint arXiv:1003.1409.
Yazdani, M., & Jolai, F. (2016). Lion optimization algorithm (loa): A nature-inspired metaheuristic algorithm. *Journal of Computational Design and Engineering, 3*(1), 24–36.
Yilmaz, S., & Sen, S. (2019). Electric fish optimization: A new heuristic algorithm inspired by electrolocation. *Neural Computing and Applications*, 1–36.
Zhenyu, G., Bo, C., Min, Y., & Binggang, C. (2006). Self-adaptive chaos differential evolution. In *Proceedings of the international conference on natural computation* (pp. 972–975). Springer.