Advances in Engineering Software 198 (2024) 103784

Contents lists available at ScienceDirect

Advances in Engineering Software

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

Centroid opposition-based backtracking search algorithm for global optimization and engineering problems

Sanjib Debnath [a,b], Swapan Debbarma [a], Sukanta Nama [c,*], Apu Kumar Saha [d,*], Runu Dhar [e], Ali Riza Yildiz [f], Amir H. Gandomi [g,h,**]

[a] *Department of Computer Science & Engineering, National Institute of Technology Agartala, Tripura 799046, India*
[b] *Department of Computer Science & Engineering, The ICFAI University Tripura, Tripura, 799210, India*
[c] *Department of Science & Humanities, Gomati District Polytechnic, Fulkumari, Udaipur, Tripura 799013, India*
[d] *Department of Mathematics, National Institute of Technology Agartala, Tripura 799046, India*
[e] *Department of Mathematics, Maharaja Bir Bikram University, Agartala, Tripura 799004, India*
[f] *Department of Mechanical Engineering, Bursa Uludag University, Turkey*
[g] *Faculty of Engineering and IT, University of Technology Sydney, Ultimo, NSW 2007, Australia*
[h] *University Research and Innovation Center (EKIK), Óbuda University, Budapest 1034, Hungary*

A R T I C L E I N F O

Keywords:
Centroid opposition-based learning (CODL)
Backtracking search algorithm
Multiple learning
Chaos elite strategy
Engineering design problem

A B S T R A C T

Evolutionary algorithms (EAs) have a lot of potential to handle nonlinear and non-convex objective functions. Particularly, the backtracking search algorithm (BSA) is a popular nature-based evolutionary optimization method that has attracted many researchers due to its simple structure and efficiency in problem-solving across diverse fields. However, like other optimization algorithms, BSA is also prone to reduced diversity, local optima, and inadequate intensification capabilities. To overcome the flaws and increase the performance of BSA, this research proposes a centroid opposition-based backtracking search algorithm (CoBSA) for global optimization and engineering design problems. In CoBSA, specific individuals simultaneously acquire current and historical population knowledge to preserve population variety and improve exploration capability. On the other hand, other individuals execute the position from the current population's centroid opposition to progress convergence speed and exploitation potential. In addition, an elite process based on logistic chaotic local search was developed to improve the superiority of the current individuals. The suggested CoBSA was validated on a set of benchmark functions and then employed in a set of application examples. According to extensive numerical results and assessments, CoBSA outperformed the other state-of-the-art methods in terms of accurateness, reliability, and execution capability.

1. Introduction

In our everyday lives, optimization emerges as a pervasive and fundamental element intricately connected with technology. Whether dealing with operations research, machine learning, drug design, engineering design, or other numerical fields, optimizing functions is a common task that must be completed. As a result, optimization methods are gaining importance day-by-day. Due to the increasing complexity of the optimization problems involved in many real-world scenarios, researchers worldwide focus on developing various efficient optimization algorithms. One of the successful endeavours of this search is the introduction of metaheuristic algorithms. Although these algorithms do not always provide the best solution due to their highly stochastic and heuristic nature, they are useful tools to solve almost all types of optimization problems, especially complicated problems in terms of non-linearity, non-differentiability, higher dimensionality, etc. Numerous metaheuristic algorithms have already been introduced, and the development and improvement of such algorithms continue. Together with their improved variants, these methods have been applied to solve numerous optimization problems in different fields [1–9].

The backtracking search algorithm (BSA) was developed by Civicioglu as a novel population-based heuristic technique for solving complex optimization problems across various branches of science and engineering [1]. The procedure has a straightforward process, requires

* Corresponding authors.
** Corresponding author at: Faculty of Engineering and IT, University of Technology Sydney, Ultimo, NSW 2007, Australia.
E-mail addresses: sukanta1122@gmail.com (S. Nama), apusaha_nita@yahoo.co.in (A.K. Saha), gandomi@uts.edu.au (A.H. Gandomi).

https://doi.org/10.1016/j.advengsoft.2024.103784
Received 27 October 2023; Received in revised form 19 September 2024; Accepted 22 September 2024
Available online 12 October 2024

only one algorithm-specific control parameter, and is unaffected by the control parameter's starting value [2–4]. BSA contains a memory pool that stores individuals' knowledge from the preceding formation population at random, then collectively forms the search direction matrix with the present individuals. In terms of versatility and efficiency, BSA and its variations have been widely employed for a wide variety of real-life optimization problems, such as economic dispatch problems [5,6], feature selection [7], artificial neural networks [7–9], optimal power flow [10,11], parameter identification [12–14], flow shop scheduling [15], hydrothermal plant [16], energy management [17], and nonlinear optimal control problem [18].

Furthermore, the backtracking search method, which uses population, randomness, and simple structures, performs exceptionally well in addressing global optimization problems. The basic idea of BSA is simple and easy to understand, making it accessible for implementation. BSA systematically explores the solution space by trying out different possibilities one at a time. Yet, BSA has disadvantages in some circumstances. For instance, it has mixed parameters that need to be fine-tuned for optimal performance, as selecting appropriate values for these parameters can be a non-trivial task. BSA does not guarantee to find the optimal solution since it often traps trapped in the local optima. Due to these challenges, many studies have improved the performance of BSA. For example, one study applied specular reflection learning in order to provide promising solutions during the execution of optimization [19]. Zhao et al. [20] presented a multi-population cooperative evolution strategy that directs BSA with hierarchical knowledge (HKBSA), thereby improving its performance. Hannan et al. [21] described the implementation of an artificial neural network (ANN)-based binary BSA (BBSA) for optimum planning controller on an IEEE 14-bus system for regulating MGs generated virtual power plants (VPPs) towards RESs integration. Nama et al. [22] developed a method to eliminate the disadvantages of classic Symbiotic Organisms Search (SOS) by providing a new collaborative method dubbed e-SOSBSA to change the degree of exploration and exploitation. To tackle the original PSO algorithm's issues, Zaman and Gharehchopogh [23] introduced an enhanced PSO with BSA (PSOBSA), in which the operators of BSA are adjusted to raise the convergence rate through the neighborhood. Because BSA's single search process makes it challenging to address non-separable problems, Zhao et al. [24] suggested BSA based on knowledge of different populations (DKBSA) to find the optimum output of continuous optimization problems. In [25], an improved BSA (ImBSA) was proposed by considering a multi-population approach and modified control parameter settings to apprehend an ensemble of various mutation strategies. Kuyu et al. [26] suggested a new hybridization technique that combines BSA and differential evolution (DE) algorithms. To sustain population diversity, the proposed method implements diversity beating and stagnation detection procedures and improvements to the mutation strategy of the constituent procedures to increase search proficiency. These changes are self-adaptive and executed at the same time. To enhance the global search ability of BSA, another study proposed an improved version of BSA called the backtracking search algorithm driven by generalized mean position (GMPBSA) [27]. Based on the comprehensive learning mechanism and local escape operator, an enhanced BSA (EBSA) is reported in [28], which is inspired by the characteristics of the considered flight planning problem and the weak ability of the backtracking search algorithm to escape from a local optimum. Other researchers developed a new ensemble algorithm called e-mPSOBSA with the aid of the reformed BSA and PSO taking into account that PSO gets stuck in a suboptimal solution due to an unbalanced trade-off between exploitation and exploration [29]. A backtracking search optimization algorithm with a dual scatter search strategy (BSA-DS) was proposed in [39], which incorporates a dual scatter search (DS) strategy into the BSA with strong exploration capability. A new population-based optimization algorithm called BSA with dynamic population (BSADP), consisting of the proposed enhanced BSA (EBSA) and the designed population adjustment mechanism with opposition-based learning (PAMOBL), was presented in [30].

From the literature, it is concluded that:

- The historical population is incorporated by BSA, which suggests that individuals execute their locations based on the experiences of earlier generations [1].
- During the search process, BSA drifts into the surrounding optimum solution and suffers from a lack of intensification [7,15,19–21,23,24,26].
- Without a plan, BSA progressively decreases population diversity [1,9,20,22]. It may find a valid solution but not necessarily the optimal solution.
- Individuals cannot readily locate the feasible search domain for the best individual because information about the finest individual in the current population is not considered, causing delayed convergence of BSA [3,4,9,22,31–33].
- BSA's crossover operator is controlled by the mixrate option, and selecting appropriate values for these parameters can be a non-trivial task [1,3,4,31].

1.1. Motivation

The presented study investigated the challenges of utilizing BSA on unimodal and multimodal benchmark test problems, which were employed to measure functional requirement space for acceleration and examination of solutions as well as the capacity to avoid neighbourhood optima. As a relatively newer algorithm, BSA has several drawbacks, including slow execution speed. Initially, only historical population data were used to direct the search. Since the current population's data are underutilized, resulting in a population diversity that cannot be maintained effectively, BSA's exploration ability is limited. Another issue is that there is little instruction on how to approach the present best individual in the execution procedure, causing delayed convergence and poor exploitation abilities. Moreover, to the best of the researchers' knowledge, no study exists in the literature that proposes the implementation of centroid opposition-based learning models into BSA to improve its performance.

According to the No Free Lunch (NFL) theorems [34], an ideal optimization procedure that solves all complex hard problems cannot be devised. In other words, if an optimization method works well on some specific optimization problems, it may not perform well on another specific type of optimization problem. Centroid opposition-based learning mechanism is a popular approach that many researchers use to improve the performance of various algorithms [35]. For instance, the sine cosine algorithm (SCA) gets trapped in local optima and suffers from premature convergence for some problems due to a lack of search space exploration. Therefore, a new method, COBSCA, was proposed based on SCA and centroid opposition-based computing (COBC) to enhance SCA's exploration ability [36]. Manafi et al. [37] proposed COCRO by combining centroid opposition and coral reefs algorithm (CRA) for solving an automated guided vehicle routing problem with a recharging constraint. The neighborhood centroid opposition-based grasshopper optimization algorithm (NCOGOA) was proposed based on combining the grasshopper optimization algorithm (GOA) and neighbourhood centroid opposition-based learning to improve the algorithm's exploration and exploitation ability [38]. By incorporating the centroid opposition-based learning (COBL) [35] process into BSA, the designed CoBSA can boost exploration and exploitation. The uniformly distributed population generation method in COBL improves the diversity of the initial set of individuals in CoBSA. Furthermore, the COBL technique generates new populations based on the centroid population during the evolutionary process. The primary principle behind COBL is to consider a random position and its opposite position based on the centroid position simultaneously to obtain a better position for the present candidate solution.

1.2. Contributions

Depending on a random probability, in CoBSA, certain individuals upgrade their locations by taking into account both the current and historical population information at the same time, which might increase population variety. Comparatively, other individuals update their places according to the best individuals in the present set to accelerate convergence. This can increase the execution of the optimization technique during the implementation of algorithm steps. In addition, an elite mechanism based on chaotic local search is included in the suggested method to further enhance the location of the current population.

The main contributions of this work may be summarized as follows:

- A new, improved variant of BSA called CoBSA is proposed, which combines centroid opposition-based learning, multiple learning methods based on centroid opposition-based jumping and chaotic logistic map-based elite local search mechanism.
- Implementation of centroid opposite-based learning mechanisms to CoBSA can increase exploration in the early stages to ensure a diverse search and then shift towards exploitation during the optimization process.
- The proposed mutation strategy based on chaotic 'logistic map' can exploit the solution space during the optimization process.
- The proposed multiple-learning strategy, which combines the centroid opposition-based learning and chaotic mutation strategy, can change the characteristics of individuals during the optimization process and can help strike a balance between exploration and exploitation.
- The proposed chaotic logistic map-based elite local search mechanism maintains population diversity to ensure a balance between exploration and exploitation. This helps prevent premature convergence to suboptimal solutions.
- The proposed CoBSA was extensively tested on 20 classical benchmark functions, and the CEC2020 test suite, and it was also applied to real-world engineering design problems.
- Upon evaluation in terms of numerical evaluation, nonparametric statistical analysis, scalability analysis and sensitivity analysis, it was found that CoBSA yields highly comparable and often better results than those acquired by popular optimization methods in the literature.

The following is the breakdown of the paper's structure. The classical BSA technique is defined in Section 2. Section 3 describes the theory and implementation of the suggested CoBSA. Section 4 evaluates the performance of CoBSA on some selected typical benchmark functions. Section 5 presents the application of the suggested CoBSA on six application examples. Finally, Section 6 concludes the present work.

2. Backtracking search algorithm (BSA)

The stochastic evolutionary algorithm is a type of algorithm that uses a random number generator in which BSA is executed based on the solution set as an iterative process. Initialization, selection-I, mutation, crossover, and selection-II are the five primary components of BSA's search space execution.

Initialization: Initially, the set of solutions is defined by Eq. (1):

$$Pop_{i,j} = Pop_{min} + r_1 * (Pop_{max} - Pop_{min});\ r_1 \in (0,1),\ \forall\ i = 1\ to\ NP\ and\ j = 1\ to\ D \tag{1}$$

where NP is the size of the solution set; D represents the number of variables of the optimization problem; and $Pop_{min} and Pop_{max}$ are the inferior and superior bounds of the ith individuals, respectively.

Selection-I: In this step, the initial historical population (*OldPop*) is determined, which is defined in Eq. (2):

$$OldPop_{i,j} = Pop_{min} + r_2 * (Pop_{max} - Pop_{min});\ r_2 \in (0,1),\ \forall\ i = 1\ to\ NP\ and\ j = 1\ to\ D \tag{2}$$

After defining the initial *OldPop*, it is restructured at every generation by utilizing Eq. (3):

$$If\ a < b,\ OldPop = Pop;\ end;\ a, b \in (0,1) \tag{3}$$

Subsequently, the location of individuals in a set of *OldPop* is redefined using Eq. (4):

$$OldPop = permuting\ (OldPop) \tag{4}$$

Mutation: For each individual, a set of trial populations, called the mutant, is determined via Eq. (5):

$$Mutant = Pop + CF * (OldPop - Pop) \tag{5}$$

where the search-direction matrix (*OldPop* − *Pop*) is monitored by CF. As the historical population is engaged in calculating the search-direction matrix, BSA produces a set of trial individuals by benefiting from the capabilities of the preceding iteration. The value of CF is defined by $CF = 3*Rn$, where *Rn*is a normally distributed random number from $N(0, 1)$.

Crossover: In the BSA crossover process, at first, a binary integer-valued matrix (*map*) of size *NP*D* is determined by Eqs. (6) and (7), in which the mix rate parameter (M) indicates how many individuals are manipulated with the current individual to produce a set of new individuals. After that, the ultimate form of the trial population (T) is determined by Eq. (8):

$$map_{i,u(1:M*r_3*D)} = 0 | \mathrm{u} = \mathrm{permutating}(\langle 1,\ 2,\ 3,\ \ldots,\ \mathrm{D}\rangle);\ r_3 \in (0,1) \tag{6}$$

$$map_{i,randi(D)} = 1;\ randi\ is\ a\ integer\ number\ between\ 1\ to\ D \tag{7}$$

$$T_{i,j} = \begin{cases} Mutant_{i,j}\ if\ map_{i,j} = 1 \\ Pop_{i,j}\ if\ map_{i,j} = 0 \end{cases} \tag{8}$$

Selection II: The final set of individuals using the trial population (T) and current population is formed using Eq. (9):

$$P_{i,j} = \begin{cases} T_{i,j}\ if\ f(T_{i,j}) < f(P_{i,j}) \\ P_{i,j}\ Otherwise \end{cases}; \tag{9}$$

3. Centroid opposition-based BSA (CoBSA)

Considering that the historical population is employed in BSA, as previously mentioned, individuals upgrade their positions by leveraging past generations' knowledge [1]. In contrast to other population-based methods that do not employ the population of prior steps, this technique makes BSA distinctive and compelling. However, because the historical population is formed at the start of each generation by randomly picking from the previous and recent generations' populations, this cannot guarantee that the present set of individuals' locations is always collected to build the historical population [22]. As a consequence, the set of solutions upgrades their locations exclusively under the supervision of preceding iteration populations, which may lead to a rapid loss in population diversification if no effort to increase it is implemented [33]. In inclusion, the finest candidate solution in the existing population is not considered, resulting in a poor convergence speed for BSA since a set of solutions may be incapable of quickly discovering the prospective exploration zone suggested by the best candidate solution.

Furthermore, even though BSA has effectively solved issues in various domains, the authors are unaware of any attempt to apply COBL in BSA. Accordingly, CoBSA is proposed to improve BSA's performance in estimating real-world problems. To be more explicit, multiple learning approaches are suggested to attain the appropriate stability between intensification and diversification capabilities. A chaotic-based elite local search mechanism is also presented to increase the current population's superiority. The following section provides an explanation of CoBSA's central concept.

3.1. Centroid opposition-based learning (COBL)

Opposition-based learning (OBL), which was developed by Tizhoosh [39], is applied to EAs to speed up convergence and identify optimal global solutions for a specific optimization task. Within the solution space of OBL, the present population and its opposite population are both formed simultaneously. The goal of OBL is to create opposing numbers closer to the global solution than any other number generated at random.

However, there is still a need to speed up the OBL scheme. When computing the opposite point, two bounds (min and max) are picked from two extreme ends in the set of individuals for each dimension. The enduring ideas of the set of individuals are ignored. For the formation of the opposite set of individuals, the suggested COBL considers the total set of individuals. As a result, COBL attempts to progress BSA in terms of convergence speed and individual correctness by utilizing a much more significant amount of data: the entire population vs. extreme locations. The proposed centroid opposition-based point and centroid point are subsequently specified.

Centroid Point: Let $\{x_i;\ i = 1, 2, 3, \ldots, n\}$represent n arguments in a D-dimensional domain field, each with a unit mass. The body's centroid M can be defined as follows:

$$M = \frac{x_1 + x_2 + x_3 + \ldots + x_n}{n} \qquad (9)$$

And

$$M_i = \frac{\sum_{i=1}^{n} x_{i,j}}{n},\ j = 1, 2, 3, \ldots, D \qquad (10)$$

Centroid Opposite Point: The opposite-point x_i^{cop} of an argument x_i with the centroid is denoted by M and is calculated via Eq. (11):

$$x_i^{cop} = 2.M - x_i \qquad (11)$$

When a centroid-opposite point leaves the domain field, it must be reflected back to an arbitrary argument in order to remain a valid opposite point—using $[a, b]$ for the search space and M for the centroid point. Eq. (12) is utilized to reflect in the search space:

$$\underset{x}{\smile} = d + rand(0,1).(c - d) \qquad (12)$$

where $d = \begin{cases} M, & \check{x} > b \\ a, & \check{x} < a \end{cases}$, $c = \begin{cases} b, & \check{x} > b \\ M, & \check{x} < a \end{cases}$

The centroid opposition-based initialization and centroid oppositional jumping based on the centroid opposite point are presented in Algorithm 1 and Algorithm 2, respectively.

Implementation of centroid opposite-based learning mechanisms to the proposed method can increase exploration in the early stages to ensure a diverse search and then shift towards exploitation as the optimization progresses.

Algorithm 1
Pseudo-code of the centroid opposite-based initialization.

```
Define the set of individuals $P_0$ from uniformly distributed manner from the domain field.
For $i = 1$ to NP
  For $j = 1$ to D
    $OP_{i,j} = 2^*M_{i,j} - P_{i,j}$
  End
End
As the initial set of individuals $P_0$, identify the NP number of individuals from the set $\{P_0 \cup OP_0\}$
```

3.2. Centroid opposite and chaotic based on multiple learning strategies

As demonstrated in Eq. (5), the mutation operator plays a vital character in the BSA search process in creating new individuals. Yet, only historical population data are utilized to steer the exploration, and the present set of individual data is not guaranteed to be utilized adequately at each iteration step. As a result, BSA's population variety and exploratory ability rapidly dwindled.

Furthermore, information regarding the best individual in the present population is omitted, which is critical to BSA's ability to converge. As a result, multiple learning techniques for updating individual positions are added in order to establish appropriate stability among exploration and exploitation skills. Specifically, some individuals learn from the centroid opposite learning approach and the present set of individuals simultaneously, while others acquire knowledge from the best-guided individual in the present set of individuals according to a random probability. As seen in Algorithm 3, multiple learning approaches are applied to develop new individuals.

The proposed chaotic-based multiple-learning strategy can dynamically adjust the individual's behavior based on the current state of optimization. The combination of the centroid opposite and chaotic-based multiple learning approaches presented in Algorithm 3 changes the characteristics of individuals during the optimization process and can help strike a balance between exploration and exploitation. While the initial point is 0.7, Z_k is formed from the 'logistic map described by Eq. (14), for which the chaotic behavior is illustrated in Fig. 1.

$$x_{i+1} = ax_i(1 - x_i);\ a = 4,\ x_0 = 0.7 \qquad (14)$$

Some individuals use the proposed multiple learning method to improve population diversity by gaining information from both the centroid opposite population and the present population. Comparatively, others use it to improve convergence speed by gaining from the best candidate individual in the present population. As a result, the algorithm's execution in solving optimization problems can be improved by taking into account both exploration and exploitation abilities.

3.3. Chaotic logistic map-based elite local search mechanism

The current population is employed to preserve the better individuals and is considered to assist in the search during the entire evolution procedure. For example, the feature of current population's during the execution of the optimization method is critical and can influence possible search directions. This work presents an elite method constructed on a chaotic logistic map to further refine the current population's quality. Finding a better alternative solution near the current most effective solution is the primary objective of the local search. The well-recognized logistic map shown in Eq. (14) is utilized because chaotic sequences have randomicity and ergodicity, which is particularly advantageous for further improving the superiority of one individual by creating additional solutions around it [28,32].

Furthermore, the best individual is expected to have more disturbances in the early search stages for a better individual. Since the finest candidate solution is closest to the global optimum, further knowledge about the best candidate solution should be kept. To build a new V* about the best candidate solution, the chaotic local search provided in Eq. (15) is used. The elite approach is then employed to receive the new candidate solution if its fitness value is higher than the lowest in the present population. As a result, the present population's quality can be refined at each generation by using a chaotic logistic map-based elite local search mechanism.

$$V_j = \begin{cases} \mu(1 + z_m)\left(Pop_{best,max} + Pop_{best,min}\right) - z_m Pop_{i,j}\ if\ r_4 < 1 - (FE/MAX\_FE) \\ Pop_{best},\ Otherwise \end{cases} \qquad (15)$$

where m denotes the number of generations; z_m is the value of mth

Algorithm 2
Opposition-based generation jumping.

```
Upgrade the domain field to $_{j}^{min}P$, $_{j}^{max}P$
// **Opposition-based generation jumping.**
**If ($k$<Jr)|$k \in$ rand(0, 1)**
// **evaluation of centroid point.**
For $i = 1$ to D
  $M_i = 0$
  For $j = 1$ to NP
    $M_i = M_i + P_{G,i,\ j}$
  End
End
// **End evaluation of centroid point.**
// **Calculation of centroid opposite population $OP_0$**
For $i = 1$ to NP
  For $j = 1$ to D
    $OP_{G,i,\ j} = 2^{*}M_{i,j} - P_{G,i,\ j}$
  End
End
// **End calculation of centroid opposite population $OP_0$**
If any opposite point goes outside the domain field, it will initialize again between $M_i$
and the boundary point.
As the initial population $\boldsymbol{P_G}$, identify the NP number of individuals from the set
$\{\boldsymbol{P_G} \cup \boldsymbol{OP_G}\}$
**End if**
```

Algorithm 3
Centroid opposite and chaotic-based multiple learning strategy.

```
If rand < Jr
  Find $V_{i,j}$ by utilizing Algorithm 2
Else
  $V_{i,j} = Z_k.X_{best} + (Z_k - \mu).(X_{i,k} - X_{j,k})$, $\mu = 0.5$ (13)
End
```

chaotic generation; the initial value of $z_0 = 0.7$; FE is the present quantity of objective function evaluations; MAX_FE is the total fitness evaluation number; and $\mu = 0.5$; $r_4 \in (0, 1)$. The proposed chaotic logistic map-based elite local search mechanism maintains population diversity to ensure a balance between exploration and exploitation, preventing premature convergence to suboptimal solutions.

3.4. CoBSA framework

The pseudo-code of the suggested CoBSA technique is summarized in Algorithm 4, and the flowchart of CoBSA is illustrated in Fig. 2. The foregoing modifications to the standard BSA search method were made to improve the exploitation of the domain field with the individual best memory of the population set and deliver a path to the population's suitable individual. At the beginning step of the algorithm for producing the initial population, the centroid opposition population keeps the surplus of variety from skipping the genuine solutions to the population set and preserves the appropriate stability among exploration and exploitation. The multiple learning techniques retain the finest characteristic of the individual from the population set and prevent the solution set from overflowing with variability. When trapped in the local optimum solution and unable to steer the search, the elite mechanism, with the help of chaotic logistic local search, assists in exploring favourable provinces about the finest individual of the population. All the mechanisms described in the above section are combined into the proposed CoBSA to improve in exploration or exploitation. This combination often leverages the strengths of different approaches to achieve a more balanced and robust performance.

If an individual goes outside the domain field, then it will reflect back to the search boundary via Eq. (16):

$$Pop_i^{new} = \begin{cases} \mathrm{Pop}_{\min} + rand\ (0,1)\ * (\mathrm{Pop}_{\max} - \mathrm{Pop}_{min})\ if\ Pop_i < \mathrm{Pop}_{\min} \\ \mathrm{Pop}_{\max} - rand(0,\ 1) * (Pop_{\max} - \mathrm{Pop}_{min})\ if\ Pop_i > \ \mathrm{Pop}_{\max} \end{cases} \tag{16}$$

where $i = 1, 2, 3, \ldots\ldots\ldots, NP$; and $\mathrm{Pop}_{\min}$ and $\mathrm{Pop}_{\max}$ indicate the lower and upper bounds of the domain field for the ith population, respectively. The suggested technique is described in more detail below.

Step 1: Initialization – Initialize the number of decision variables D and domain field of test functions, including the fixed size of individuals NP and the certain amount of iterations Max_it. Generate a set of individuals defined as P_0within the domain field using Eq. (1). Then, utilize the COBL procedure via Algorithm 1 to get its centroid opposite solutionsCOP_0. From $\{P_0 \cup COP_0\}$, select the best NP individuals for the initial population according to the fitness value. Define the*OldPop* within the domain field using Eq. (2).
Step 2: Setting *OldPop* and *BestPop* – Define the ultimate form of OldPop for every targeted individual using Eqs. (3) and (4).
Step 3: Mutation – With the mutation operator represented in Eq. (5), create the trial population.
Step 4: Crossover – Calculate the ultimate form of the trial population using the crossover technique described in Eqs. (6) and (7).

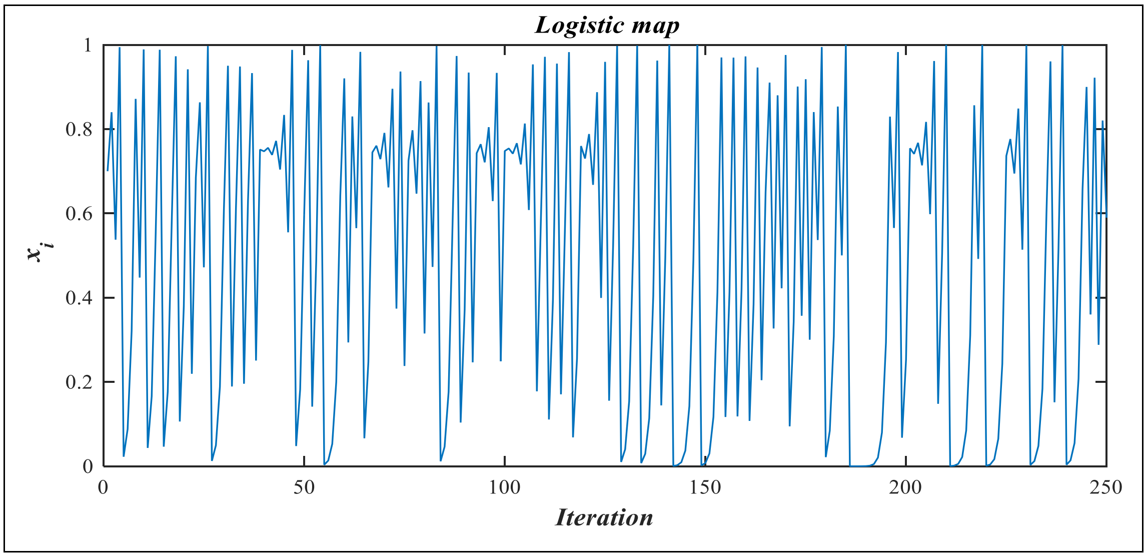

Fig. 1. Visualization of chaotic logistic maps.

Algorithm 4
Pseudo-code of the proposed CoBSA.

```
(1)    Initialization:
       Initialize number of candidate solutions, Maximum quantity of fitness
       evaluations, value of mix-rate, value of mutation rate parameter (F), and
       domain of each decision variables.
       Define the initial population by Eq. (1) and algorithm 1, select the NP
       number of the population as an initial population.
(2)    Main loop:
       While (The criteria for stopping have not been met.)
(2a)       Define OldPop by utilizing Eqs. (3) and (4).
(2b)       Using the Mutation operator provided by Eq. (5), refresh each individual
           inside the set of population.
(2c)       Create the ultimate form of each individual in the trial population through
           using Crossover technique provided by Eqs. (6), (7) and (8) respectively.
(2d)       Using Eq. (16), modify the infeasible individuals to make them feasible
           during the execution of the algorithm.
(2e)       Execute individual by multiple learning strategies using Algorithm 3.
(2f)       Using Eq. (16), modify the infeasible individuals to make them feasible
           during the execution of the algorithm.
(2 g)      Apply chaotic elite mechanism using Eq. (15).
(2 h)      Using Eq. (16), modify the infeasible individuals to make them feasible
           during the execution of the algorithm.
(2i)   End While
```

Using Eq. (15), repair each infeasible trial population, and for each trial population, determine the objective function value.
Step 5: Multiple learning strategies – By utilizing Algorithm 3, obtain the population $V_{i,j}$ constructed on the jumping rate, then calculate and choose the best NP solution from the combination of $P_{i,j}$ and $V_{i,j}$ as the starting solution for the next round.
Step 6: Elite mechanism – Apply the elite mechanism using Eq. (15) to remove the worst individual from the population set.
Step 7: Terminating condition – The procedure is repeated from Step 2 until a specific terminating condition is fulfilled.

3.5. Computational complexity

Analysis of an algorithm's complex is critical in the field of optimization. Thus, the complexities of standard BSA and the suggested CoBSA are presented in this section according to their pseudo-codes in Algorithm 4 with the usual big-O notation. Additionally, an algorithm's computing complexity is distributed into three portions: initialization, calculation of the goal function, and population update method. A comprehensive explanation of the computational complexities of conventional BSA and the suggested CoBSA is subsequently provided.

Complexity of conventional BSA:

- The set of individuals is initialized in O (NP $\times$ D) time in the conventional BSA. Here, NP is the size of the population, and D denotes the number of decision variables of the desired problem.
- Calculating the objective function value takes O (NP) times.
- Determining the set of the historical population takes O (NP $\times$ D) time.
- (NP $\times$ D) time is required to utilize the mutation operator.
- (NP $\times$ D) time is required to use the crossover operator.
- (NP $\times$ D) time is required when modernizing the location of each individual in the conventional BSA.

Thus, conventional BSA's computational complexity is approximately equal to O(NP $\times$ D $\times$ T_{max}). Here, T_{max} represents the total iterations.

Complexity of suggested CoBSA:

- The proposed CoBSA initializes the set of individuals in O (NP $\times$ D) time.
- (NP) times are required to calculate the objective function value.
- (NP $\times$ D) times are required to define the set of the historical population.
- (NP $\times$ D) times are required to utilize multiple learning operators.
- (NP $\times$ D) times are required to employ the crossover operator.
- (NP $\times$ D) times are required when updating the position of each individual.
- Utilization of an elite local search constructed on a chaotic map requires O(D) time.

Thus, the suggested CoBSA's computational complexity is approximately equal to O(NP $\times$ D $\times$ T_{max}+ D $\times$ T_{max}).

The added computational cost of CoBSA is due to the multiple learning and elite local search constructed on the chaotic map, which is more sophisticated than the original BSA. Subsequently, the complexity of conventional BSA is O(NP $\times$ D $\times$ T_{max}), and the total complexity of the suggested CoBSA is O(NP $\times$ D $\times$ T_{max} + D $\times$ T_{max}). As a result, the suggested CoBSA does not add considerably to the overall complexity of conventional BSA.

4. Performance results and analysis

This section describes the performance of CoBSA on twenty standard functions and ten IEEE CEC20 test functions.

4.1. Details of benchmark functions

The suggested CoBSA was employed on a set of 20 well-known and conventional benchmark problems [27,33] to assess its scalability, considering that these problems are scalable by nature due to dimension 45. The titles of the traditional benchmark test features, mathematical definitions, domain ranges, and optimal values are listed in Appendix A. The symbol "D" in Appendix A represents the problem dimensions. All investigations were implemented on MATLAB R2015a version with Intel (R) Core (TM) i5 CPU @ 2.40 GHz on an 8-GB RAM configuration computer.

For rational evaluation, the population size and quantity of function evaluations for each algorithm used to measure the performance of CoBSA were considered to be the same. Thirty independent tests were run on each algorithm to ensure consistent solutions and search strategies. The value of NP was set to 30, and 15,000 function evaluations were performed for BSA variants, including BSA [1], Adaptive Backtracking Search Algorithm (ABSA) [31], Improved Backtracking Search Algorithm (IBSA) [3], hybrid DE and BSA (hDEBSA) [4], Hybrid Backtracking Search Algorithm (HBSA) [41], Opposition-Based Backtracking Search Algorithm (OBSA) [42], Multi learning Backtracking Search Algorithm (MLBSA) [33], and Backtracking Search Algorithm with Specular Reflection Learning (BSA_SRL) [19], and some other well-established algorithms, such as the Bat Algorithm (BA) [43], Cuckoo Search (CS) [44], Firefly Algorithm (FA) [45], Flower Pollination Algorithm (FPA) [46], Particle Swarm Optimization (PSO) [47], Differential Evolution (DE) [48], Artificial Bee Colony (ABC) [49], Moth Flame Optimization (MFO) [50], Sine Cosine Algorithm (SCA) [51], Spherical Search Algorithm (SSO) [52], Tunicate Swarm Algorithm (TSA) [53], Salp Swarm Algorithm (SSA) [54], and Seagull Optimization Algorithm (SOA) [55].

4.2. Comparison of proposed CoBSA, standard BSA, and its variants

This subsection compares the proposed CoBSA to a standard BSA version and includes some BSA variants for traditional benchmarking concerns. Table 1 represents the results of dimension 45 based on the mean statistical measures and standard deviation of the fitness value in 30 independent runs. Table 2 displays the results where CoBSA performed superior, inferior, and equal to the compared algorithm. It is apparent that CoBSA outperformed BSA, ABSA, IBSA, hDEBSA, HBSA, OBSA, MLBSA, and BSA_SRL on 18, 19, 20, 18, 18, 19, 19, and 19 test functions. In other words, CoBSA is not inferior to any of the compared algorithms.

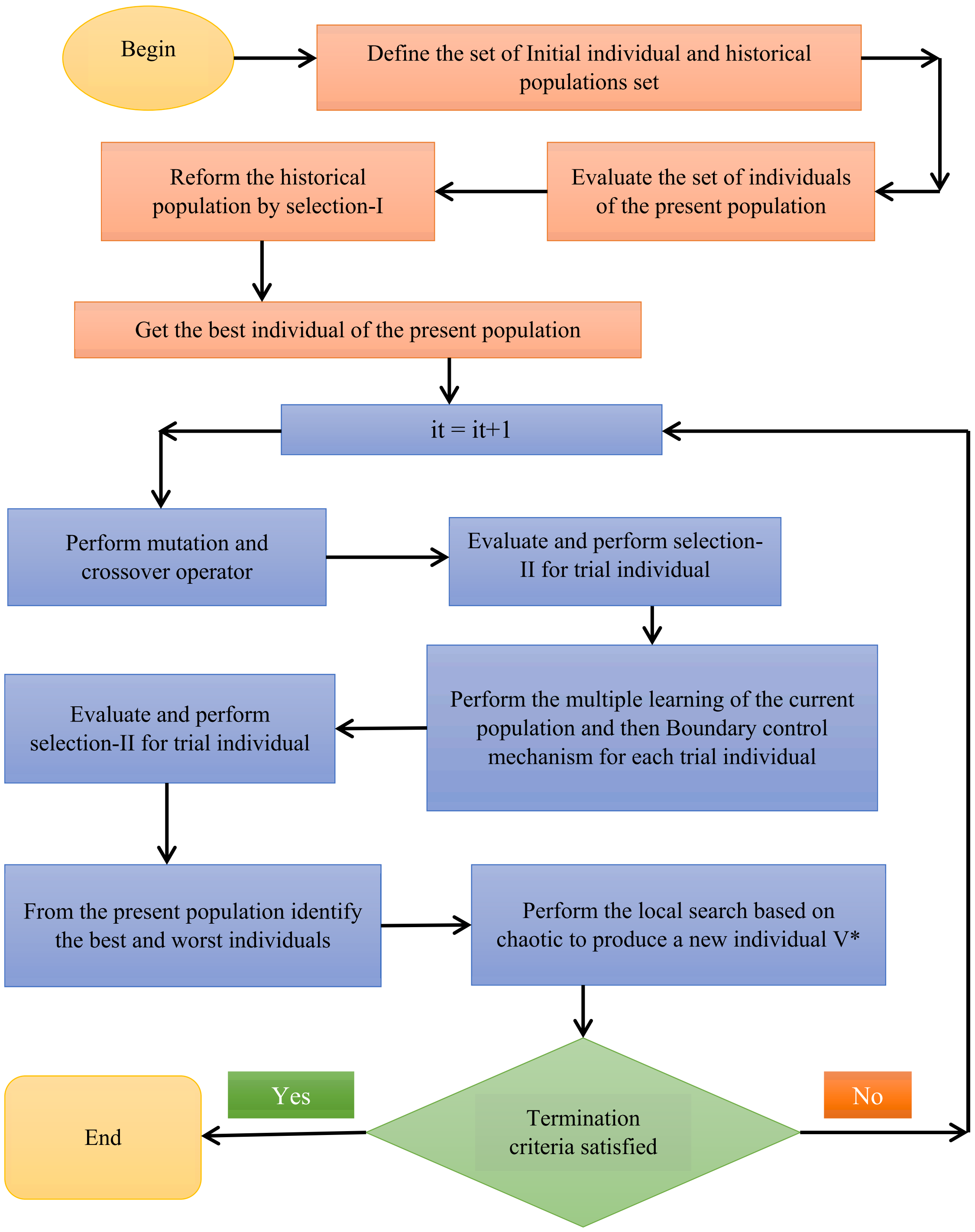
Begin
Define the set of Initial individual and historical populations set
Evaluate the set of individuals of the present population
Reform the historical population by selection-I
Get the best individual of the present population
it = it+1
Perform mutation and crossover operator
Evaluate and perform selection-II for trial individual
Perform the multiple learning of the current population and then Boundary control mechanism for each trial individual
Evaluate and perform selection-II for trial individual
From the present population identify the best and worst individuals
Perform the local search based on chaotic to produce a new individual V*
Termination criteria satisfied
Yes
No
End

Fig. 2. Flowchart of CoBSA.

Fig. 3 depicts a schematic view of the results, which reveals that the ensemble operator improves the solution execution process, allowing the optimizer to escape local optimum and globally converge to the best solution. According to the comparison data, CoBSA considerably upgraded the executions of BSA and obtained competitive outcomes in contrast to other state-of-the-art optimization approaches. The overall review of the results on these 20 objective functions reveals that CoBSA demonstrated a better convergence rate, development, and harmonizing of exploration and exploitation power compared to conventional BSA and its variants. As a result, the recommended CoBSA can be considered an excellent optimizer to classic BSA, which includes several alternatives for solving optimization problems, like unconstrained nonlinear challenging problems, with high precision.

4.2.1. *Diversity analysis*

Previous work reported that sustaining a set of diverse individuals is a crucial premise of continuous evolution during optimization [56]. Studies on population diversity in evolutionary algorithms can aid in gaining a better understanding of the optimization process. Therefore, researchers are currently investigating the population diversity of algorithms from various perspectives. In this work, a one measure method to evaluate the population diversity of CoBSA was utilized based on the population position's standard deviation, depending on the properties of the algorithm. The following is the definition of population diversity:

Definition. [56]: If individuals of a population $S = \{X_1, X_2, X_3, \ldots X_i., X_{N-1}, X_N\}$ get their positions $\{X_1(t), X_2(t), X_3(t), \ldots X_i(t)., X_{N-1}(t), X_N(t)\}$ at generation t, then $X_i(t)$ can be expressed as a vector $X_i(t) = (X_{i1}(t), X_{i2}(t), X_{i3}(t), X_{i4}(t), \ldots\ldots X_{iD}(t))$. Let $\overline{X}(t) = \left(\overline{X}^{(1)}, \overline{X}^{(2)}, \overline{X}^{(3)}, \ldots \overline{X}^{(j)}(t) \ldots . \overline{X}^{(D)}\right)$ and $\overline{X}^{(j)}(t) = \frac{1}{N}\sum_{i=1}^{N} X_{i,j}(t)$. The 'average distance around the population center' is calculated as:

$$AVG_{distance}(t) = \frac{1}{N}\sum_{i=1}^{N}\left(\sqrt{\sum_{j=1}^{D}\left(X_{i,j}(t) - \overline{X}^{(j)}(t)\right)^2}\right) \tag{18}$$

The larger the value of $AVG_{distance}(t)$ location (Variety), the more individuals of diverse sorts there are in the population, and the more the diversity is visible. Eq. (18) represents the population diversity from the point-of-view of individuals distributed in the domain of the problem agreeing to the distribution of population diversity indicators.

The average distance among the individuals in each iteration is presented in Fig. 4, which shows the increased diversity (exploration) of individuals and exploitation capabilities. First, it is worth noting that all of CoBSA's diversity measures decline toward zero on all functions. All of the diversity measures meet the requirements outlined above. Besides the fact that CoBSA has the most significant average diversity, its applicability to functions may be high.

As a result, the spacing between solutions in CoBSA is higher than in conventional BSA, confirming that the individuals are varied and a promising search location is explored. Because the distance graph in CoBSA is more peregrinated, it demonstrates the use of previously known search regions and the benefit of combining a centroid opposition learning chaotic elitism mechanism with multiple learning operators.

4.2.2. *Convergence analysis*

Fig. 5 illustrates the convergence curves for several functions. The horizontal axis in these graphs represents function evaluation, while the vertical axis represents function value. The dimension size of 45 was used to plot these curves. F1, F2, F3, F4, F7, F8, F16, F17, and F18 benchmark test problems from Appendix A were used for convergence curves in Fig. 5. Compared to simple BSAs, the figures demonstrate that CoBSA converges quickly into the global optima and attains high exactness. While other optimization algorithms can readily be translated to the local optimum, the convergence rate is likewise slow if the global optimum is reached first. This indicates that the suggested methodology is highly accurate and superior in terms of convergence rate to existing optimization techniques.

4.2.3. *Statistical analysis*

The analysis of statistical tests to increase the evaluation procedure of a new method's performance has become a common strategy in computational intelligence in recent years. Typically, a statistical analysis is performed in any experimental investigation to define whether one method is better than another. This problematic effort has become necessary in order to determine whether a newly proposed solution is an important advance over the existing algorithm for a given problem.

Kendall's W rank test [57] is a two-sample non-parametric technique used in hypothesis testing that determines if two samples characterize two separate populations or identify substantial variations among two sample means and rank two methods. Kendall's W is a non-parametric statistic, also known as Kendall's coefficient of concordance, that goes from 0 (no agreement) to 1 (total agreement). It is a normalization of the Friedman test statistic that can be applied to determine rater agreement. Suppose that $r_{i,j}$ is the rank of method 'i' by examining number j, where it is considered that there is 'n' number of methods and m number of judges. Then, the total number of ranks for the method 'I' is defined by $R_i = \sum_{j=1}^{m} r_{i,j}$ and the average quantity of the total ranks is $\overline{R} = \sum_{i=1}^{n} R_i$. The sum of squared deviations is denoted by S and is calculated as $S =$

Table 1
Performance results of BSA, ABSA, IBSA, hDEBSA, HBSA, OBSA, MLBSA, BSA_SRL and proposed CoBSA on twenty well known test function with dimension 45.

F	BSA		ABSA		IBSA		hDEBSA		HBSA		OBSA		MLBSA		BSA_SRL		CoBSA	
	Mean	Std	Mean	Std	Mean	Std	Mean	Std	Mean	Std	Mean	Std	Mean	Std	Mean	Std	Mean	Std
1	1.66E+02	7.21E+01	1.08E+04	2.77E+03	2.66E+02	1.39E+02	6.24E+01	4.07E+01	5.75E+01	5.10E+01	5.46E+02	1.48E+03	2.54E-01	1.39E-01	1.20E+01	8.43E+00	3.91E-179	0.00E+00
2	5.60E+00	1.35E+00	8.10E+01	7.18E+00	2.96E+00	1.68E+00	1.72E+00	3.23E-01	2.41E+00	1.71E+00	9.25E-01	1.61E+00	2.97E+00	2.11E+00	1.69E+00	7.48E-01	4.62E-90	2.43E-89
3	3.02E+03	1.60E+03	2.33E+05	8.01E+04	4.15E+03	1.77E+03	1.17E+03	6.08E+02	1.26E+03	1.40E+03	1.04E+04	3.68E+04	1.37E+01	1.03E+01	3.42E+02	4.17E+02	1.29E-176	0.00E+00
4	1.89E+01	3.36E+00	7.02E+01	6.66E+00	1.89E+01	3.26E+00	3.62E+01	5.87E+00	1.78E+00	4.18E+00	9.76E-02	2.95E-01	2.61E+01	5.28E+00	8.44E+00	2.37E+00	2.40E-97	9.02E-97
5	6.57E+03	4.39E+03	1.21E+07	4.85E+06	6.25E+04	8.14E+04	6.62E+03	9.68E+03	5.63E+03	5.40E+03	1.59E+07	4.59E+07	3.66E+02	1.33E+02	7.71E+02	6.72E+02	4.28E+01	1.34E-01
6	1.60E+02	8.19E+01	1.06E+04	3.00E+03	2.87E+02	2.01E+02	5.98E+01	2.69E+01	4.91E+02	4.53E+02	2.52E+03	5.75E+03	6.33E+01	5.98E+01	7.89E+01	9.36E+01	0.00E+00	0.00E+00
7	1.86E-01	5.95E-02	9.28E+00	2.98E+00	2.13E-01	8.23E-02	1.85E-01	9.85E-02	2.99E-01	1.10E-01	4.31E+00	2.06E+01	6.56E-01	2.99E-01	8.51E-02	5.34E-02	7.30E-05	8.05E-05
8	4.45E+01	5.25E+01	1.44E+02	1.20E+02	3.15E+03	6.89E+02	1.18E-02	3.20E-02	2.45E+00	2.29E+00	1.89E+02	1.41E+02	8.15E+03	1.20E+03	5.30E+03	1.70E+03	3.20E+02	9.57E+02
9	1.55E+02	2.15E+01	2.02E+02	2.02E+01	1.45E+01	4.64E+00	1.41E+02	1.93E+01	5.28E+01	2.96E+01	4.34E+01	5.74E+01	8.33E+01	2.35E+01	1.39E+02	2.94E+01	0.00E+00	0.00E+00
10	6.86E+00	2.16E+00	1.30E+01	8.86E-01	2.60E+00	6.15E-01	4.05E+00	7.56E-01	5.90E+00	1.12E+00	2.69E+00	4.02E+00	1.39E+01	2.38E+00	9.43E-01	1.17E+00	8.88E-16	0.00E+00
11	2.48E+00	9.88E-01	8.87E+01	2.46E+01	3.10E+00	1.34E+00	1.53E+00	2.75E-01	1.59E+00	6.98E-01	1.74E+01	8.17E+01	3.17E-01	2.23E-01	1.08E+00	1.16E-01	0.00E+00	0.00E+00
12	1.97E+00	8.12E-01	8.86E+06	7.99E+06	7.38E+02	2.68E+03	1.82E+00	8.42E-01	4.16E+00	1.60E+00	8.52E+06	3.15E+07	7.08E+00	2.51E+00	9.60E-01	6.58E-01	1.74E-03	4.76E-03
13	9.70E+00	5.10E+00	2.94E+07	1.86E+07	1.82E+04	9.96E+04	7.22E+00	3.54E+00	6.21E+01	2.85E+01	1.43E+07	2.87E+07	1.51E+02	4.06E+01	3.07E+00	2.13E+00	2.22E+00	2.26E+00
14	4.65E+00	6.38E-01	1.61E+01	1.59E+00	3.32E+00	5.56E-01	4.09E+00	4.37E-01	4.46E+00	9.32E-01	1.52E+00	2.66E+00	8.18E+00	1.55E+00	2.84E+00	4.45E-01	0.00E+00	0.00E+00
15	1.08E+02	3.91E+01	3.35E+02	3.53E+01	9.74E+01	1.99E+01	1.56E+02	3.28E+01	2.08E+01	5.67E+00	8.28E+00	3.27E+01	1.79E+01	8.70E+00	1.78E+00	1.65E+00	1.21E-180	0.00E+00
16	7.44E-06	9.82E-06	4.99E-01	4.92E-01	5.75E-03	2.61E-02	1.91E-09	2.39E-09	2.51E-14	4.83E-14	6.46E-01	3.54E+00	6.94E-01	5.10E-01	9.75E-01	1.77E+00	1.02E-174	0.00E+00
17	2.10E+02	8.19E+01	1.14E+04	3.02E+03	7.22E+02	4.44E+02	6.50E+01	3.34E+01	2.32E+02	1.99E+02	3.66E+03	7.80E+03	4.79E-01	2.76E-01	1.36E+02	1.11E+02	3.17E+02	1.31E+02
18	1.38E+05	6.44E+04	9.63E+06	2.24E+06	2.01E+05	1.07E+05	5.83E+04	3.88E+04	5.78E+04	4.70E+04	2.25E+06	5.26E+06	4.81E+03	3.13E+03	1.11E+04	9.24E+03	8.82E-181	0.00E+00
19	6.77E-03	4.65E-03	1.66E-01	5.39E-02	8.22E-03	4.16E-03	9.42E-03	6.79E-03	3.57E-03	3.32E-03	8.46E-02	1.26E-01	5.86E-03	1.03E-02	5.47E-04	5.61E-04	0.00E+00	0.00E+00
20	2.84E-01	1.39E-01	3.92E+00	6.24E-01	8.59E-02	6.11E-02	6.24E-02	3.45E-02	4.03E-01	2.13E-01	4.03E-01	1.39E+00	1.45E+00	4.87E-01	1.00E-01	1.43E-01	0.00E+00	0.00E+00

Table 2
Schematic view of superior, inferior and equal to CoBSA with other compared algorithm.

	BSA	ABSA	IBSA	hDEBSA	HBSA	OBSA	MLBSA	BSA_SRL
Superior	18	19	20	18	18	19	19	19
inferior	2	1	0	2	2	1	1	1
equal	0	0	0	0	0	0	0	0

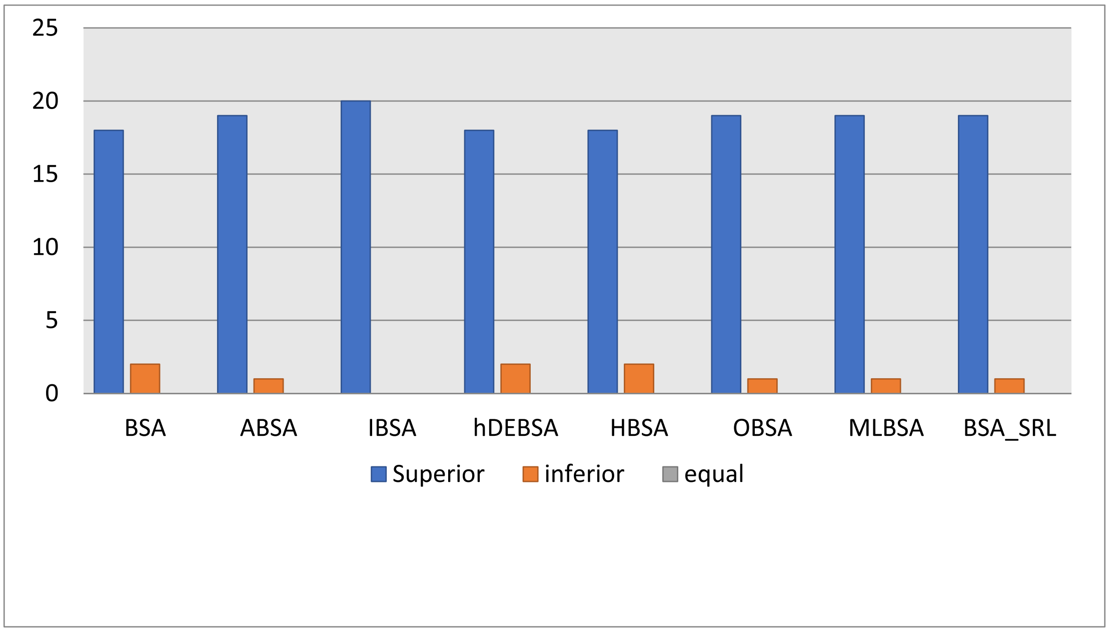

Fig. 3. Schematic view showing performance comparison of CoBSA and other algorithms.

$\sum_{i=1}^{n}(R_i - \overline{R})^2$. Then, Kendall's W is calculated by $W = \frac{12S}{m^2(n^3-n)}$.

When the value of statistic W is 1, then all of the judges or survey respondents are in agreement, and the list of objects or concerns is arranged in the same order by each judge or respondent. If W is 0, the respondents' responses are essentially random because there is no broad pattern of agreement among them. W scores in the middle range suggest a more significant or smaller degree of agreement between the judges or responders.

Given its widespread use, the calculation of the W value for this test is typically done by SPSS statistical software packages. The results of Kendall's W rank test regarding the 45 dimensions of required test functions and the mean fitness values over 30 runs are provided in Table 3. It was realized that the value of Kendall's W is 0.488, which is between 0 and 1. This ensures that the results obtained by different algorithms agree with each other reasonably. In Table 3, the (column) mean ranks indicate which algorithm was rated most favourably, where CoBSA achieves the first rank in Kendall's W test. Accordingly, the proposed CoBSA can be seen as an improved approach to the many comparable methods for finding the outcome of an optimization problem with high precision.

4.2.4. *Scalability analysis*

This subsection examines the scalability of the suggested CoBSA, which is essential because real-world situations frequently involve a large number of variables. To do so, test functions F8 and F12 were used, each containing a different set of parameters. The number of parameters was modified in steps of 200 from 100 to 1000. Fig. 6 demonstrates how the value of the test function changes as the quantity of decision variables increases and the performance of CoBSA declines. This is predictable, given that the number of possible individuals was set when executing the method. The important finding in this figure is that solving problems containing a large number of variables does not significantly affect CoBSA's efficiency. By increasing the number of variables, the outcomes can be enhanced.

In summary, the previous subsection used a variety of test functions with different dimensions to quantitatively and subjectively benchmark and confirm CoBSA's super execution ability. CoBSA was found to have a high level of exploration and escape from local optima. This can be explained by the solutions in CoBSA that tend to cooperate with one another, making it difficult for them to gravitate toward a local solution. According to the simulated individuals, CoBSA can explore the domain field and eventually advance near the global optimum. This can be attributed to the fact that in each to the fact that in each iteration, the best solution achieved thus far is saved, along with the tendency of solutions to gravitate towards the optimum. Individual connections also pull the population in the direction of the global optimum. Because of the use of centroid opposition, multiple learning, and chaotic elitism mechanisms, the entire population converges near the optimum location proportionate to the generation number. Furthermore, CoBSA successfully balances exploration and exploitation, which can be contributed to the integrated multiple learning that prioritizes examination in the primary optimization stages while emphasizing exploitation in later iterations.

4.2.5. *Sensitivity analysis of population size*

The impacts of population size on the algorithm were evaluated using the parameter sensitivity test. The population was divided into groups of 100, 200, 400, 600, 800, and 1000 individuals. In this work, the F8, F12, F13, and F17 test functions were considered to examine the synergistic influence of the two parameters on the method in order to investigate the role of populations and fitness evaluation on CoBSA. Fig. 7 shows that as the population size grew, the averages of function values improved for F8 and F13 but decreased for F12 and F17. This can be attributed to the enhanced search efficiency with larger population

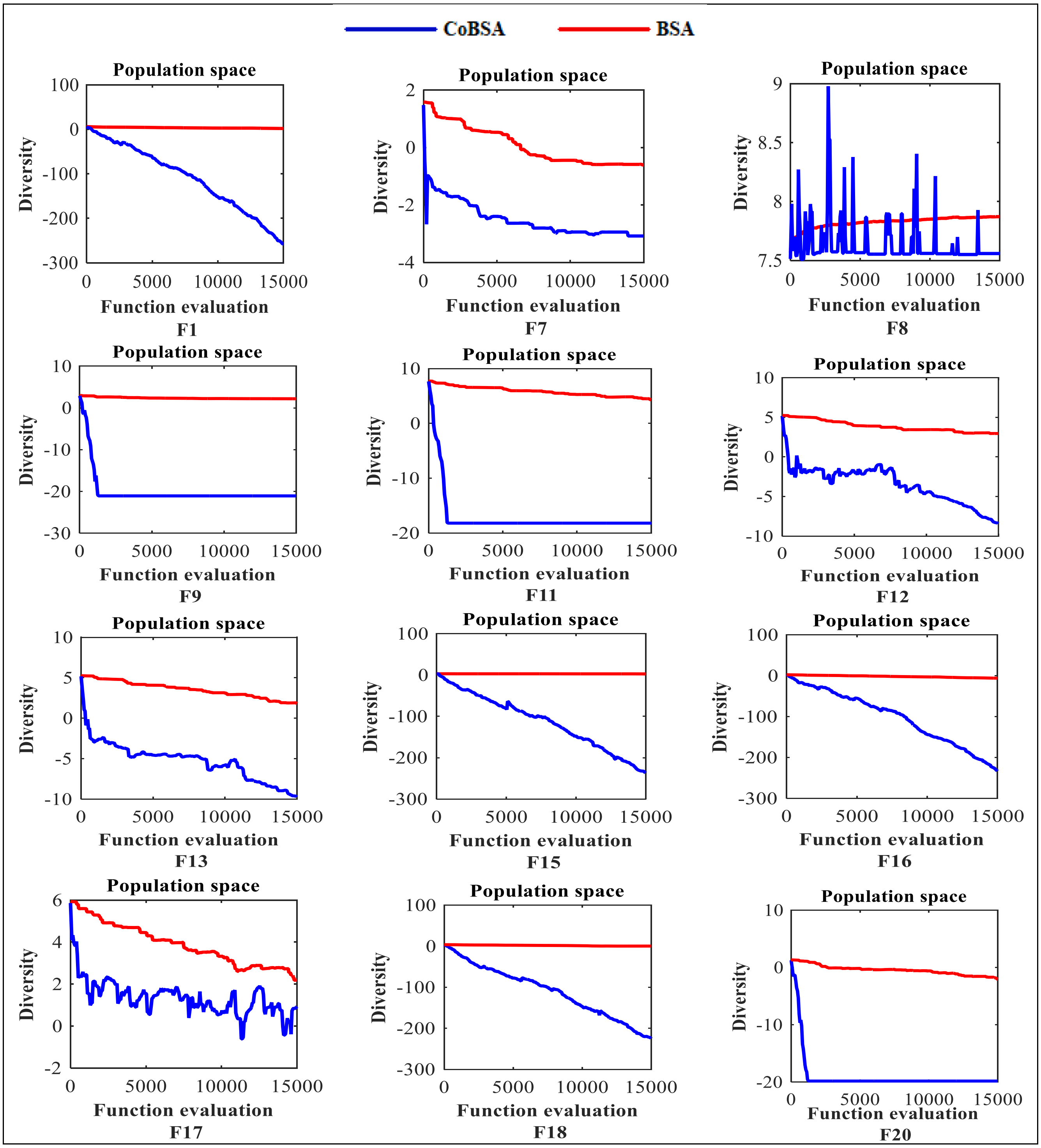

Fig. 4. Diversity curve of some selected functions.

sizes, resulting in shorter search times and greater accuracy in subsequent examinations. However, because the global approximation optimum was primarily established, the results did not grow correspondingly as the set of individuals and iterations increased. Based on particular challenges, scientists can choose the appropriate population and fitness evaluation (iterations).

4.3. Comparison of the proposed CoBSA and other metaheuristic algorithms

CoBSA is compared to 15 other algorithms, including well-established and recent methods. The current approaches include the Sine Cosine Algorithm (SCA), Salp Swarm Algorithm (SSA), Moth-flame Optimization (MFO), Spherical Search Optimizer (SSO), Tunicate Swarm Algorithm (TSA), and Seagull Optimization Algorithm (SOA). The well-established methods include Particle Swarm Optimization (PSO), Firefly Algorithm (FA), Bat Algorithm (BA), Differential

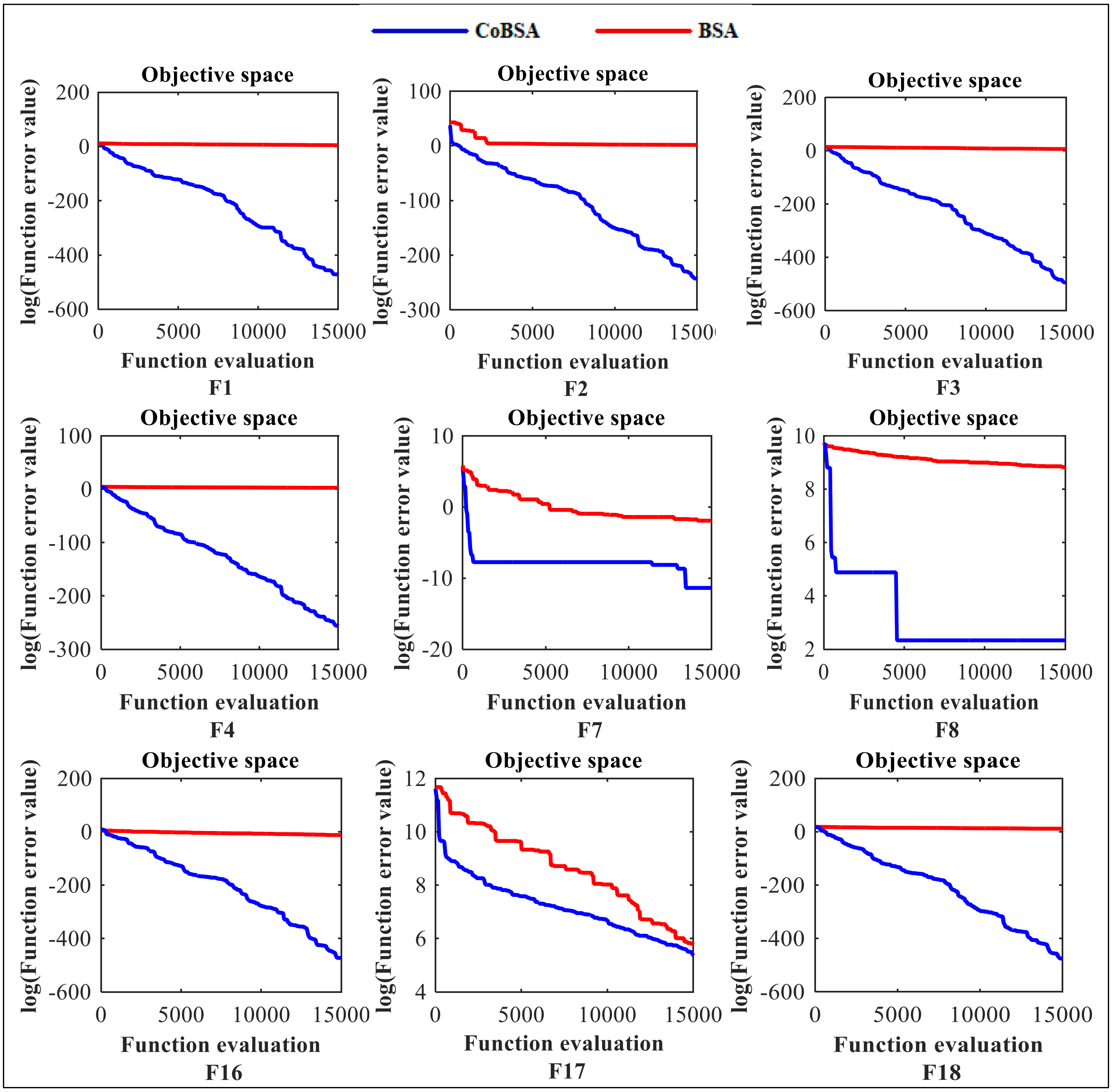

Fig. 5. Convergence graph of some selected function.

Table 3
Result of Kendall's W rank test obtained by mean result of 20 test functions for the algorithm BSA, ABSA, IBSA, hDEBSA, HBSA, OBSA, MLBSA, BSA_SRL, CoBSA.

Algorithm	Mean rank	Final rank
BSA	5.48	6
ABSA	8.5	9
IBSA	5.72	7
hDEBSA	4.35	3
HBSA	4.62	4
OBSA	6.18	8
MLBSA	5.1	5
BSA_SRL	3.55	2
CoBSA	1.5	1
W	0.488	

Evolution (DE), Flower Pollination Algorithm (FPA), Cuckoo Search (CS), and Artificial Bee Colony (ABC). The parameter settings of each algorithm were set identically to those in their original article. For all compared algorithms, the set of individuals was set to 30 and 15,000 function evaluations were used.

Tables 4 and 5 present the comparison outcomes of CoBSA with other conventional metaheuristic algorithms (MAs). Table 6 shows that CoBSA outperformed BA, CS, FA, FPA, PSO, DE, and ABC on test functions 20, 20, 20, 20, 18, 16, and 17, respectively. Table 7 reveals that CoBSA outperformed MFO, SCA, SSO, TSA, SSA, and SOA on test functions 20, 20, 20, 20, 20, 19, and 19, respectively. The comparison results suggest that the ensemble operator improves the solution execution process, allowing the optimization method to escape local minima and globally converge to the best solution. According to the comparison data, CoBSA considerably upgraded the performance of the BSA method and obtained better results than the other state-of-the-art optimization

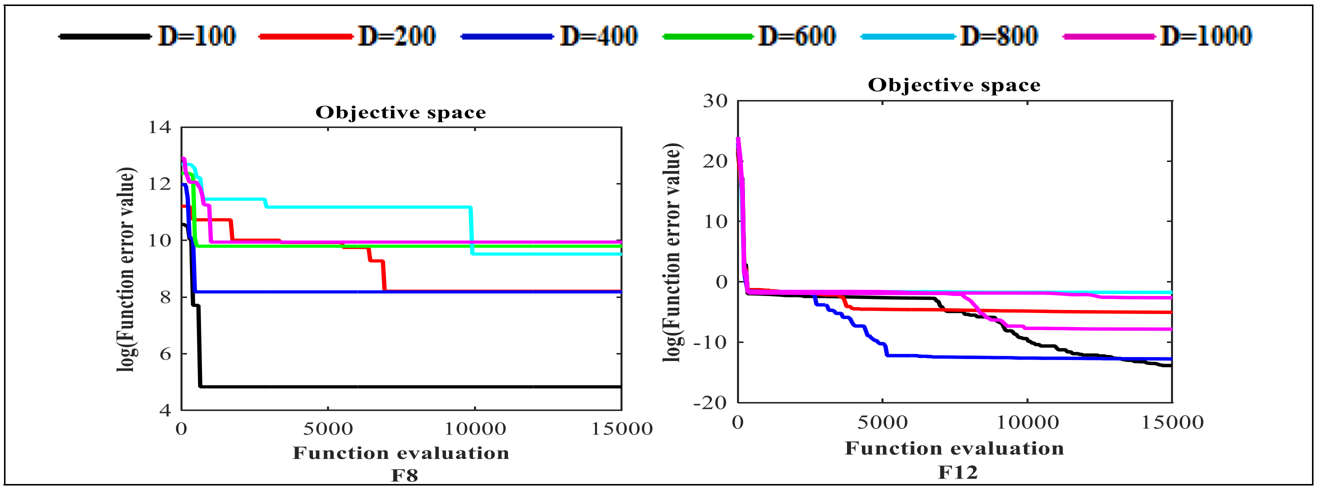

Fig. 6. CoBSA's performance on F8 and F12 functions with increasing dimension and function evaluation.

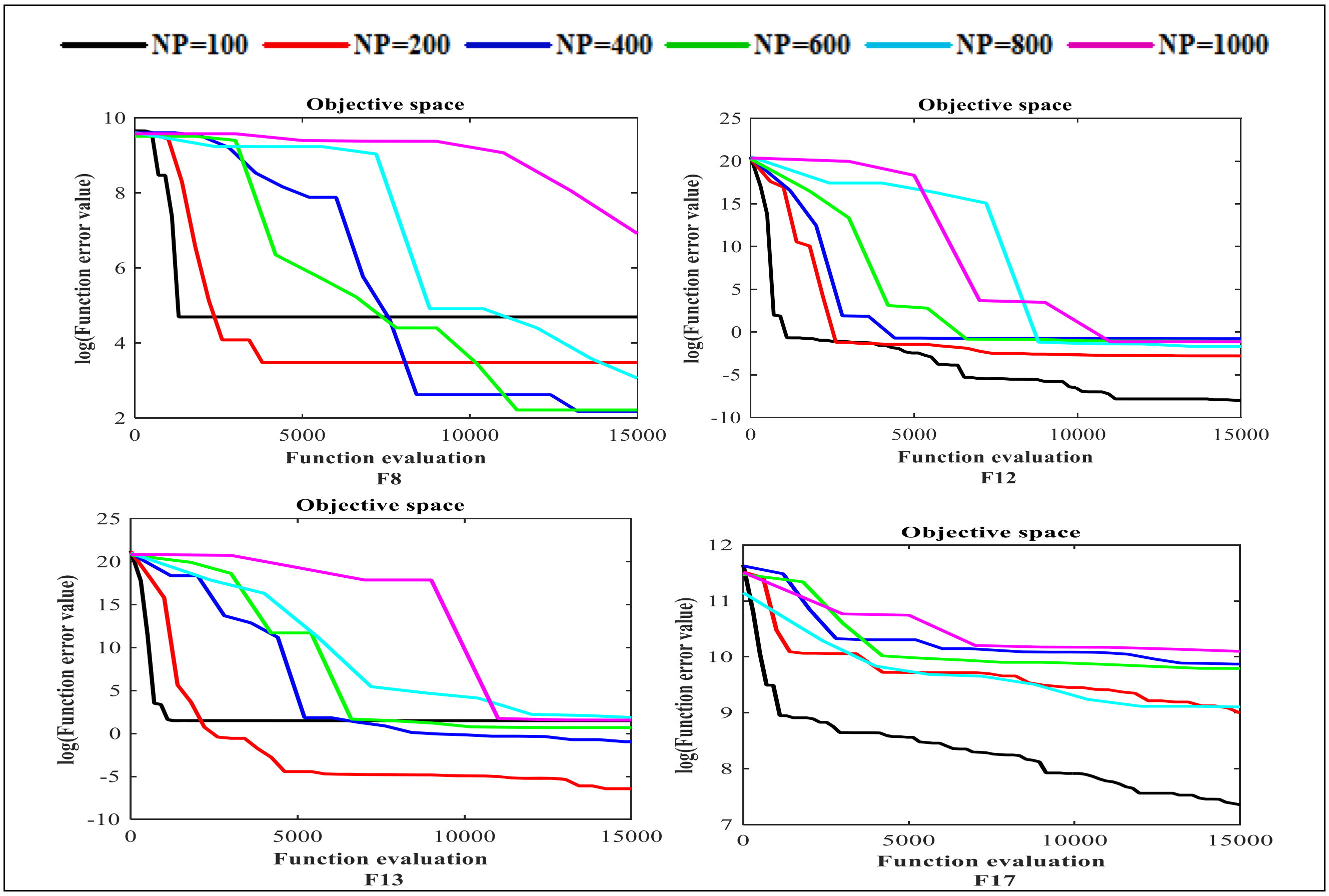

Fig. 7. Variation of population size on some selected functions.

methods.

Tables 8 and 9 summarize Kendall's W rank test findings. Kendall's W values were determined to be 0.563 and 0.654, respectively, which are between 0 and 1. This ensures that the results obtained by various algorithms are reasonably consistent. It is also evident that CoBSA achieved first rank in Kendall's W test and, thus, outperformed its competitors in the test functions. These findings further confirm that CoBSA ranked first and, therefore, can be seen as a superior analyzer to other comparative methods in terms of finding the outcome of an optimization problem with high precision.

4.4. Analysis to large scale global optimization

Large Scale Global Optimization refers to the process of optimizing complex systems that involve a large number of variables. The key challenge lies in efficiently exploring the vast solution space to identify the optimal solution or a set of near-optimal solutions. To analyse it, in this work, thirteen well-known test functions have been considered from Appendix A, in which F1-F7 are unimodal functions, and F8–F13 are multimodal functions. To tackle such complex optimization problems presented in Appendix A (F1–F13), the proposed CoBSA has been employed.

Table 10 presents the performance results of thirteen test functions (F1-F13) in 100 dimensions and D*100 function evaluations execution over 30 runs with a population size of 50, expressed in the form of Mean ± Standard Deviation. The performance results are compared with other algorithms such as mISOS, BSA, ABSA, CLPSO, CPSO—H, FDR-PSO, FI-PS, UPSO, EPSDE, TSDE, CPI-DE, ACoS-PSO, and HBSA. The performance results of all algorithms except the proposed CoBSA method are taken from [40], which are illustrated in Table 10. Across functions, F1-F7, CoBSA demonstrates significantly lower mean values and standard deviations compared to its competitors. For instance, in functions F1-F4 and F7, CoBSA achieves notably smaller mean values and narrower confidence intervals, indicating its superior convergence to optimal solutions. Furthermore, in functions F8-F13, CoBSA continues to exhibit competitive performance, maintaining low mean values and standard deviations, which underscores its robustness and adaptability across different optimization landscapes. In contrast, the comparisons with other state-of-the-art algorithms show higher mean values and wider standard deviations, indicating less reliable performance and potentially inferior convergence. These consistent results across various test functions underscore the efficacy of CoBSA in achieving high-quality solutions with reliable convergence characteristics. Thus, CoBSA emerges as a promising and effective optimization method, demonstrating its superiority and potential as a preferred choice for solving complex optimization problems in high-dimensional spaces.

Table 11 presents the results of Kendall's W rank test comparing the performance of various optimization algorithms, including CoBSA, across 13 test functions with 100 dimensions. CoBSA emerges as the top-performing algorithm with a mean rank of 1.12 and a final rank of 1, indicating its consistent superiority over other compared algorithms across the test functions. In contrast, other algorithms like mISOS, FDR-PSO, and HBSA also demonstrate competitive performance, but CoBSA consistently outranks them. Algorithms like BSA, ABSA, and FI-PS exhibit comparatively poorer performance, with higher mean and final ranks. The consistency of CoBSA's top ranking across multiple test functions indicates its robustness and adaptability to different optimization scenarios. Its superiority over other state-of-the-art algorithms also signifies its potential as a preferred choice for solving high-dimensional optimization problems.

Overall, the results strongly support the efficacy of the proposed CoBSA method, positioning it as a promising solution for a wide range of optimization tasks.

4.5. *Performance analysis of the CEC20 test functions*

The Institute of Electrical and Electronics Engineers Congress on Evolutionary Computation (IEEE CEC2020) [58] benchmark set was considered to further demonstrate the CoBSA's efficiency. This objective function set is separated into four types, namely Unimodal Functions (F1), Simple Multimodal Functions (F2-F4), Hybrid Functions (F5-F7), and Composition Functions (F8-F10), which are discussed in detail in reference [58]. For experimental analysis, the extreme quantity of fitness value calculation was set to 5×10^4, 10^6, and 3×10^6; the set of individuals' size was set to 50; and dimensions were set respectively at 5, 10, and 15. For the execution of every method, the run time was set to 30 for each test function. The empirical statistical analyses were performed using Kendall's W rank test [57].

The suggested CoBSA was contrasted with nine algorithms on the IEEE CEC 2020 objective function, including Cumulative Population Distribution Information in Differential Evolution (CPI-DE) [59], Dynamic Neighborhood Learning-based Particle Swarm Optimizer (DNLPSO) [60], Firebug Swarm Optimization (FSO) [61], MFO [50], SCA [51], Spotted Hyena Optimization (SHO) [62], SOA [55], SSA [54], and TSA [53]. The stated algorithms' parameter settings were set identically to those in their original article.

Tables 12–14 show the comprehensive comparison results for

Table 4
Performance results of BA, CS, FA, FPA, PSO, DE,ABC and proposed CoBSA on twenty well known test function with dimension 45.

F	BA		CS		FA		FPA		PSO		DE		ABC		CoBSA	
	Mean	Std	Mean	Std	Mean	Std	Mean	Std	Mean	Std	Mean	Std	Mean	Std	Mean	Std
1	7.55E+04	1.20E+04	6.04E+02	1.75E+02	5.88E+04	5.57E+03	2.54E+03	5.62E+02	2.10E+02	1.10E+02	3.62E-01	9.71E-02	4.99E-02	1.32E-01	3.91E-179	0.00E+00
2	1.83E+07	8.94E+07	1.60E+01	2.77E+00	2.62E+10	5.65E+10	3.92E+01	6.06E+00	4.90E+00	1.82E+00	1.43E-01	2.43E-02	6.28E-02	3.70E-02	4.62E-90	2.43E-89
3	1.52E+06	2.63E+05	1.04E+04	2.51E+03	1.25E+06	1.13E+05	5.36E+04	1.24E+04	3.69E+03	1.81E+03	5.84E+00	1.76E+00	1.43E-01	1.66E-01	1.29E-176	0.00E+00
4	7.75E+01	2.86E+00	2.49E+01	3.01E+00	7.37E+01	3.34E+00	2.25E+01	2.34E+00	4.49E+01	5.64E+00	2.93E+01	2.54E+00	7.48E+01	4.79E+00	2.40E-97	9.02E-97
5	6.17E+07	1.97E+07	8.45E+04	5.73E+04	1.49E+08	2.32E+07	4.23E+05	1.79E+05	7.00E+04	5.84E+04	5.91E+02	1.38E+02	1.80E+02	1.42E+02	4.28E+01	1.34E-01
6	8.97E+04	7.73E+03	6.01E+02	1.94E+02	5.74E+04	4.58E+03	2.67E+03	6.43E+02	2.36E+02	1.12E+02	0.00E+00	0.00E+00	3.10E+00	2.50E+00	0.00E+00	0.00E+00
7	3.20E-01	1.25E-01	3.31E-01	1.19E-01	1.96E-01	5.10E-02	4.56E-01	2.01E-01	5.16E-01	1.33E-01	1.40E-01	2.69E-02	7.39E-01	2.06E-01	7.30E-05	8.05E-05
8	1.43E+04	4.00E+02	1.23E+04	7.13E+02	1.55E+04	3.90E+02	1.10E+04	2.65E+02	2.72E+02	1.84E+02	1.27E-04	3.14E-13	1.97E+01	4.49E+01	3.20E+02	9.57E+02
9	2.43E+02	3.64E+01	3.23E+02	2.52E+01	4.80E+01	1.07E+01	3.00E+02	1.66E+01	1.87E+02	3.18E+01	1.95E+02	1.12E+01	2.54E+01	6.29E+00	0.00E+00	0.00E+00
10	1.90E+01	2.85E-01	6.76E+00	4.76E-01	1.96E+01	2.29E-01	9.97E+00	8.13E-01	5.03E+00	9.48E-01	1.77E-01	3.72E-02	2.52E+00	5.72E-01	8.88E-16	0.00E+00
11	7.98E+02	8.34E+01	6.87E+00	2.06E+00	5.29E+02	3.50E+01	2.37E+01	5.12E+00	2.58E+00	8.83E-01	4.91E-01	7.80E-02	2.30E-01	1.85E-01	0.00E+00	0.00E+00
12	1.74E+08	6.09E+07	1.61E+01	3.74E+00	2.40E+08	4.78E+07	5.62E+01	1.45E+02	3.41E+03	1.23E+04	4.53E-01	1.31E-01	3.10E-02	3.04E-02	1.74E-03	4.76E-03
13	4.79E+08	1.17E+08	9.02E+01	2.55E+01	5.44E+08	1.02E+08	1.50E+04	4.96E+04	2.84E+03	1.47E+04	1.82E-01	5.64E-02	4.30E-03	5.83E-03	2.22E+00	2.26E+00
14	3.03E+01	1.64E+00	5.13E+00	4.50E-01	2.47E+01	1.04E+00	6.17E+00	7.71E-01	4.71E+00	6.42E-01	2.54E+00	2.05E-01	8.75E+00	1.34E+00	0.00E+00	0.00E+00
15	1.25E+02	1.97E+01	1.01E+02	1.43E+01	3.17E+00	1.72E+00	3.33E+01	9.82E+00	1.45E+02	2.03E+01	1.91E+02	2.17E+01	2.67E+02	3.06E+01	1.21E-180	0.00E+00
16	1.68E-01	3.79E-01	3.08E+01	1.04E+01	2.78E-06	8.24E-06	1.44E+02	4.69E+01	2.68E-12	7.94E-12	5.39E-19	4.64E-19	1.35E-16	6.27E-17	1.02E-174	0.00E+00
17	8.52E+04	9.16E+03	1.16E+03	3.53E+02	7.16E+04	7.26E+03	9.36E+03	1.32E+03	1.45E+02	9.36E+01	3.83E-01	9.92E-02	1.16E-02	1.43E-02	3.17E+02	1.31E+02
18	1.07E+06	1.10E+06	5.44E+05	1.82E+05	4.32E+07	1.74E+07	2.45E+06	5.03E+05	1.68E+05	9.26E+04	2.86E+02	6.22E+01	6.74E+00	6.56E+00	8.82E-181	0.00E+00
19	4.16E-14	6.64E-15	3.01E-02	8.39E-03	2.43E-12	2.62E-13	1.16E-01	2.32E-02	9.55E-03	4.85E-03	1.93E-05	7.74E-06	1.62E-04	1.41E-04	0.00E+00	0.00E+00
20	3.01E+00	4.43E-01	1.60E+00	3.01E-01	5.81E-01	2.26E-01	3.22E+00	3.90E-01	8.79E-01	2.87E-01	7.59E-04	2.38E-04	1.89E-02	3.49E-02	0.00E+00	0.00E+00

Table 5
Performance results of MFO, SCA, SSO, TSA, SSA, SOA and proposed CoBSA on twenty well known test function with dimension 45.

F	MFO		SCA		SSO		TSA		SSA		SOA		CoBSA	
	Mean	Std	Mean	Std	Mean	Std	Mean	Std	Mean	Std	Mean	Std	Mean	Std
1	3.79E+03	6.47E+03	4.21E+02	5.38E+02	6.75E+02	3.47E-13	8.34E-17	1.98E-16	1.04E-01	2.59E-01	4.82E-09	6.74E-09	3.91E-179	0.00E+00
2	6.38E+01	2.24E+01	3.80E-01	4.18E-01	1.47E+01	5.42E-15	4.39E-11	6.00E-11	7.14E+00	3.17E+00	8.37E-07	5.15E-07	4.62E-90	2.43E-89
3	2.10E+05	1.69E+05	8.11E+03	1.20E+04	1.81E+04	3.70E-12	1.68E-15	3.10E-15	1.57E+03	9.55E+02	1.31E-07	2.14E-07	1.29E-176	0.00E+00
4	8.18E+01	4.65E+00	6.22E+01	8.54E+00	6.44E+01	4.34E-14	6.10E+00	3.30E+00	1.81E+01	3.36E+00	2.56E+00	6.04E+00	2.40E-97	9.02E-97
5	1.40E+07	3.02E+07	3.76E+06	6.78E+06	2.12E+04	1.11E-11	4.33E+01	6.70E-01	1.49E+03	4.09E+03	4.33E+01	4.92E-01	4.28E+01	1.34E-01
6	6.07E+03	7.25E+03	4.08E+02	5.78E+02	7.39E+02	0.00E+00	2.00E+00	4.08E+00	7.20E+01	3.88E+01	0.00E+00	0.00E+00	0.00E+00	0.00E+00
7	1.36E+01	1.87E+01	1.54E+00	1.31E+00	1.19E+00	4.52E-16	1.62E-02	6.48E-03	4.27E-01	1.16E-01	4.62E-03	2.74E-03	7.30E-05	8.05E-05
8	6.79E+03	1.29E+03	1.43E+04	4.07E+02	6.74E+03	2.78E-12	1.09E+04	7.33E+02	7.99E+03	1.15E+03	1.23E+04	8.46E+02	3.20E+02	9.57E+02
9	2.79E+02	3.29E+01	1.20E+02	6.04E+01	1.77E+02	1.45E-13	3.30E+02	6.87E+01	8.04E+01	1.88E+01	5.70E+00	1.73E+01	0.00E+00	0.00E+00
10	1.96E+01	6.09E-01	1.60E+01	7.27E+00	7.96E+00	4.52E-15	7.08E-01	1.32E+00	3.82E+00	8.72E-01	2.00E+01	1.73E-03	8.88E-16	0.00E+00
11	7.75E+01	8.15E+01	5.14E+00	4.04E+00	6.79E+00	4.52E-15	1.01E-02	1.20E-02	1.76E-01	8.70E-02	1.25E-02	2.56E-02	0.00E+00	0.00E+00
12	8.82E+06	4.68E+07	7.35E+06	1.17E+07	8.69E+00	0.00E+00	9.92E+00	4.18E+00	8.01E+00	1.39E+00	1.88E-01	1.94E-02	1.74E-03	4.76E-03
13	3.11E+07	1.04E+08	7.12E+06	1.13E+07	4.85E+01	0.00E+00	4.82E+00	8.55E-01	1.17E+02	4.44E+01	3.51E+00	1.94E-01	2.22E+00	2.26E+00
14	1.24E+01	4.53E+00	3.44E+00	1.83E+00	9.90E+00	0.00E+00	4.63E-01	6.15E-02	4.39E+00	5.50E-01	1.93E-01	3.65E-02	0.00E+00	0.00E+00
15	2.62E+02	8.97E+01	6.14E+01	2.39E+01	2.62E+02	1.73E-13	1.33E-05	2.05E-05	3.00E+01	2.43E+01	1.36E-03	2.67E-03	1.21E-180	0.00E+00
16	2.93E-15	6.09E-15	2.05E-14	8.60E-14	1.60E-07	0.00E+00	8.71E-43	3.23E-42	2.23E-11	1.39E-11	1.93E-27	8.46E-27	1.02E-174	0.00E+00
17	1.24E+04	1.01E+04	1.99E+04	2.34E+03	8.86E+02	4.63E-13	1.26E+04	2.38E+03	9.58E-02	1.81E-01	1.47E+04	2.06E+03	3.17E+02	1.31E+02
18	6.10E+06	8.12E+06	4.53E+05	6.53E+05	6.80E+05	1.18E-10	7.62E-14	1.26E-13	6.60E+01	7.15E+01	4.06E-06	6.13E-06	8.82E-181	0.00E+00
19	1.87E-01	2.26E-01	3.64E-02	5.08E-02	2.39E-02	0.00E+00	2.92E-16	5.44E-17	6.35E-06	1.34E-05	4.27E-13	1.06E-12	0.00E+00	0.00E+00
20	1.94E+00	1.02E+00	1.34E-01	2.18E-01	1.87E+00	6.78E-16	2.15E+00	1.68E+00	1.79E+00	4.33E-01	2.68E-12	5.41E-12	0.00E+00	0.00E+00

Table 6
Schematic view of superior, inferior and equal to CoBSA with other compared algorithm BA, CS, FA, FPA, PSO, DE,ABC.

	BA	CS	FA	FPA	PSO	DE	ABC
Superior	20	20	20	20	18	16	17
inferior	0	0	0	0	2	2	3
equal	0	0	0	0	0	1	0

Table 7
Schematic view of superior, inferior and equal to CoBSA with other compared algorithm MFO, SCA, SSO, TSA, SSA, SOA.

	MFO	SCA	SSO	TSA	SSA	SOA
Superior	20	20	20	20	19	19
inferior	0	0	0	0	1	0
equal	0	0	0	0	0	1

Table 8
Result of Kendall's W rank test obtained by mean result of 20 test functions for the algorithm BA, CS, FA, FPA, PSO, DE,ABC, CoBSA.

Algorithm	Mean rank	Final rank
BA	6.7	8
CS	5.15	5
FA	6.2	7
FPA	5.9	6
PSO	4.45	4
DE	2.88	2
ABC	3.3	3
CoBSA	1.42	1
w	0.563	

Table 9
Result of Kendall's W rank test obtained by mean result of 20 test functions for the algorithm MFO, SCA, SSO, TSA, SSA, SOA, CoBSA.

Algorithm	Mean rank	Final rank
MFO	6.32	7
SCA	5.2	6
SSO	5.18	5
TSA	3.22	3
SSA	3.95	4
SOA	3.05	2
CoBSA	1.08	1
w	0.654	

dimensions 5, 10, and 15. It can be observed that CoBSA performed the best with a substantially lower mean and demonstrates a strong capacity to find optimal solutions for most functions. Table 12 reveals that CoBSA performed well on CEC20 test functions, defeating CPI-DE, SHO, MFO, and TSA on all objective functions. However, DNLPSO, FSO, SCA, SOA, and SSA outperformed CoBSA on F3, F9, F9, F2, and F3, respectively.

Tables 13 and 14 illustrate that CoBSA outperformed the compared algorithms on all test functions. Composition functions, being a fixed-dimensional multimodal function, have a huge number of local optimum solutions, necessitating a high-performance algorithm. CoBSA ranked top in composite functions with dimensions 10 and 15, indicating that its overall performance is strong enough to effectively balance exploration and exploitation.

According to Fig. 8, CoBSA had a high convergence speed in F1, F2, F3, F4, F5, F6, F8, F9, and F10 functions with a comparatively high solution accuracy. Even in the intermediate of the repetition, some algorithms reached the local optimum. While composite functions F8, F9, and F10 have a large number of local optima, surprisingly, CoBSA's convergence performance in these scenarios was superior and rapid.

The desired location may be discovered during the first cycle phase,

Table 10
Performance results of thirteen test function (F1-F13) at 100D and D*100 FEs over 30 runs with 50 population size in the form of Mean±Standard Deviation.

Function	F1	F2	F3	F4	F5	F6	F7
CoBSA	1.23e-012 ± 1.35e-011	1.75e-007 ± 1.00e-006	1.10e-009 ± 1.18e-009	3.62e-008 ± 1.19e-009	1.17e+000 ± 2.36e-002	0.00e+000 ± 0.00e+000	5.49e-007 ± 2.31e-007
mISOS	3.78e-010 ± 1.58e-010	1.92e-005 ± 1.01e-005	2.00e-008 ± 1.05e-008	9.66e-005 ± 2.79e-005	9.87e+001 ± 4.85e-002	0.00e+000 ± 0.00e+000	4.72e-003 ± 1.82e-003
BSA	1.86e+004 ± 1.48e+004	1.00e+002 ± 2.29e+001	8.22e+005 ± 5.59e+005	4.26e+001 ± 8.46e+000	8.02e+006 ± 1.38e+007	1.37e+004 ± 8.08e+003	1.48e+001 ± 1.75e+001
ABSA	1.66e+005 ± 1.37e+004	4.04e+036 ± 1.54e+037	7.78e+006 ± 7.96e+005	9.16e+001 ± 1.93e+000	5.67e+008 ± 7.97e+007	1.65e+005 ± 1.29e+004	9.05e+002 ± 1.42e+002
CLPSO	5.48e+004 ± 4.12e+003	2.85e+002 ± 1.61e+002	2.43e+006 ± 2.53e+005	7.71e+001 ± 3.87e+000	7.75e+007 ± 1.29e+007	5.29e+004 ± 4.00e+003	1.12e+002 ± 2.31e+001
CPSO—H	7.08e+002 ± 1.29e+002	1.97e+001 ± 1.87e+000	3.99e+004 ± 1.15e+004	9.39e+001 ± 1.36e+000	1.08e+006 ± 2.13e+005	7.88e+002 ± 1.82e+002	2.93e+000 ± 4.23e-001
FDR-PSO	5.34e+002 ± 1.55e+002	1.38e+001 ± 2.46e+000	2.55e+004 ± 8.15e+003	3.02e+001 ± 3.00e+000	9.01e+004 ± 3.95e+004	6.67e+002 ± 1.52e+002	1.11e+000 ± 2.20e-001
FI-PS	7.07e+004 ± 5.43e+003	2.30e+010 ± 8.74e+010	3.05e+006 ± 2.49e+005	7.85e+001 ± 2.67e+000	1.36e+008 ± 2.06e+007	6.92e+004 ± 3.57e+003	2.06e+002 ± 3.02e+001
UPSO	8.93e+003 ± 1.20e+003	1.99e+002 ± 4.36e+001	3.63e+005 ± 5.60e+004	8.59e+001 ± 3.64e+000	8.80e+006 ± 2.03e+006	8.79e+003 ± 1.45e+003	1.18e+001 ± 2.83e+000
EPSDE	7.08e+002 ± 2.01e+002	3.93e+001 ± 3.79e+001	3.19e+004 ± 9.25e+003	8.57e+001 ± 1.72e+001	9.59e+005 ± 1.85e+006	7.78e+002 ± 1.92e+002	1.23e+000 ± 1.06e+000
TSDE	1.83e+004 ±3.22e+003	1.34e+002 ± 1.75e+001	8.77e+005± 1.19e+005	5.01e+001± 4.30e+000	9.25e+006± 3.05e+006	1.91e+004± 4.67e+003	1.46e+001± 5.52e+000
CPI-DE	2.75e+005± 1.14e+004	2.88e+048± 5.44e+048	1.30e+007± 7.53e+005	9.58e+001± 1.11e+000	1.12e+009± 9.61e+007	2.66e+005± 1.14e+004	1.79e+003± 1.76e+002
ACoS-PSO	1.05e+004± 2.86e+003	3.08e+002± 2.44e+001	6.64e+005± 1.73e+005	6.39e+001± 9.06e+000	7.09e+006± 7.46e+006	1.26e+004± 5.18e+003	1.31e+001± 8.55e+000
HBSA	8.83e+002 ± 1.60e+002	1.52e+001 ± 2.78e+000	3.58e+004 ± 8.22e+003	1.78e+001 ± 1.81e+000	6.27e+004 ± 2.74e+004	8.55e+002 ± 1.95e+002	6.66e-001 ± 2.14e-001

Function	F8	F9	F10	F11	F12	F13
CoBSA	1.31e+001 ± 2.11e+001	1.11e+000 ± 2.09e+000	1.19e-007 ± 8.58e-006	1.75e-011 ± 5.13e-009	1.71e-005 ± 4.15e-001	1.02e+001 ± 3.46e-001
mISOS	3.11e+004 ± 5.19e+002	3.28e+001 ± 5.08e+001	1.85e-005 ± 7.18e-006	2.70e-009 ± 7.03e-009	1.90e-001 ± 3.87e-003	1.03e+001 ± 4.86e-001
BSA	2.68e+004 ± 6.80e+002	8.22e+002 ± 6.80e+001	1.55e+001 ± 2.18e+000	1.25e+002 ± 5.68e+001	2.18e+006 ± 7.31e+006	1.90e+007 ± 6.62e+007
ABSA	2.68e+004 ± 6.70e+002	1.20e+003 ± 4.84e+001	2.01e+001 ± 2.54e-001	1.51e+003 ± 1.20e+002	1.22e+009 ± 2.43e+008	2.42e+009 ± 3.05e+008
CLPSO	2.41e+004 ± 1.09e+003	9.65e+002 ± 3.71e+001	1.79e+001 ± 2.27e-001	4.89e+002 ± 4.36e+001	7.81e+007 ± 2.21e+007	2.27e+008 ± 4.48e+007
CPSO—H	2.06e+003 ± 3.35e+002	9.12e+001 ± 8.54e+000	9.37e+000 ± 4.98e-001	7.81e+000 ± 1.66e+000	2.22e+000 ± 3.51e+000	2.64e+001 ± 6.74e+000
FDR-PSO	1.84e+004 ± 3.15e+003	2.81e+002 ± 4.02e+001	5.15e+000 ± 5.07e-001	5.33e+000 ± 1.17e+000	1.31e+002 ± 2.58e+002	3.60e+002 ± 7.42e+001
FI-PS	2.88e+004 ± 1.14e+003	1.13e+003 ± 3.50e+001	1.81e+001 ± 2.03e-001	6.24e+002 ± 6.11e+001	2.00e+008 ± 4.19e+007	5.25e+008 ± 6.64e+007
UPSO	1.68e+001 ± 7.89e+001	7.37e+002 ± 5.05e+001	1.18e+001 ± 5.03e-001	7.83e+001 ± 9.73e+000	4.33e+006 ± 2.21e+006	1.74e+007 ± 7.67e+006
EPSDE	2.91e+000 ± 2.79e+000	7.21e+002 ± 4.52e+001	5.07e+000 ± 5.14e-001	7.82e+000 ± 2.24e+000	2.36e+006 ± 6.48e+006	8.64e+005 ± 2.51e+006
TSDE	2.83e+004± 6.82e+002	9.77e+002± 4.84e+001	1.35e+001± 8.61e-001	1.74e+002± 4.17e+001	1.72e+006± 1.65e+006	1.39e+007± 8.02e+006
CPI-DE	3.76e+004± 1.04e+003	1.62e+003± 5.17e+001	2.09e+001± 7.17e-002	2.39e+003± 1.26e+002	2.80e+009± 2.25e+008	5.19e+009± 3.92e+008
ACoS-PSO	2.13e+004± 2.59e+003	1.07e+003± 1.13e+002	1.27e+001± 1.66e+000	1.14e+002± 4.63e+001	2.45e+006± 2.72e+006	1.22e+007± 9.19e+006
HBSA	4.52e+001 ± 5.06e+001	6.51e+002 ± 4.65e+001	5.23e+000 ± 4.38e-001	8.33e+000 ± 1.94e+000	9.62e+000 ± 2.66e+000	1.12e+002 ± 3.19e+001

demonstrating that CoBSA's exploratory trends successfully circumvent slipping into local optimum. Fig. 8 further reveals that CoBSA has a good sense of stability in exploration and exploitation. Using this approach to harmonize the main searching stages can be tested in composite test functions. CoBSA produced outstanding results and satisfactory performance, according to Fig. 8.

The strong ability of CoBSA to harmonize a variety of solutions and focus on the location of superior solutions at advanced levels is the basis of its excellent efficacy. This capability can be attributed to centroid opposition-based learning, centroid and chaotic-based multiple learning, and an elite mechanism that considers the change in an individual search range during iteration. CoBSA features a handy function that ensures that the search steps are focused at a certain pace. This feature helps to thoroughly investigate the solution space while substantially evaluating the feature domain in the early phases.

Kendall's W rank test results for dimensions 5, 10, and 15 are presented in Tables 15–17, in which the value of 'w' is 0.793, 0.788, and 0.733, respectively. As these values are all less than 1, the outcomes acquired by different algorithms are expressively diverse. Also, based on the mean rank in Tables 15–17, it can be concluded that the rank follows the order: CoBSA < DNLPSO = SSA < MFO < SOA < FSO < SCA < TSA < SHO < CPI-DE; CoBSA < SSA < DNLPSO < SOA < MFO < SCA < FSO < TSA < SHO < CPI-DE; and CoBSA < DNLPSO < SSA < MFO < FSO < SOA < SCA < TSA < SHO < CPI-DE. All tables indicate that the proposed CoBSA ranks first. It can also be seen that the pyramid's height obtained by CoBSA is less than that of the other algorithms, which implies that the

Table 11
Ranking of **CoBSA** and some selected others algorithm variants by the Kendall's W rank test on 13 test functions with 100D.

Algorithm	Mean Rank	Final Rank
CoBSA	1.12	1
mISOS	2.88	2
BSA	8.71	9
ABSA	12.62	13
CLPSO	10.33	11
CPSO—H	5.38	6
FDR-PSO	4	3
FI-PS	11.75	12
UPSO	7.58	7
EPSDE	5.12	5
TSDE	9	10
CPI-DE	14	14
ACoS-PSO	8	8
HBSA	4.5	4
W		0.844

rank obtained by CoBSA is less than that of the other compared methods. In other words, the ultimate rank of CoBSA is 1; hence, CoBSA has statistical significance on all functions for each dimension.

Table 18 shows the comprehensive comparison results for CoBSA, COBSCA, COCRO, NCOGOA, and GMPBSA on CEC20 test functions for dimension 10. Compared to COBSCA, the performance of CoBSA proved to be better for nine functions except F9. CoBSA outperformed COCRO, NCOGOA, and GMPBSA on all test functions. It is known that composition functions, being fixed-dimensional multimodal functions, have a considerable number of local optimum solutions, necessitating a high-performance algorithm. CoBSA performed well on CEC20 composite functions compared to COBSCA, COCRO, NCOGOA, and GMPBSA, as these functions have a huge number of local optimum solutions. This suggests that CoBSA demonstrates a superior ability to balance between exploration and exploitation compared to the algorithms COBSCA, COCRO, NCOGOA, and GMPBSA.

For the statistical analysis, the results of Kendall's W rank test are presented in Table 19, where the value of 'w' is 0.651. Also, the mean rank of CoBSA is less than that of COBSCA, COCRO, NCOGOA, and GMPBSA, and the order of rank is as follows: CoBSA $<$ GMPBSA $<$ COCRO $<$ NCOGOA $<$ COBSCA. So, the analysis indicates that the rank of CoBSA is 1, and hence, CoBSA has statistical significance on all test functions for dimension 10.

5. Application of CoBSA to engineering problems

In real-world optimization scenarios, most problems have constraints. Constraint processing is the procedure of considering equality

Table 12
Results of IEEE CEC20 test function with $D = 5$ obtained by CoBSA, CPI-DE, DNLPSO, FSO, MFO, SCA, SHO, SOA, SSA, and TSA.

F	Algorithm	CoBSA	CPI-DE	DNLPSO	FSO	MFO	SCA	SHO	SOA	SSA	TSA
1	Mean	9.11E-03	1.89E+09	6.04E+03	3.08E+03	3.44E+03	7.10E+06	1.11E+09	1.62E+04	3.42E+03	1.09E+08
	SD	4.85E-02	8.17E+08	8.83E+03	1.66E+03	4.89E+03	5.00E+06	7.60E+08	1.11E+04	3.20E+03	2.81E+08
2	Mean	1.58E+02	1.17E+03	1.75E+02	3.04E+02	1.76E+02	1.80E+02	8.87E+02	1.09E+02	1.70E+02	3.22E+02
	SD	1.15E+02	1.34E+02	1.11E+02	1.69E+02	1.29E+02	8.82E+01	1.76E+02	8.83E+01	1.18E+02	2.33E+02
3	Mean	9.24E+00	1.08E+02	7.13E+00	1.05E+01	1.07E+01	1.40E+01	4.02E+01	1.19E+01	8.98E+00	1.59E+01
	SD	3.79E+00	2.48E+01	1.98E+00	3.77E+00	4.46E+00	2.57E+00	7.67E+00	3.82E+00	2.97E+00	6.12E+00
4	Mean	2.48E-01	2.00E+04	2.96E-01	1.20E+00	4.20E-01	1.90E+00	3.17E+03	7.73E-01	3.60E-01	3.81E+02
	SD	2.13E-01	2.28E+04	8.34E-02	8.36E-01	2.39E-01	4.91E-01	7.46E+03	3.29E-01	1.91E-01	1.15E+03
5	Mean	1.83E-01	7.36E+06	1.10E+02	1.50E+02	3.29E+01	3.46E+02	5.95E+05	2.84E+02	8.74E+02	1.19E+03
	SD	3.49E-01	1.41E+07	1.95E+02	1.73E+02	3.33E+01	2.52E+02	1.90E+06	1.63E+02	1.17E+03	2.25E+03
6	Mean	1.16E-01	2.69E+02	6.45E-01	2.02E+01	2.16E+00	2.14E+00	1.34E+02	2.31E+00	1.77E+00	3.95E+01
	SD	1.53E-01	1.15E+02	3.21E-01	2.73E+01	5.49E+00	6.07E-01	8.51E+01	3.28E+00	3.51E+00	5.35E+01
8	Mean	2.55E+00	4.51E+02	1.96E+01	5.61E+01	9.18E+00	1.98E+01	2.23E+02	9.69E+00	9.10E+00	9.12E+01
	SD	3.89E+01	2.37E+02	3.74E+01	4.83E+01	6.24E+00	5.19E+00	1.12E+02	7.76E+00	6.89E+00	6.55E+01
9	Mean	1.38E+02	3.82E+02	1.83E+02	1.11E+02	1.67E+02	1.05E+02	2.44E+02	1.87E+02	1.73E+02	2.53E+02
	SD	6.62E+01	3.71E+01	1.06E+02	2.92E+01	9.12E+01	2.03E+00	7.38E+01	1.04E+02	1.85E+01	1.01E+02
10	Mean	3.44E+02	5.05E+02	3.46E+02	3.48E+02	3.47E+02	3.48E+02	4.42E+02	3.46E+02	3.46E+02	3.57E+02
	SD	1.20E+01	9.87E+01	8.65E+00	1.30E-01	1.46E-02	2.89E-01	5.42E+01	8.44E+00	6.40E+01	2.47E+01

Table 13
Results of IEEE CEC20 test function with $D = 10$ obtained by CoBSA, CPI-DE, DNLPSO, FSO, MFO, SCA, SHO, SOA, SSA, TSA.

F	Algorithm	CoBSA	CPI-DE	DNLPSO	FSO	MFO	SCA	SHO	SOA	SSA	TSA
1	Mean	8.43E+02	1.71E+10	7.64E+03	3.60E+07	6.21E+03	2.79E+08	1.36E+10	2.33E+08	4.17E+03	1.96E+09
	SD	1.13E+03	4.65E+09	7.33E+03	5.08E+07	3.90E+03	1.26E+08	1.47E+09	3.61E+08	3.56E+03	2.23E+09
2	Mean	6.24E+02	2.55E+03	2.13E+02	8.46E+02	9.54E+02	6.46E+02	2.37E+03	5.05E+02	5.23E+02	6.60E+02
	SD	2.41E+02	3.55E+02	9.21E+01	3.70E+02	3.78E+02	8.51E+01	3.73E+02	1.65E+02	1.59E+02	7.69E+01
3	Mean	2.03E+01	4.72E+02	3.57E+01	4.66E+01	3.82E+01	5.52E+01	1.55E+02	4.69E+01	2.67E+01	8.14E+01
	SD	1.98E+01	2.70E+01	2.01E+00	1.88E+01	5.19E+00	6.77E+00	2.21E+01	7.28E+00	7.06E+00	2.12E+01
4	Mean	1.65E+00	3.92E+06	9.76E-01	8.21E+00	1.84E+00	8.61E+00	2.03E+05	1.13E+00	7.71E-01	7.18E+03
	SD	9.22E-01	1.99E+06	4.29E-01	6.49E+00	7.96E-01	1.50E+00	1.44E+05	5.97E-01	4.57E-01	6.85E+03
5	Mean	1.70E+03	2.37E+07	1.80E+04	1.02E+05	2.97E+04	7.21E+03	3.26E+06	6.03E+03	1.77E+03	2.67E+05
	SD	1.47E+03	1.21E+07	3.18E+04	4.81E+04	4.01E+04	4.73E+03	2.69E+06	2.73E+03	1.01E+03	4.09E+05
6	Mean	3.87E+00	1.52E+02	5.62E-01	1.84E+01	2.67E+00	1.12E+00	1.49E+02	7.61E-01	8.67E-01	2.99E+01
	SD	7.50E+00	1.28E+02	1.40E-01	1.63E+00	1.91E+00	1.63E-01	9.19E+01	2.54E-01	3.56E-01	2.72E+01
7	Mean	3.55E+00	9.66E+06	5.35E+01	4.09E+03	2.29E+03	2.04E+03	2.38E+06	1.22E+03	3.58E+02	7.07E+03
	SD	7.40E+00	1.34E+07	1.10E+02	3.84E+03	2.62E+03	1.82E+03	2.26E+06	8.81E+02	2.73E+02	5.97E+03
8	Mean	1.22E+01	1.99E+03	8.41E+01	1.26E+02	1.02E+02	1.23E+02	1.31E+03	2.89E+02	8.34E+01	2.85E+02
	SD	2.00E+01	3.97E+02	3.81E+01	1.96E+01	1.46E+00	2.63E+01	1.53E+02	5.51E+02	4.66E+01	1.49E+02
9	Mean	3.63E+02	6.00E+02	3.85E+02	4.10E+02	3.10E+02	2.27E+02	5.52E+02	3.35E+02	3.74E+02	4.10E+02
	SD	3.62E+00	5.72E+01	1.03E+02	1.83E+01	1.18E+02	1.24E+02	6.75E+01	2.06E+01	3.28E+00	3.05E+01
10	Mean	4.18E+02	1.72E+03	4.08E+02	4.59E+02	4.43E+02	4.28E+02	1.19E+03	4.24E+02	4.26E+02	4.97E+02
	SD	2.52E+01	7.14E+02	2.15E+01	5.48E+00	2.64E+01	1.59E+01	2.26E+02	2.06E+01	2.52E+01	9.73E+01

Table 14
Results of IEEE CEC20 test function with $D = 15$ obtained by CoBSA, CPI-DE, DNLPSO, FSO, MFO, SCA, SHO, SOA, SSA, and TSA.

F	Algorithm	CoBSA	CPI-DE	DNLPSO	FSO	MFO	SCA	SHO	SOA	SSA	TSA
1	Mean	5.89E+03	2.11E+10	1.16E+04	1.78E+08	5.65E+03	1.30E+09	1.57E+10	1.62E+09	1.77E+03	5.45E+09
	SD	9.67E+03	2.37E+10	2.11E+04	1.95E+08	5.93E+03	3.32E+09	7.19E+10	5.18E+09	1.96E+03	6.74E+09
2	Mean	1.52E+02	3.77E+03	3.07E+02	1.59E+03	9.81E+02	3.59E+03	3.36E+03	9.13E+02	8.73E+02	1.42E+03
	SD	2.87E+02	1.50E+03	1.97E+02	5.76E+02	2.71E+02	1.29E+03	1.57E+03	1.51E+02	3.19E+02	9.12E+02
3	Mean	8.77E+01	1.25E+03	9.45E+01	6.67E+01	4.95E+01	1.06E+02	2.84E+02	1.02E+02	9.62E+01	1.38E+02
	SD	3.27E+01	1.19E+03	1.95E+01	2.77E+01	2.15E+01	1.02E+02	1.97E+02	1.54E+02	5.12E+01	1.14E+02
4	Mean	3.39E+00	4.39E+06	8.85E-01	8.75E+01	2.39E+02	5.30E+01	5.71E+05	2.51E+00	7.06E+00	2.11E+04
	SD	1.20E+00	3.32E+06	7.25E-01	7.95E+01	1.95E+02	4.00E+01	3.92E+05	1.96E+00	1.05E+00	1.86E+04
5	Mean	3.01E+02	1.73E+08	1.19E+03	8.79E+04	1.72E+04	6.45E+04	8.16E+07	1.06E+04	1.65E+03	2.32E+04
	SD	2.96E+02	1.49E+08	1.98E+03	1.95E+04	1.19E+04	5.12E+04	3.20E+07	1.02E+04	1.59E+03	1.97E+04
6	Mean	3.14E+00	4.81E+02	1.44E+02	1.60E+00	1.07E+02	3.22E+02	7.20E+02	4.22E+02	5.65E-01	7.73E-12
	SD	0.00E+00	1.79E+02	1.39E+02	1.52E+00	1.06E+02	1.98E+02	5.30E+02	2.98E+02	5.98E-01	5.08E-12
7	Mean	3.92E+03	2.96E+07	1.96E+03	3.56E+04	1.69E+03	1.68E+04	1.12E+07	4.23E+04	9.65E+03	8.74E+06
	SD	2.01E+03	1.92E+07	1.59E+02	7.73E+04	1.63E+03	1.20E+04	1.11E+07	1.30E+04	7.91E+02	5.14E+06
8	Mean	1.01E+02	4.22E+03	1.09E+02	2.10E+02	3.66E+02	1.89E+02	3.25E+03	1.42E+03	1.01E+02	2.54E+02
	SD	6.56E-01	1.28E+03	1.01E+02	2.02E+02	3.20E+02	1.74E+02	2.30E+03	1.20E+03	1.01E+02	2.19E+02
9	Mean	1.12E+02	1.35E+03	3.95E+02	4.70E+02	4.38E+02	4.64E+02	1.02E+03	4.32E+02	4.30E+02	5.91E+02
	SD	7.81E+00	1.19E+03	1.50E+02	3.97E+02	4.20E+02	2.70E+02	1.01E+03	1.19E+02	1.30E+02	5.17E+02
10	Mean	4.00E+02	4.34E+03	4.00E+02	5.76E+02	5.69E+02	6.44E+02	1.95E+03	6.23E+02	4.00E+02	7.09E+02
	SD	2.64E-13	4.11E+03	3.80E+02	5.16E+02	1.99E+01	1.74E+02	1.37E+02	2.13E+02	7.90E+02	1.03E+02

and inequality constraints during optimization. A heuristic algorithm's potential solutions can be classified as feasible or infeasible according to the analysis of constraints. Currently, a number of constraint techniques are accessible, such as the death penalty, annealing, static, dynamic, co-evolutionary, and adaptive constraint. This work employed the simple death penalty procedure with little processing cost when dealing with search individuals that exceeded constraints despite the possibility that useful information could be lost during the abandoning process. Moreover, CoBSA was tested on six design problems, including the three-bar truss, welded beam, speed reducer, cantilever beam, I-beam and car side impact design problems. The mathematical formulations of these problems are presented in Appendix B. The performance of CoBSA on these problems is detailed below.

5.1. *Three-bar truss design problem*

In the research area of optimization, the three-bar truss design optimization problem [63] is a well-known structural engineering benchmark NP-hard optimization problem. Fig. 9 depicts the number of different components that make up this problem, in which the volume of a three-bar truss should be kept to a minimum due to stress (σ) constraints on individual truss components. This problem aims to determine the best cross-sectional regions and only involves two design variables ($A_1 = A_3 = x_1 and A_2 = x_2$), representing the cross-sectional areas of the bar members. The details description of this problem can be seen in Ref. [64]..

Table 20 compares the optimum outcomes obtained by CoBSA and other competitive methods reported in the literature, including e-SOSBSA [22], MFO [50], DEDS [65], MBA [66], CS [64], BSA [1], SOS [67], SCA [51], and Ray & Saini algorithm [68]. To obtain the optimum value of this problem, the population size is set to 25, and the number of function evaluations, is set to 15,000 for all the algorithms. The optimal values of the design variables ($x_1 and x_2$), obtained by the suggested CoBSA method are 0.7884761 and 0.4087993, respectively. The obtained outcome of volume of the three-bar truss was determined to be 263.8946488454. This indicates that CoBSA successfully minimizes the volume of the statically loaded truss while satisfying stress constraints on its members. Across various algorithms, CoBSA consistently shows marginal differences in objective function values, indicating its effectiveness. Notably, comparison of CoBSA with e-SOSBSA, MFO, DEDS, MBA, BSA, and SOS, CoBSA demonstrates the improvement of the volume of the three-bar truss, ranging from, approximately, 0.000453% to 0.000504%. Even compared to the algorithms like CS and SCA, the proposed CoBSA remains highly competitive as the improvement of the objective function values are around 0.029% and 0.015%, respectively. The analysis of the results indicates the ability of CoBSA's to provide optimal or near-optimal solutions efficiently, compared to the other state-of-the-art methods. Moreover, the larger improvement of volume of the three-bar truss to Ray & Saini's approach (approximately 0.154%) showcases CoBSA's robustness, indicating its capability to outperform certain methods significantly. Overall, these results strongly favor the proposed CoBSA method as a reliable and efficient optimization technique for the three-bar truss design problem.

5.2. *Welded beam design problem*

The goal of this problem is to reduce the cost of constructing the welded beam, as shown in [69]. The constraints of the optimization problem include shear stress (s), bending stress (r) in the beam, buckling load on the bar (P_c), end deflection of the beam (d), and side constraints. The decision variables of this optimization problem are the thickness of the weld (h), length of the clamped bar (l), height of the bar (t), and thickness of the bar (b). These can be seen in Fig. 10, which are represented as $\vec{x} = (x_1, x_2, x_3, x_4) = (h,\ l,\ t,\ b)$. The mathematical formulation and the details description of the application example is presented in [69].

Table 21 compares the results obtained by the suggested CoBSA with those achieved by e-SOSBSA [22], Ray Optimization (RO) [70], Gravitational Search Algorithm (GSA) [71], Symbiotic Organisms Search (SOS) [67], Colliding Bodies Optimization (CBO) [72], BSA [1], Whale Optimization Algorithm (WOA) [69], Random [73], David [73], Harmony Search (HS) [74], Improved HS [75], APPROX [73], and Simplex [73]. The number of individuals and the number of function evaluations has been set to 20 and 10,000, respectively, for obtaining the optimum solution for this problem.

The optimum values of the thickness of the weld (x_1), length of the clamped bar (x_2), height of the bar (x_3), and thickness of the bar (x_4) were determined by CoBSA to be 0.20583967, 3.4704779, 9.0367147, and 0.2056296, respectively. The optimum cost of welded beam was calculated as 1.724280316192244. The comparative analysis reveals the superiority of the proposed CoBSA method over various optimization algorithms such as e-SOSBSA, RO, GSA, SOS, CBO, BSA, WOA, Random, David, HS, Improved HS, APPROX, and Simplex. Calculating the improvement percentage of the fabrication cost of the welded beam obtained with CoBSA compared with the other considered algorithms indicates its efficacy in minimizing costs. While compared to e-SOSBSA and RO, CoBSA show marginal (0.0332%) and slight (0.6376%) improvements of the cost of the welded beam. CoBSA demonstrates notable

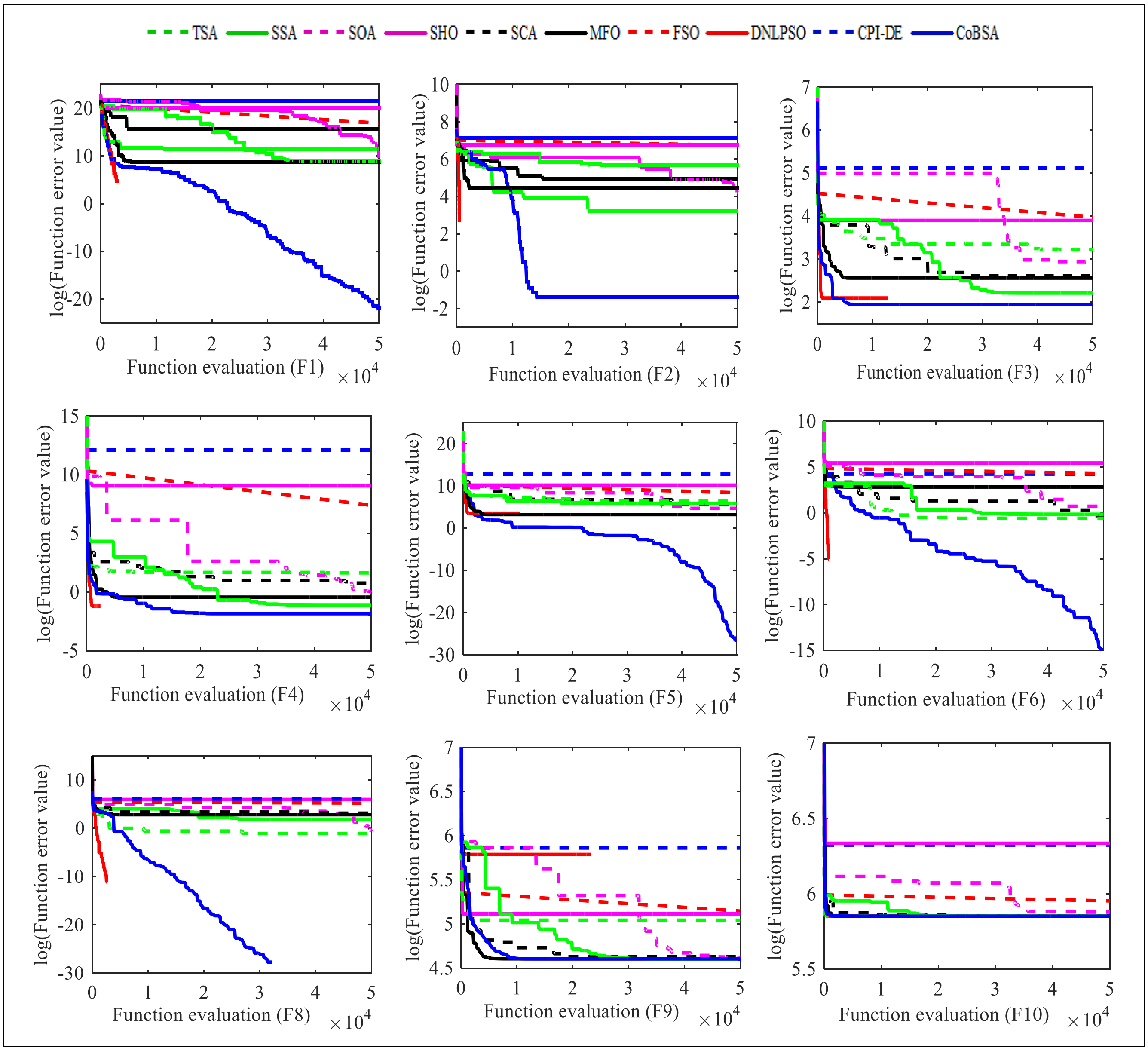

Fig. 8. Convergence curve of CEC20 test function with dimension 5.

Table 15
Result of Kendall's W rank test obtained by mean result of IEEE CEC20 test functions with $D = 5$ for the algorithm CoBSA, CPI-DE, DNLPSO, FSO, MFO, SCA, SHO, SOA, SSA, and TSA.

Algorithm	Mean rank	Final rank
CoBSA	1.56	1
CPI-DE	10	9
DNLPSO	3.44	2
FSO	5.06	5
MFO	4.11	3
SCA	5.61	6
SHO	8.89	8
SOA	4.78	4
SSA	3.44	2
TSA	8.11	7
w	0.793	

Table 16
Result of Kendall's W rank test obtained by mean result of IEEE CEC20 test functions with $D = 10$ for the algorithm CoBSA, CPI-DE, DNLPSO, FSO, MFO, SCA, SHO, SOA, SSA, and TSA.

Algorithm	Mean rank	Final rank
CoBSA	2.50	1
CPI-DE	10.00	10
DNLPSO	2.80	3
FSO	6.45	7
MFO	4.90	5
SCA	5.00	6
SHO	9.00	9
SOA	4.00	4
SSA	2.70	2
TSA	7.65	8
w	0.788	

Table 17
Result of Kendall's W rank test obtained by mean result of IEEE CEC20 test functions with $D = 15$ for the algorithm CoBSA, CPI-DE, DNLPSO, FSO, MFO, SCA, SHO, SOA, SSA, and TSA.

Algorithm	Mean rank	Final rank
CoBSA	2.20	1
CPI-DE	9.90	10
DNLPSO	2.80	2
FSO	5.40	5
MFO	4.20	4
SCA	6.30	7
SHO	9.00	9
SOA	5.60	6
SSA	2.90	3
TSA	6.70	8
w	0.733	

efficiency improvement of the cost of the welded beam over GSA (8.2957%) and substantial superiority over SOS (38.9647%). Additionally, CoBSA exhibits consistent performance with minimal (0.0222% to 2.1225%) improvements of the cost of the welded beam over CBO and BSA. It maintains a competitive edge with slight improvements of the cost of the welded beam (0.3592% to 0.6376%) over WOA and RO. The most significant improvements of the cost of the welded beam are observed over algorithms like Random (58.1608%), David (27.6984%), HS (27.6291%), APPROX (27.7123%), and Simplex (31.8567%), highlighting CoBSA's leap forward in optimization capabilities. In summary, CoBSA outperforms all compared algorithms, consistently showcasing its efficacy and potential as the preferred optimization method, with percentage improvements of the cost of the welded beam uniformly in favour of CoBSA, ranging from marginal to substantial across the evaluated algorithms.

5.3. Speed reducer design problem

The speed reducer design problem [64] contains several variables to be optimized, including the number of teeth on the pinion (z), face width (b), module of teeth (m), diameter of shaft 2 (d_2), length of shaft 1 between bearings (l_1), length of shaft 2 between bearings (l_2), diameter of shaft 1 (d_1), and length of shaft 2 as presented in Fig. 11. The goal of this technical benchmark architecture optimization problem is to reduce the total weight of the speed reducer. Bending stress of the gear teeth, surface stress, transverse deflections of shafts 1 and 2 due to transmitted force, and stresses in shafts 1 and 2 are all restrictions in this problem. Here, the design variables are $\vec{x} = (x_1,x_2,x_3,x_4,x_5,x_6,x_7) = (b,\ m,\ z,\ l_1,\ l_2,\ d_1,\ d_2)$. This problem is analytically expressed in ref [64]..

A set of different algorithms was previously employed to solve this application example, as reported in Ref. [22]. Table 22 summarizes the best solution obtained by the suggested CoBSA method for this problem. To make fare comparisons, the results of all algorithms were obtained using identical parameter settings as presented in Ref. [22]. The population size and number of function evaluations has been set to 50 and 50, 000, respectively, to find the optimum solution for all the algorithms. The obtained outcome of CoBSA was compared with those of e-SOSBSA [22], modified Sine Cosine Algorithm (m-SCA) [76], SCA [51], PSO [47], weighted PSO (wPSO) [77], opposition-based SCA (OBSCA) [76], MFO [50], Grey Wolf Optimizer (GWO) [78], WOA [69], weighted GWO (wGWO) [79], CS [64], Montes and Coello [80], SSA [54], modified GWO (mGWO) [81], Chaotic-SSA [82], and Improved Sine Cosine Algorithm (ISCA) [22]. The value of all decision variables (x_1,x_2,x_3,x_4,x_5, x_6,x_7) obtained by CoBSA were determined to be 3.5000000, 0.7000000, 17.0000000, 7.3000000, 7.7161987, 3.3503597, and 5.2865659, respectively, and the optimum weight of speed reducer was calculated as 2994.471058507. Compared to other algorithms such as e-SOSBSA, m-SCA, SCA, PSO, wPSO, OBSCA, MFO, GWO, WOA, wGWO, CS, Montes and Coello, SSA, mGWO, Chaotic SSA, and ISCA, CoBSA consistently demonstrates superior performance. Calculating the improvement percentage of the total weight of the speed reducer obtained by CoBSA's over each algorithm's result, it is found that CoBSA exhibits marginal to substantial improvements over its competitors. For instance, against e-SOSBSA, the improvement of the total weight of the speed reducer is a negligible (0.0000002536%) and it's indicating the comparable performance. However, CoBSA demonstrates significant superiority over algorithms like MFO and m-SCA, with the improvement of the total weight of the speed reducer of 21.95% and 1.28%, respectively, indicating its effectiveness. Other algorithms such as SCA,

Table 19
Result of Kendall's W rank test obtained by mean result of IEEE CEC20 test functions with $D = 10$ for the algorithm CoBSA, COBSCA, COCRO, NCOGOA, and GMPBSA.

Algorithm	Mean Rank	Final Rank
CoBSA	1.10	1
COBSCA	4.50	5
COCRO	3.10	3
NCOGOA	3.70	4
GMPBSA	2.60	2
w	0.651	

Table 18
Results of IEEE CEC20 test function with $D = 10$ obtained by CoBSA, COBSCA, COCRO, NCOGOA, and GMPBSA.

F	Algorithm	CoBSA	COBSCA	COCRO	NCOGOA	GMPBSA
1	Mean	8.43E+02	1.34E+08	1.98E+03	1.74E+05	1.71E+03
	SD	1.13E+03	3.41E+08	8.03E+03	1.84E+06	3.79E+03
2	Mean	6.24E+02	1.55E+03	9.11E+03	8.76E+02	8.58E+02
	SD	2.41E+02	7.05E+02	9.50E+02	3.30E+02	3.17E+02
3	Mean	2.03E+01	1.92E+02	3.17E+01	5.62E+01	3.73E+01
	SD	1.98E+01	5.30E+01	1.20E+01	1.59E+01	5.19E+00
4	Mean	1.65E+00	2.16E+05	2.10E+01	8.67E+00	1.79E+00
	SD	9.22E-01	1.60E+04	1.04E+01	6.79E+00	1.23E+00
5	Mean	1.70E+03	7.11E+07	1.20E+04	1.41E+05	1.98E+04
	SD	1.47E+03	1.21E+06	2.16E+04	4.11E+04	4.14E+04
6	Mean	3.87E+00	1.70E+02	6.06E+01	7.84E+01	3.98E+00
	SD	7.50E+00	1.21E+02	3.01E+01	1.72E+00	4.17E+00
7	Mean	3.55E+00	2.05E+05	7.39E+01	5.19E+03	2.92E+03
	SD	7.40E+00	1.30E+06	1.20E+02	1.82E+03	1.91E+03
8	Mean	1.22E+01	1.36E+03	1.71E+01	1.57E+02	1.71E+02
	SD	2.00E+01	7.67E+02	1.86E+01	1.76E+01	1.79E+00
9	Mean	3.63E+02	2.42E+02	5.15E+02	7.02E+02	3.97E+02
	SD	3.62E+00	1.02E+01	1.12E+02	1.90E+01	1.79E+02
10	Mean	4.18E+02	1.92E+03	5.98E+02	4.99E+02	4.72E+02
	SD	2.52E+01	2.54E+02	3.18E+01	5.97E+00	2.78E+01

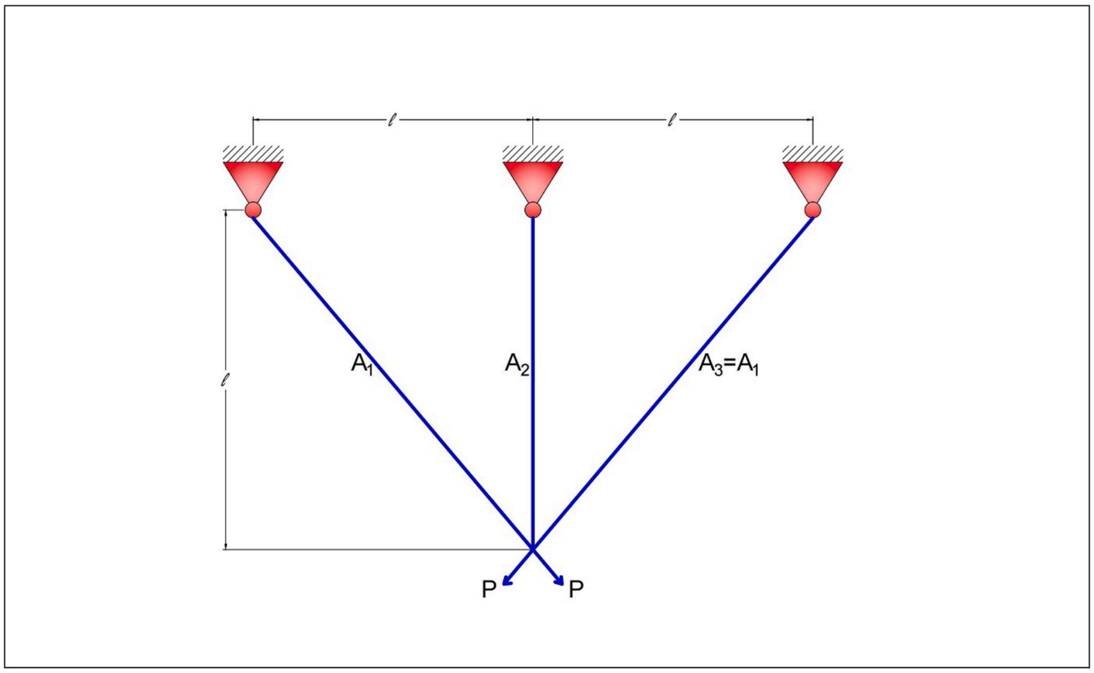

Fig. 9. Three-bar truss design problem ($A_1 = A_3 = x_1 and A_2 = x_2$).

Table 20
Performance comparison of CoBSA with selected algorithms for the three-bar truss design problem.

Algorithm	Decision variables		Objective function value
	x_1	x_2	
CoBSA	**0.7884761**	**0.4087993**	**263.8946488454**
e-SOSBSA [22]	0.7886751	0.4082483	263.8958434
MFO [50]	0.788244770931922	0.788244770931922	263.895979682
DEDS [65]	0.78867513	0.40824828	263.8958434
MBA [66]	0.7885650	0.4085597	263.8958522
CS [64]	0.78867	0.40902	263.9716
BSA [1]	0.7886662	0.4082735	263.8958437
SOS [67]	0.7886548	0.4083059	263.8958441
SCA [51]	0.78669	0.41426	263.9348
Ray & Saini, [68]	0.795	0.395	264.30

OBSCA, and GWO also exhibit notable advantages in favor of CoBSA, with percentage improvement of the total weight of the speed reducer ranging from 1.14% to 0.85%. Moreover, CoBSA outperforms PSO, wPSO, WOA, and CS with improvement of the total weight of the speed reducer ranging from 0.36% to 0.52%, highlighting its consistent superiority across various optimization landscapes. The analysis of the results emphasizes CoBSA's robust and substantial performance advantages across various optimization algorithms, positioning it as a highly effective choice for optimization tasks due to its consistent and superiority achievement over alternative methods.

5.4. Cantilever beam design problem

The geometry of the problem is depicted in Fig. 12. Here, a cantilever beam [22] consists of five hollow square blocks with uniform thickness. The beam is fixed at one end, while a vertical force acts on the free end of the fifth block. The goal of this problem is to minimize the beam's weight while satisfying the constraint of an upper limit on the vertical displacement of the free end. This task involves optimizing the depths or widths of the five blocks, which serve as the decision variables(x_1,x_2,x_3, x_4,andx_5). The problem is expressed analytically in Ref. [64].

The suggested CoBSA has been applied to solve this application example using the same parameter settings as in [22]. Specifically, 50 individuals and 2500 function evaluations have been utilized to obtain the optimal output. Table 23 compares the results of the suggested CoBSA with e-SOSBSA [22], MFO [50], SOS [67], OMFO [83], E-MFO [84], BOA [85], BSA [1], Convex Linearization (CONLIN) [64], CS [64], multimodulus blind equalization algorithm (MMA) [64], SCA [50], Generalized Convex Approximation (GCA(II)) [86], and Generalized Convex Approximation (GCA(I)) [86]. The values of all common control parameters of all algorithms were set identically to those in Ref. [22]. The values of optimum width (or height) (x_1,x_2,x_3,x_4,andx_5) of the beams obtained by CoBSA, which are 6.01605591, 5.3091931, 4.4946927, 3.501525, and 2.1521718, respectively. The value of the optimum weight was calculated as 1.339955043024. The CoBSA method was compared with various optimization algorithms by calculating the percentage of improvements of weight of the beam obtained by CoBSA over each compared algorithm's value. Notable algorithms like e-SOSBSA and SOS show marginal improvements of weight of the beam in favour of CoBSA, ranging from approximately 0.0001% to 0.0004%, reaffirming its comparable performance. However, against algorithms such as MFO, OMFO, E-MFO, and BOA, CoBSA demonstrates substantial improvements, with improvements of weight of the beam approximately 14%, 2.26%, 2.70%, and 2.06% respectively, emphasizing its significant superiority. Other algorithms like BSA, CONLIN, CS, MMA, SCA, GCA (II), and GCA (I) exhibit minimal improvement of weight of the beam in favour of CoBSA, ranging around 0.0007% to 0.0034%, further highlighting its effectiveness. Overall, these analyses emphasize CoBSA's consistent dominance in achieving lower objective function values, solidifying its position as a compelling choice for optimization tasks due to its evident superiority over alternative methods.

5.5. I-beam design problem

CoBSA was also applied to solve the I-beam design problem with four design variables. The goal of this application example is to reduce the vertical deflection of an I-beam, as presented in [87]. Here, the length of the beam (L) and modulus of elasticity (E) are respectively 200 cm and $2 \times 10^4 kN/cm^2$; and $P = 600$ kN. Fig. 13 describes the design of I-beam.

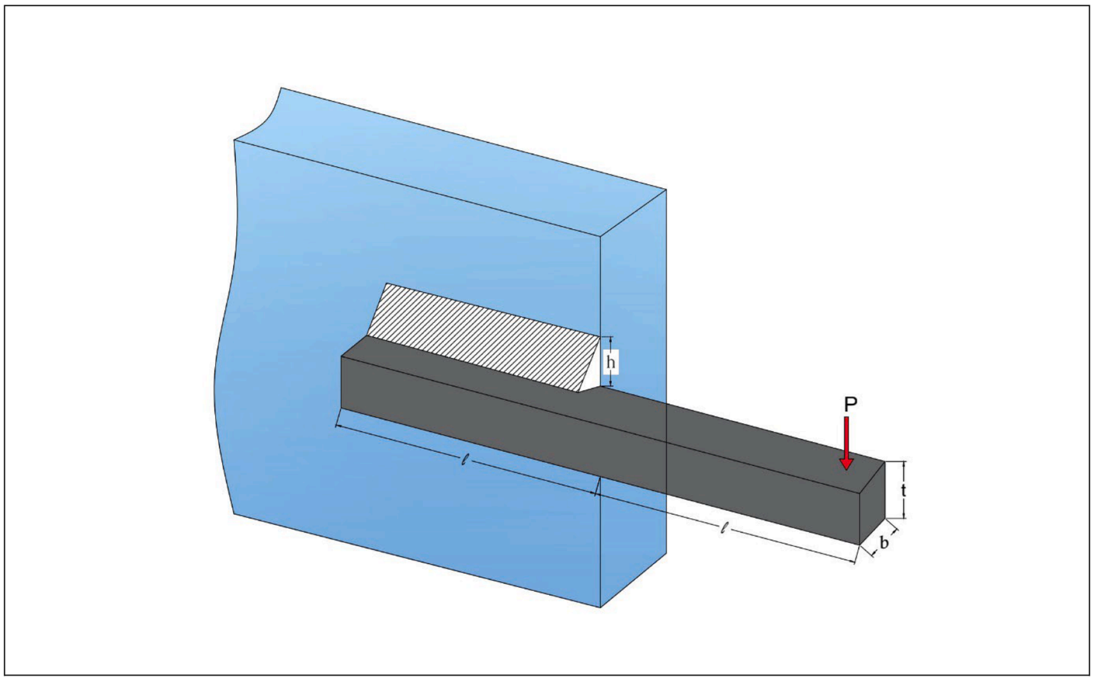

Fig. 10. Welded beam design problem $[\vec{x} = (x_1, x_2, x_3, x_4) = (h,\ l,\ t,\ b)]$.

Table 21
Performance comparison of CoBSA with some well-established algorithms for welded beam design problem.

Algorithm	Optimum variables				Optimum cost
	x_1	x_2	x_3	x_4	
CoBSA	**0.20583967**	**3.4704779**	**9.0367147**	**0.2056296**	**1.724280316192244**
e-SOSBSA [22]	0.2057296	3.4704887	9.0366239	0.2057296	1.7248523
RO [70]	0.203687	3.528467	9.004233	0.207241	1.735344
GSA [71]	0.182129	3.856979	10.00000	0.202376	1.879952
SOS [67]	0.2982538	3.1934693	6.6381460	0.4568884	2.8225627
CBO [72]	0.205722	3.47041	9.037276	0.205735	1.724663
BSA [1]	0.1950195	3.7025696	9.1140183	0.2069155	1.7616706
WOA [69]	0.205396	3.484293	9.037426	0.206276	1.730499
Random [73]	0.4575	4.7313	5.0853	0.6600	4.1185
David [73]	0.2434	6.2552	8.2915	0.2444	2.3841
HS [74]	0.2442	6.2231	8.2915	0.2443	2.3807
Improved HS [75]	0.20573	3.47049	9.03662	0.2057	1.7248
APPROX [73]	0.2444	6.2189	8.2915	0.2444	2.3815
Simplex [73]	0.2792	5.6256	7.7512	0.2796	2.5307

The I-beam design optimization problem contains the cross-sectional area and stress requirements under specified loads. Here, the design variables are $\vec{x} = (x_1, x_2, x_3, x_4)$. The mathematical formulation of the optimization problem is presented in [87]. The results obtained by e-SOSBSA [22], BSA [1], CS [64], GWO [78], Improved Adaptive Response Surface Method (Improved ARSM) [87], Adaptive Response Surface Method (ARSM) [87], SOS [67], and MFO [50] were compared with the proposed CoBSA. 25 individuals and 4570 function evaluations have been utilized to find the optimum solution by proposed CoBSA. The optimum value obtained by CoBSA utilized 4570 function evaluation where as other compared algorithm utilized 5000 function evaluation. Table 24 presents the optimum output of the decision variables x_1,x_2,x_3, andx_4 obtained by CoBSA are 79.9999929, 49.9998265, 0.9000011, and 2.3217993, respectively, and the optimum value of the vertical deflection is 0.0130741. Compared to other algorithms such as e-SOSBSA, BSA, CS, GWO, Improved ARSM, ARSM, SOS, and MFO, CoBSA consistently delivers superior results, producing the lowest objective function value with lower number of evaluations of the objective function. This signifies CoBSA's effectiveness in optimizing the I-beam design to meet objectives while adhering to constraints. e-SOSBSA, SOS and MFO obtain the same optimum objective function value as obtained by CoBSA but CoBSA requires less function evaluation than e-SOSBSA, SOS and MFO. These results confirm that CoBSA converges faster than the other methods on solving this problem. Calculating the percentage improvement of the vertical deflection of the I-beam obtained with CoBSA compared with the other considered algorithms reveals its effectiveness in solving this problem. Notably, CoBSA achieves identical results to e-SOSBSA, SOS, and MFO, indicating its comparable performance. However, CoBSA demonstrates marginal improvements over CS, BSA, and GWO with percentage improvement of the vertical deflection of an I-beam of approximately 0.0046%, 0.96%, and 0.98%, respectively. Most significantly, CoBSA outperforms ARSM by achieving substantially lower results, with a percentage improvement of the vertical deflection of an I-beam of approximately 20.08%, showcasing its superiority. Similarly, CoBSA shows a notable improvement over Improved ARSM with a percentage improvement of the vertical deflection of an I-beam of approximately 0.198%. The above analysis of the results highlights CoBSA's ability to achieve competitive or superior results compared to most of the other algorithms, making it an excellent choice for optimization problem.

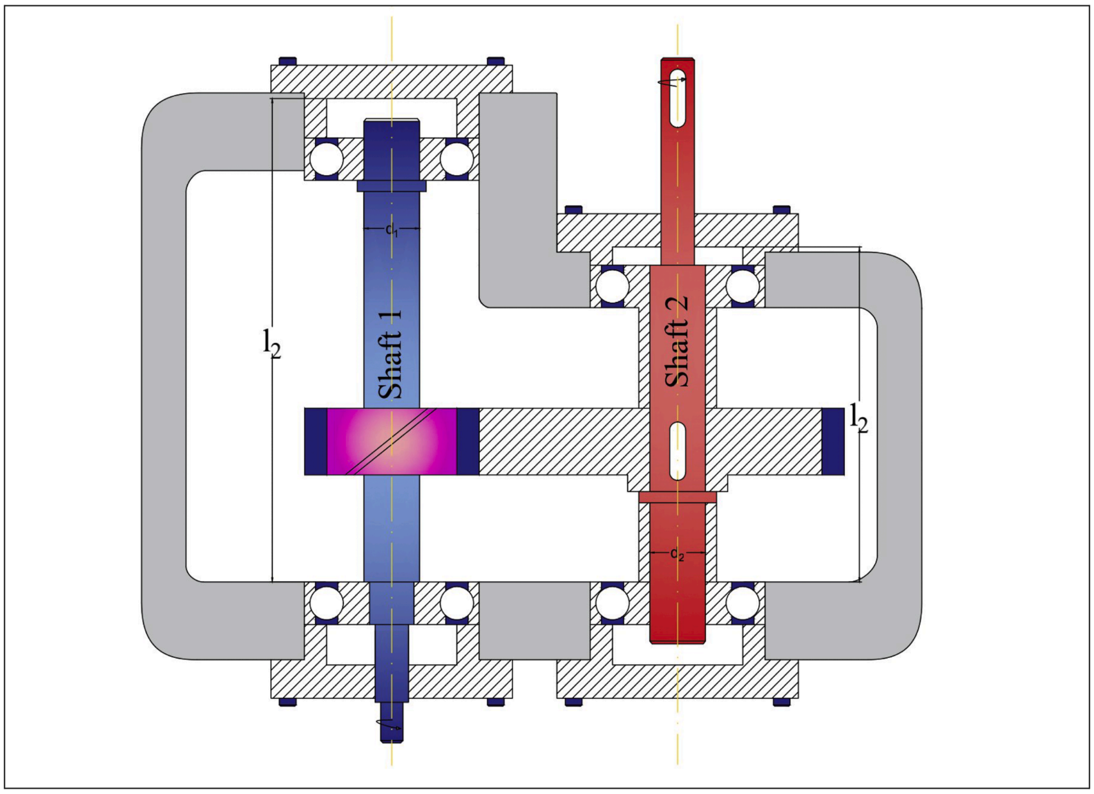

Fig. 11. Graphical view of speed reducer design problem [$\vec{x} = (x_1, x_2, x_3, x_4, x_5, x_6, x_7) = (b,\ m,\ z,\ l_1, l_2,\ d_1,\ d_2)$.].

Table 22
Performance comparison of CoBSA against various state-of-the-art algorithms for speed reducer problem.

Algorithm	x_1	x_2	x_3	x_4	x_5	x_6	x_7	f_{min}
CoBSA	**3.5000000**	**0.7000000**	**17.0000000**	**7.3000000**	**7.7161987**	**3.3503597**	**5.2865659**	**2994.471058507**
e-SOSBSA [22]	3.5000000	0.7000000	17.0000000	7.3000000	7.7153199	3.3502147	5.2866545	2994.4710661
m-SCA [76]	3.52394	0.7	17	7.3	7.8	3.36280	5.32467	3033.2845
SCA [51]	3.51889	0.7	17	7.3	8.3	3.35899	5.30519	3028.8657
PSO [47]	3.58147	0.7	17.8282	7.98445	7.82083	3.15398	5.1873	3005.3248
wPSO [77]	3.50662	0.7	17	7.44735	7.88468	3.2725	5.38998	3003.7983
OBSCA [76]	3.00576	0.72755	21.8423	7.30835	8.15455	3.36452	5.25164	3027.5130
MFO [50]	3.59093	0.70554	19.7972	8.08267	7.84181	3.70621	5.48167	3836.2164
GWO [78]	3.6	0.8	28	7.3	8.3	2.9	5.0	3020.2331
WOA [69]	3.52111	0.7	17	7.3	7.8	3.35021	5.29533	3010.1480
wGWO [79]	3.50008	0.7	17	7.3193	7.81168	3.35072	5.28692	2997.085
CS [64]	3.50150	0.7	17	7.6050	7.8181	3.3520	5.2875	3000.9810
Montes and Coello [80]	3.50616	0.700831	17	7.46018	7.962143	3.3629	5.3090	3025.005
SSA [54]	3.50031	0.7	17	7.80001	7.85001	3.35247	5.2867	3002.5678
mGWO [81]	3.50128	0.7	17	7.34965	7.80177	3.35087	5.28712	2997.7748
Chaotic SSA [82]	3.50031	0.7	17	7.80001	7.85001	3.35247	5.2867	3002.5678
ISCA [22]	3.50081	0.7	17	7.3	7.8	3.35129	5.28698	2997.1295

5.6. *Car side impact design problem*

A car side-impact design is considered a practical example of a mechanical optimization problem with mixing discrete and continuous design variables. The goal of this problem is to reduce the weight of the car while maintaining or improving dummy safety performance, which is measured by the European Enhanced Vehicle-Safety Committee (EEVC) side impact safety rating score. The problem is a minimization of the total weight of a car using eleven design variables and subject to ten inequality constraints. The design variables of this problem are thicknesses and materials of critical parts of the car, which includes thickness of B-pillar inner (x_1), B-pillar reinforcement (x_2), floor side inner (x_3), cross members(x_4), door beam (x_5), door belt line reinforcement (x_6), roof rail (x_7), material of B-pillar inner (x_8) and floor side inner (x_9), and Barrier height (x_{10}) and Barrier hitting position (x_{11}). Fig. 14 presents the design of the car side impact. The mathematical model of this application example is presented in Ref. [88,89]. The design variables of this problem are $\vec{x} = \{x_1,\ x_2, x_3,\ x_4,\ x_5,\ x_6,\ x_7, x_8, x_9,\ x_{10}, x_{11}\}$.

The car side impact design problem has been solved by many other methods, such as ABC [90], PSO [47], MFO [50], Ant Lion Optimizer (ALO) [91], Evaporation Rate Based WCA (ER-WCA) [92], GWO [78], WCA [93], Mine Blast Algorithm (MBA) [66], SSA [51], and WOA [69], in the literature [89]. To find the optimal solution, 20 individuals and 20,000 function evaluations have been utilized for this problem. Table 25 presents the comparison results of CoBSA with the previously applied techniques in this problem. It can be seen that the proposed CoBSA obtained an optimum value of 19.6504496. Compared to other optimization algorithms such as MFO, ABC, PSO, ALO, ER-WCA, GWO,

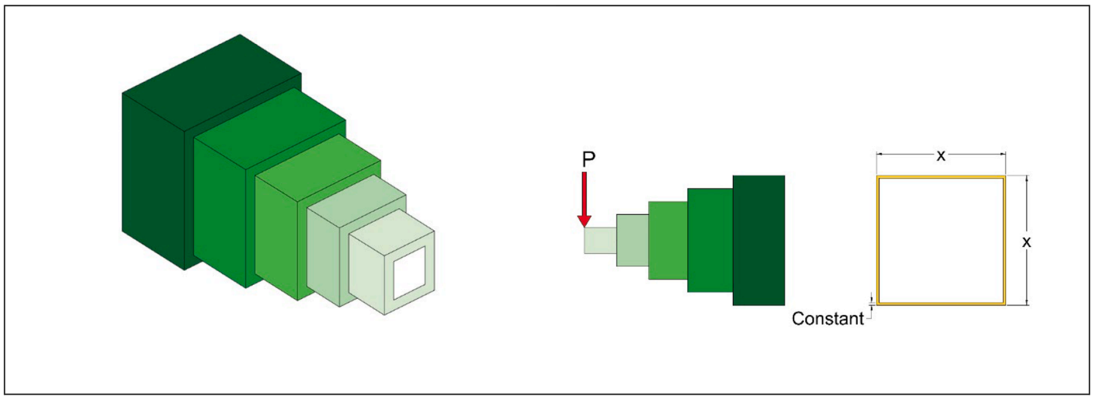

Fig. 12. Cantilever beam structure.

Table 23
Performance comparison of CoBSA with various algorithms for the cantilever beam design problem.

Algorithm	Optimum decision variables					Objective function value
	x_1	x_2	x_3	x_4	x_5	
CoBSA	**6.01605591**	**5.3091931**	**4.4946927**	**3.501525**	**2.1521718**	**1.339955043024**
e-SOSBSA [22]	6.0160377	5.3091474	4.4943187	3.5014931	2.1526627	1.3399564
BSA [1]	6.01605697	5.3093745	4.4947517	3.5016815	2.1519321	1.33996491
CONLIN [64]	6.0100	5.3000	4.4900	3.4900	2.1500	1.3400
CS [64]	6.0089	5.3049	4.5023	3.5077	2.1504	1.33999
MMA [64]	6.0100	5.3000	4.4900	3.4900	2.1500	1.3400
SCA [50]	6.0100	5.3000	4.4900	3.4900	2.1500	1.3400
GCA (II) [86]	6.0100	5.3000	4.4900	3.4900	2.1500	1.3400
GCA (I) [86]	6.0100	5.3000	4.4900	3.4900	2.1500	1.3400
MFO [50]	6.0783	4.0783	4.8385	5.7777	2.4201	1.5275
SOS [67]	6.01878	5.30344	4.49587	3.49896	2.15564	1.33996
OMFO [83]	5.3791	6.9067	3.6337	4.9205	2.2560	1.37036
E-MFO [84]	5.5091	4.8861	4.2490	4.6868	3.0253	1.3761
BOA [85]	5.2982	5.6551	4.3305	3.6671	2.8271	1.3676

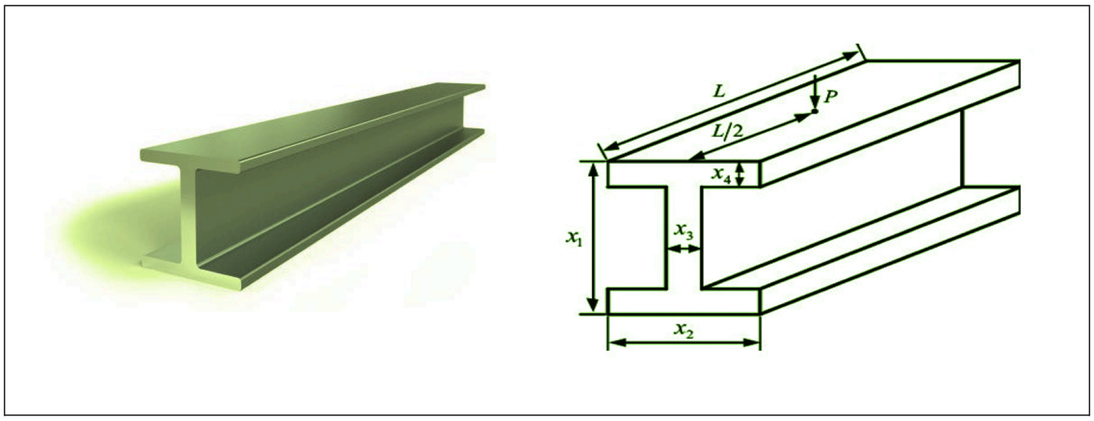

Fig. 13. Diagram of I-beam design problem.

WCA, MBA, SSA, and WOA, CoBSA consistently achieves one of the lowest optimum values, indicating its effectiveness in optimizing the car-side crash problem. In comparison to various optimization algorithms, the CoBSA method consistently achieves lower objective function values, as indicated by the improvement percentage of the total weight of the car. Notably, CoBSA demonstrates significant improvements over each algorithm, with improvement percentage of the total weight of the car. Specifically, CoBSA outperforms MFO, PSO, ALO, ER-WCA, and WCA in comparison to the improvement percentage of the total weight of the car by approximately 16.24%, showcasing its substantial superiority. Moreover, CoBSA exhibits comparable improvements over SSA, WOA and ABC, with improvement percentage of the total weight of the car of around 17.26% to 17.94%. Additionally, GWO and MBA display slightly smaller improvement percentage of the total weight of the car of approximately 16.29% and 16.27%, respectively, favouring CoBSA. In summary, Table 25 highlights CoBSA as a robust optimization technique for solving the car-side crash problem, showcasing its capability to generate solutions with superior performance metrics compared to other algorithms considered in the comparison.

Table 24
Performance comparison of CoBSA with various well-known algorithms for I-beam design problem.

Algorithm	Optimum decision variables x_1	x_2	x_3	x_4	Objective function value f_{min}	No. of function evaluation
CoBSA	**79.9999929**	**49.9998265**	**0.9000011**	**2.3217993**	**0.0130741**	**4570**
e-SOSBSA [22]	79.9999665	49.9995406	0.9000013	2.3218126	**0.0130741**	5000
BSA [1]	79.9457542	47.2805539	0.9057700	2.4417997	0.0132002	5000
CS [64]	80	50	0.9	2.3216715	0.0130747	5000
GWO [78]	80	42.8154	0.9	2.7179	0.013202	5000
Improved ARSM [87]	79.99	48.42	0.9	2.40	0.01310	NA
ARSM [87]	80	37.05	1.71	2.31	0.01570	NA
SOS [67]	80	50	0.90	2.3217920	**0.0130741**	5000
MFO [50]	80	50	0.90	2.3217923	**0.0130741**	5000

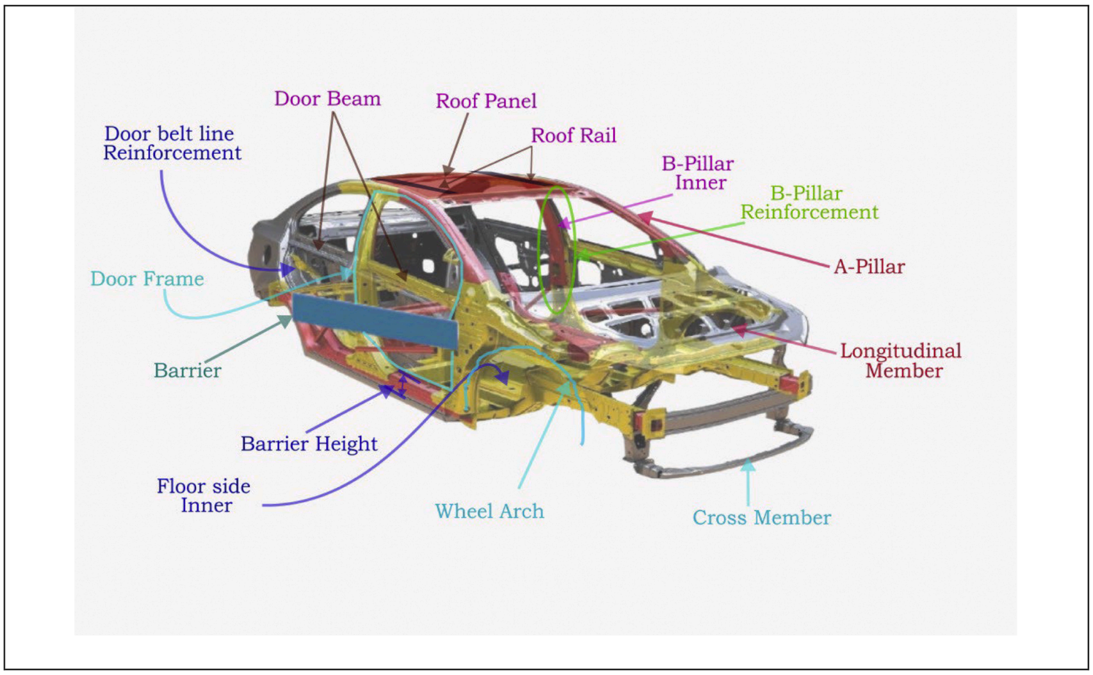

Fig 14. Car side impact design.

Table 25
Performance comparison of CoBSA with several state-of-the-art algorithms for car-side impact design problem.

Variables	CoBSA	MFO [50]	ABC [90]	PSO [47]	ALO [91]	ER-WCA [92]	GWO [78]	WCA [93]	MBA [66]	SSA [51]	WOA [69]
x_1	**0.500000**	0.500000	0.5000000	0.5000000	0.5000000	0.5000000	0.5000000	0.5000000	0.5000000	0.5000000	0.5000000
x_2	**1.107895**	1.116539	1.0624205	1.1165737	1.115960	1.118688	1.111484	1.1155932	1.1172701	1.1093195	1.108001
x_3	**0.510000**	0.500000	0.5148211	0.5000000	0.500000	0.500000	0.500000	0.5000000	0.5000030	0.5000000	0.534477
x_4	**0.50100**	1.301908	1.4491503	1.3018547	1.302860	1.298407	1.312203	1.3034919	1.3008438	1.3148010	1.305770
x_5	**0.50000**	0.500000	0.5000000	0.5000000	0.5000000	0.5000000	0.501214	0.5000146	0.5000000	0.5000000	0.500000
x_6	**1.50000**	1.500000	1.500000	1.5000000	1.5000000	1.500000	1.500000	1.5000000	1.4999867	1.4999998	1.473844
x_7	**0.50000**	0.500000	0.5000000	0.5000000	0.500000	0.500000	0.500034	0.5000000	0.5000000	0.5000000	0.500000
x_8	**0.34500**	0.345000	0.3450000	0.3450000	0.345000	0.345000	0.345000	0.3450000	0.3450000	0.3450000	0.345000
x_9	**0.19200**	0.345000	0.1920000	0.3450000	0.192000	0.192000	0.192000	0.1920000	0.3450000	0.1920000	0.192000
x_{10}	**−30.00000**	- 19.5304	- 29.34755	- 19.52470	- 19.6330	- 19.1461	- 20.6057	- 19.69967	- 19.40045	- 20.821793	- 19.69924
x_{11}	**−0.00000**	- 0.000006	0.7410998	- 0.019297	0.023649	- 0.01527	- 0.25531	- 0.023854	- 0.379205	0.4412962	3.4816923
f_{min}	**19.6504496**	22.84297	23.175889	22.842984	22.84298	22.84298	22.85279	22.843596	22.846514	23.042162	23.042162

6. Conclusion and recommendations for future study

The following improvements have been investigated in this work to increase the performance of BSA.

- The centroid opposition-based initialization incorporated into the BSA allows the population's positions to contract specifically, assuring early exploration efficiency and subsequent exploitation precision.
- Since multiple learning methods based on centroid opposition-based jumping and chaotic mutation are incorporated into the suggested CoBSA method, it can maintain a good population diversity while ensuring quick convergence, attempting to avoid the most efficient local trapping during rapid convergence.

- The adequate utilization of the mutation operator allows CoBSA to make superior resolutions constructed on historical information.
- The implementation of a chaotic-based elite mechanism can ensure CoBSA's flexibility in diverse examination stages.

The overall findings from the suggested CoBSA method analysis are as follows.

- In terms of quantitative analysis, i.e., population size sensitivity, dimension scalability, average fitness, and convergence analysis on test functions, CoBSA demonstrated superior performance to the other compared algorithms.
- Kendall's W ranking statistical test was performed to measure the algorithm's efficacy in comparison to other methods. The investigational outcomes show that CoBSA can ensure the algorithm's better statistical performance compared to other algorithms.
- In addition, CoBSA was utilized to solve five classic structural engineering challenges, which reveals that CoBSA may be applied to solve real-world engineering optimization tasks to produce satisfactory outcomes.

Future studies should investigate the use of CoBSA to solve constrained and multi-objective optimization problems in various domains. Researchers can further test CoBSA's utility in tackling real-world problems, such as optimal control problems, energy scheduling and management, binary feature selection, complex renewable energy problems, image segmentation, and large-scale optimization problems.

Declarations

Funding

No funding has been received during the course of this study from any funding agency.

Data availability statement

All the data used in this study were obtained under the same investigational environment. All algorithms used in this study for comparison were coded according to their original references. The numerous benchmark functions considered in this study are cited with the respective references. The authors solemnly declare that this paper's information is accurate and reliable.

CRediT authorship contribution statement

Sanjib Debnath: Writing – original draft, Methodology, Investigation, Conceptualization. **Swapan Debbarma:** Writing – original draft, Investigation, Conceptualization, Supervision. **Sukanta Nama:** Writing – review & editing, Supervision, Investigation, Conceptualization. **Apu Kumar Saha:** Writing – review & editing, Supervision, Methodology, Conceptualization. **Runu Dhar:** Writing – review & editing, Methodology, Investigation. **Ali Riza Yildiz:** Writing – review & editing, Methodology, Formal analysis. **Amir H. Gandomi:** Writing – review & editing, Supervision.

Declaration of competing interest

The authors state that they have no known competing financial interests or personal ties that could have influenced the research presented in this study.

Data availability

No data was used for the research described in the article.

Appendix A

Twenty well-known benchmark functions were applied to validate the proposed method ($F_{\min} = 0$, S=Search space)

Functions	Formulation	S
F1. Sphere	$f(x) = \sum_{i=1}^{D} x_i^2$	[−100, 100]
F2. Schwefel2.22	$f(x) = \sum_{i=1}^{D} \lvert x_i \rvert + \prod_{i=1}^{D} \lvert x_i \rvert$	[−10, 10]
F3. Schwefel1.2	$f(x) = \sum_{i=1}^{D} \sum_{j=1}^{i} x_i^2$	[−100, 100]
F4. Schwefel2.21	$f(x) = \max\{\lvert x_i \rvert, 1 \le i \le D\}$	[−100, 100]
F5. Rosenbrock	$f(x) = \sum_{i=1}^{D} \left[100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2\right]$	[−30, 30]
F6. Step	$f(x) = \sum_{i=1}^{D} (x_i + 0.5)^2$	[−100, 100]
F7. Quartic	$f(x) = \sum_{i=1}^{D} i x_i^4 + random(0,1)$	[−1.28, 1.28]
F8. Schwefel	$f(x) = 418 : 9829\text{n} - \sum x_i \sin\left(\sqrt{\lvert x_i \rvert}\right)$	[−500, 500]
F9. Rastrigin	$f(x) = 10D + \sum_{i=1}^{D} \left[x_i^2 - 10\cos(2\pi x_i)\right]$	[−5.12, 5.12]
F10.Ackley	$f(x) = 20 + e - 20e^{\frac{1}{D}\left(\sqrt{\frac{1}{D}\sum_{i=1}^{D} x_i^2}\right)} - e^{\frac{1}{D}\left(\sum \cos(2\pi x_i)\right)}$	[−32, 32]
F11.Griewank	$f(x) = \sum_{i=1}^{D} \frac{x_i^2}{4000} - \prod_{i=1}^{D} \cos\left(\frac{x_i}{\sqrt{i}}\right) - 1$	[−600, 600]
F12.Penalized1	$f(x) = \frac{\pi}{D}\left\{10\sin^2(\pi y_i) + \sum_{i=1}^{D-1} (y_i - 1)^2\left[1 + 10\sin^2(3\pi y_{i+1}) + (y_D - 1)^2\right]\right\} + \sum_{i=1}^{D} u(x_i, 10, 100, 4)$	[−50, 50]
	Where $y_i = 1 + \frac{1}{4}(x_i + 1)$, $u(x_i, a, k, m) = \{ \begin{matrix} k(x_i - a)^m & x_i > a \\ 0 & -a < x_i < a \\ k(-x_i - a)^m & x_i < a \end{matrix}$	
F13. Penalized2	$f(x) = 0.1\left\{10\sin^2(\pi x_i) + \sum_{i=1}^{D-1} (x_i - 1)^2\left[1 + 10\sin^2(3\pi x_{i+1}) + (x_D - 1)^2\left[1 + \sin^2(2\pi x_D)\right]\right]\right\} + \sum_{i=1}^{D} u(x_i, 5, 100, 4)$	[−50, 50]
F14. Salomon	$f(x) = 1 - \cos\left(2\pi\sqrt{\sum_{i=1}^{D} x_i^2}\right) + 0.1\sqrt{\sum_{i=1}^{D} x_i^2}$	[−100, 100]

(*continued on next page*)

(*continued*)

Functions	Formulation	S
F15. Zakharov	$f(x) = \sum_{i=1}^{D} x_i^2 + \left(\sum_{i=1}^{D} \frac{ix_i}{2}\right)^2 + \left(\sum_{i=1}^{D} \frac{ix_i}{2}\right)^2$	[−5.12, 5.12]
F16. Axis parallel hyper ellipsoid	$f(x) = \sum_{i=1}^{D} ix_i^2$	[−5.12, 5.12]
F17. Ellipsoidal	$f(x) = \sum_{i=1}^{D} (x_i - i)^2$	[−100, 100]
F18. Cigar	$f(x) = x_1^2 + 100000 \sum_{i=2}^{D} x_i^2$	[−10, 10]
F19. Exponential	$f(x) = 1 - \exp\left(-0.5 \sum_{i=1}^{D} x_i^2\right)$	[−1, 1]
F20. Cosine mixture	$f(x) = 00.1D + \sum_{i=1}^{D} x_i^2 - 0.1 \sum_{i=1}^{D} \cos(5\pi x_i)$	[−1, 1]

Appendix B

B1. Three-bar truss design problem [64]

$$\textbf{Minimize } f_1(\vec{x}) = \left(2\sqrt{2}x_1 + x_2\right) \times l,$$

Subject to

$$g_1(\vec{x}) = \frac{\sqrt{2}x_1 + x_2}{\sqrt{2}x_1^2 + 2x_1x_2} P - \sigma \le 0$$

$$g_2(\vec{x}) = \frac{x_2}{\sqrt{2}x_1^2 + 2x_1x_2} P - \sigma \le 0$$

$$g_3(\vec{x}) = \frac{1}{\sqrt{2}x_2 + x_1} P - \sigma \le 0$$

where $0 \le x_1 \le 1,\ 0 \le x_2 \le 1;\ l = 100cm,\ P = 2\frac{kN}{cm^2},\ \sigma = 2\frac{kN}{cm^2},\ \vec{x} = (x_1, x_2)$

B2. Welded beam design problem [69]

$$\textbf{Minimize } f_2(\vec{x}) = 1.10471x_1^2x_2 + 0.04811x_3x_4(14.0 + x_2),$$

Subject to

$$g_1(\vec{x}) = \tau(\vec{x}) - \tau_{max} \le 0$$

$$g_2(\vec{x}) = \sigma(\vec{x}) - \sigma_{max} \le 0$$

$$g_3(\vec{x}) = x_1 - x_4 \le 0$$

$$g_4(\vec{x}) = 0.10471x_1^2 + 0.04811x_3x_4(14.0 + x_2) - 5.0 \le 0$$

$$g_5(\vec{x}) = 0.125 - x_1 \le 0$$

$$g_6(\vec{x}) = \delta(\vec{x}) - \delta_{max} \le 0$$

$$g_7(\vec{x}) = P - P_c(\vec{x}) \le 0$$

The following equations are the definitions of other parameters:

$$\tau(\vec{x}) = \sqrt{(\tau')^2 + (\tau'')^2 + \frac{2\tau'\tau''x_2}{2R}},\ \tau' = \frac{p}{\sqrt{2}x_1x_2},\ \tau'' = \frac{MR}{J},\ M = P\left(L + \frac{x_2}{2}\right),$$

$$R = \sqrt{\left(\frac{x_1 + x_3}{2}\right)^2 + \frac{x_2^2}{4}},\ J = 2\left\{\frac{x_1x_2}{\sqrt{2}}\left[\frac{x_2^2}{12} + \left(\frac{x_1 + x_3}{2}\right)^2\right]\right\},\ \sigma(\vec{x}) = \frac{6PL}{x_4x_3^2},\ \delta(\vec{x}) = \frac{4PL^3}{Ex_4x_3^3},$$

$$P_c(\vec{x}) = \frac{4.013\sqrt{\frac{EGx_3^2x_4^6}{36}}}{L^2}\left(1 - \frac{x_3}{2L}\sqrt{\frac{E}{4G}}\right)$$

where, $P = 6000$ lb, $L = 14$, $\delta_{max} = 0.25in.$, $E = 30{,}106$ psi, $G = 122{,}106$ psi, $\tau_{max} = 13600psi$, $\sigma_{max} = 30000psi$ and $0.1 \le x_i \le 10.0 (i = 1,2,3,4),\ \vec{x} =$

$(x_1,x_2,x_3,x_4) = (h,\ l,\ t,\ b)$

B3. Speed reducer design problem [64,89]

$$\textbf{\textit{Minimize}} f_3(\vec{x}) = 0.7854x_1x_2^2\left(3.3333x_3^2 + 14.933x_3 - 43.0934\right) - 1.508x_1\left(x_6^2 + x_7^2\right)$$

$$+7.4777\left(x_6^3 + x_7^3\right) + 0.7854\left(x_4x_6^2 + x_5x_7^2\right),$$

Subject to

$$g_1(\vec{x}) = \frac{27}{x_1x_2^2x_3} - 1 \le 0$$

$$g_2(\vec{x}) = \frac{397.5}{x_1x_2^2x_3^2} - 1 \le 0$$

$$g_3(\vec{x}) = \frac{1.93x_4^3}{x_2x_6^4x_3} - 1 \le 0$$

$$g_4(\vec{x}) = \frac{1.93x_5^3}{x_2x_7^4x_3} - 1 \le 0$$

$$g_5(\vec{x}) = \frac{\left[\left(\frac{745x_4}{x_2x_3}\right)^2 + 16.9\ \times 10^6\right]^{\frac{1}{2}}}{110.x_6^3} - 1 \le 0$$

$$g_6(\vec{x}) = \frac{\left[\left(\frac{745x_5}{x_2x_3}\right)^2 + 157.5\ \times 10^6\right]^{\frac{1}{2}}}{85.0x_7^3} - 1 \le 0$$

$$g_7(\vec{x}) = \frac{x_2x_3}{40} - 1 \le 0$$

$$g_8(\vec{x}) = \frac{5x_2}{x_1} - 1 \le 0$$

$$g_9(\vec{x}) = \frac{x_1}{12x_2} - 1 \le 0$$

$$g_{10}(\vec{x}) = \frac{1.5x_6 + 1.9}{x_4} - 1 \le 0$$

$$g_{11}(\vec{x}) = \frac{1.1x_7 + 1.9}{x_5} - 1 \le 0$$

where $2.6 \le x_1 \le 3.6$, $0.7 \le x_2 \le 0.8$, $17 \le x_3 \le 28$, $7.3 \le x_4 \le 8.3$, $7.3 \le x_5 \le 8.3$, $2.9 \le x_6 \le 3.9$, and $5.0 \le x_7 \le 5.5$, $\vec{x} = (x_1,x_2,x_3,x_4,x_5,x_6,x_7) = (b,\ m,\ z,\ l_1,l_2,\ d_1,\ d_2)$

B4. Cantilever beam design problem [22,64]

$$\textbf{\textit{Minimize}}\ f_4(\vec{x}) = 0.0624(x_1 + x_2 + x_3 + x_4 + x_5),$$

Subject to

$$g_1(\vec{x}) = \frac{61}{x_1^3} + \frac{37}{x_2^3} + \frac{19}{x_3^3} + \frac{7}{x_4^3} + \frac{1}{x_5^3} - 1 \le 0$$

where $0.01 \le x_j \le 100\ (j = 1,2,3,4,5),\ \vec{x} = (x_1,x_2,x_3,x_4,x_5)$

B5. I-beam design problem [87]

$$\textbf{\textit{Minimize}}\ f_5(\vec{x}) = \frac{5000}{\frac{x_3(x_1-2x_4)^3}{12} + \frac{x_2x_4^3}{6} + 2x_2x_4\left(\frac{x_1-x_4}{2}\right)^2}$$

Subject to

$$g_1(\vec{x}) = 2x_2x_4 + x_3(x_1 - 2x_4) - 300 \le 0$$

$$g_2(\vec{x}) = \frac{18x_1 \times 10^4}{x_3(x_1-2x_4)^3 + 2x_2x_4\left(4x_4^2 + 3x_1(x_1-2x_4)\right)} + \frac{15x_2 \times 10^3}{(x_1-2x_4)x_3^3 + 2x_4{x_2}^3} - 6 \le 0$$

where $10 \le x_1 \le 80,\ 10 \le x_2 \le 50, 0.9 \le x_3 \le 5,\ and\ 0.9 \le x_4 \le 5, \vec{x} = (x_1, x_2, x_3, x_4) = \left(h, b, t_w, t_f\right)$; ($P = 5600$ kN and $Q = 550$ kN);

B6. Car side impact design problem [88,89]

Objective function:

$$Min\ f_6(\vec{x}) = 1.98 + 4.90\ x_1 + 6.67\ x_2 + 6.98\ x_3 + 4.01\ x_4 + 1.78\ x_5 + 2.73\ x_7,$$

Subject to:

$$h_1(x) = 1.16 - 0.3717\ x_2\ x_4 - 0.00931\ x_2\ x_{10} - 0.484\ x_3\ x_9 +\ 0.01343\ x_6\ x_{10}\ \le 1,$$

$$\begin{aligned} h_2(x) = 0.261 &- 0.0159\ x_1\ x_2 - 0.188\ x_1\ x_8 - 0.019\ x_2\ x_7\ + 0.0144\ x_3\ x_5 \\ &+ 0.0008757\ x_5\ x_{10} + 0.080405\ x_6\ x_9 + 0.00139\ x_8\ x_{11} \\ &+ 0.00001575\ x_{10}\ x_{11} \le 0.32, \end{aligned}$$

$$\begin{aligned} h_3(x) = 0.214 &+ 0.00817\ x_5 - 0.131\ x_1x_8 - 0.0704\ x_1x_9 + 0.03099\ x_2\ x_6 - 0.018\ x_2\ x_7 \\ &+0.0208\ x_3\ x_8 + 0.121\ x_3\ x_9\ - 0.00364\ x_5\ x_6 +\ 0.0007715\ x_5\ x_{10} \\ &-0.0005354\ x_6\ x_{10}\ + 0.00121\ x_8\ x_{11} \le 0.32, \end{aligned}$$

$$h_4(x) = 0.074 - 0.061\ x_2 - 0.163\ x_3\ x_8 + 0.001232\ x_3\ x_{10} - 0.166\ x_7\ x_9\ + 0.227\ x_2^2\ \le 0.32,$$

$$h_5(x) = 28.98 + 3.818\ x_3 - 4.2\ x_1\ x_2 + 0.0207\ x_5\ x_{10} + 6.63\ x_6\ x_9 - 7.7\ x_7\ x_8 + 0.32\ x_9\ x_{10} \le 32,$$

$$h_6(x) = 33.86 + 2.95\ x_3 + 0.1792\ x_{10} - 5.05\ x_1\ x_2 - 11.0\ x_2\ x_8\ -\ 0.0215\ x_5\ x_{10} - 9.98\ x_7\ x_8 + 22.0\ x_8\ x_9 \le 32,$$

$$h_7(x) = 46.36 - 9.9\ x_2 - 12.9\ x_1\ x_8 + 0.1107\ x_3\ x_{10} \le 32,$$

$$h_8(x) = 4.72 - 0.5\ x_4 - 0.19\ x_2\ x_3 - 0.0122\ x_4\ x_{10} + 0.009325\ x_6\ x_{10} +$$

$$0.000191\ x_{11}^2\ \le 4,$$

$$h_9(x) = 10.58 - 0.674\ x_1\ x_2 - 1.95\ x_2\ x_8 + 0.02054\ x_3\ x_{10} -\ 0.0198\ x_4\ x_{10} + 0.028\ x(6)\ x(10) \le 9.9,$$

$$h_{10}(x) = 16.45 - 0.489\ x_3\ x_7 - 0.843\ x_5\ x_6 + 0.0432\ x_9\ x_{10} - 0.0556\ x_9\ x_{11}\ - 0.000786\ x_{11}^2\ \le 15.7,$$

where $0.5 \le x_i \le 1.5,\ i = 1,\ 2,\ 3,\ 4,\ 5,\ 6,\ 7x_8{,}x_9 \in (0.192,\ 0.345),\ -30 \le x_{10}{,}x_{11} \le 30.$

References

[1] Civicioglu P. Backtracking search optimization algorithm for numerical optimization problems. Appl Math Comput 2013;219:8121–44. https://doi.org/10.1016/j.amc.2013.02.017.

[2] Madasu SD, Kumar MLSS, Singh AK. Comparable investigation of backtracking search algorithm in automatic generation control for two area reheat interconnected thermal power system. Appl Soft Comput J 2017;55:197–210. https://doi.org/10.1016/j.asoc.2017.01.018.

[3] Nama S, Saha AK, Ghosh S. Improved backtracking search algorithm for pseudo dynamic active earth pressure on retaining wall supporting c-Φ backfill. Appl Soft Comput J 2017;52:885–97. https://doi.org/10.1016/j.asoc.2016.09.037.

[4] Nama S, Saha AK. A new hybrid differential evolution algorithm with self-adaptation for function optimization. Appl Intell 2018;48:1657–71. https://doi.org/10.1007/s10489-017-1016-y.

[5] Mehmood A, Shi P, Raja MAZ, Zameer A, Chaudhary NI. Design of backtracking search heuristics for parameter estimation of power signals. Neural Comput Appl 2021;33:1479–96. https://doi.org/10.1007/s00521-020-05029-9.

[6] Xu X, Hu Z, Su Q, Xiong Z, Liu M. Multi-objective learning backtracking search algorithm for economic emission dispatch problem. Soft Comput 2021;25:2433–52. https://doi.org/10.1007/s00500-020-05312-w.

[7] Zhang C, Zhou J, Li C, Fu W, Peng T. A compound structure of ELM based on feature selection and parameter optimization using hybrid backtracking search algorithm for wind speed forecasting. Energy Convers Manag 2017;143:360–76. https://doi.org/10.1016/j.enconman.2017.04.007.

[8] Chen D, Lu R, Zou F, Li S, Wang P. A learning and niching based backtracking search optimisation algorithm and its applications in global optimisation and ANN training. Neurocomputing 2017;266:579–94. https://doi.org/10.1016/J.NEUCOM.2017.05.076.

[9] Chen D, Zou F, Lu R, Wang P. Learning backtracking search optimisation algorithm and its application. Inf Sci (Ny). 2017;376:71–94. https://doi.org/10.1016/J.INS.2016.10.002.

[10] Fadel W, Kilic U, Ayan K. Optimal reactive power flow of power systems with two-terminal HVDC and multi distributed generations using backtracking search algorithm. Int J Electr Power Energy Syst 2021;127:106667. https://doi.org/10.1016/j.ijepes.2020.106667.

[11] Lakshmi Priya J, Jaya Christa ST. An effective hybridized GWO-BSA for resolving optimal power flow problem with the inclusion of unified power flow controller. IETE J Res 2021. https://doi.org/10.1080/03772063.2021.1942245.

[12] Ahandani MA, Ghiasi AR, Kharrati H. Parameter identification of chaotic systems using a shuffled backtracking search optimization algorithm. Soft Comput 2018;22:8317–39. https://doi.org/10.1007/s00500-017-2779-0.

[13] Yuan X, Wu X, Tian H, Yuan Y, Adnan RM. Parameter identification of nonlinear muskingum model with backtracking search algorithm. Water Resour Manag 2016;30:2767–83. https://doi.org/10.1007/s11269-016-1321-y.

[14] Zhang Y, Huang C, Jin Z. Backtracking search algorithm with reusing differential vectors for parameter identification of photovoltaic models. Energy Convers Manag 2020;223:113266. https://doi.org/10.1016/j.enconman.2020.113266.

[15] Zhao F, Zhao J, Wang L, Tang J. An optimal block knowledge driven backtracking search algorithm for distributed assembly no-wait flow shop scheduling problem. Appl Soft Comput 2021;112:107750. https://doi.org/10.1016/j.asoc.2021.107750.

[16] Bhattacharjee K, Bhattacharya A, Shah K, Patel N. Backtracking search optimization applied to solve short-term electrical real power generation of hydrothermal plant. Eng Optim 2021:1–19. https://doi.org/10.1080/0305215x.2021.1954629.

[17] Garroussi Z, Ellaia R, Talbi EG, Lucas JY. A hybrid backtracking search algorithm for energy management in a microgrid. Int J Math Model Numer Optim 2021;11:143–67. https://doi.org/10.1504/IJMMNO.2021.114481.

[18] Mehmood A, Chaudhary NI, Zameer A, Raja MAZ. Backtracking search optimization heuristics for nonlinear Hammerstein controlled auto regressive auto regressive systems. ISA Trans 2019;91:99–113. https://doi.org/10.1016/j.isatra.2019.01.042.

[19] Zhang Y. Backtracking search algorithm with specular reflection learning for global optimization. Knowl-Based Syst 2021;212:106546. https://doi.org/10.1016/j.knosys.2020.106546.

[20] Zhao F, Zhao J, Wang L, Cao J, Tang J. A hierarchical knowledge guided backtracking search algorithm with self-learning strategy. Eng Appl Artif Intell 2021;102:104268. https://doi.org/10.1016/j.engappai.2021.104268.

[21] Hannan MA, Abdolrasol MGM, Mohamed R, Al-Shetwi AQ, Ker PJ, Begum RA, Muttaqi KM. ANN based binary backtracking search algorithm for VPP optimal scheduling and cost-effective evaluation. IEEE Trans Ind Appl 2021. https://doi.org/10.1109/tia.2021.3100321. 1–1.

[22] Nama S, Saha AK, Sharma S. Performance up-gradation of symbiotic organisms search by backtracking search algorithm. J Ambient Intell Humaniz Comput 2021; 1:1–42. https://doi.org/10.1007/s12652-021-03183-z.

[23] Zaman HRR, Gharehchopogh FS. An improved particle swarm optimization with backtracking search optimization algorithm for solving continuous optimization problems. Eng Comput 2021:1–35. https://doi.org/10.1007/s00366-021-01431-6.

[24] Zhao F, Zhao J, Hu X, Zhang Y, Ma W. Backtracking search algorithm based on knowledge of different populations for continuous optimization problems. In: Proceedings of the IEEE 24th international conference on computer supported cooperative work in design CSCWD 2021; 2021. p. 90–5. https://doi.org/10.1109/CSCWD49262.2021.9437657.

[25] Nama S, Saha AK. A bio-inspired multi-population-based adaptive backtracking search algorithm. Cognit Comput 2022;14:900–25. https://doi.org/10.1007/s12559-021-09984-w.

[26] Kuyu YÇ, Onieva E, Lopez-Garcia P. A hybrid optimizer based on backtracking search and differential evolution for continuous optimization. J Exp Theor Artif Intell 2021. https://doi.org/10.1080/0952813X.2021.1872109.

[27] Zhang Y. Backtracking search algorithm driven by generalized mean position for numerical and industrial engineering problems. Artif Intell Rev 2023;56: 11985–2031. https://doi.org/10.1007/s10462-023-10463-x.

[28] Zhang Y, Zhou G, Hang P, Huang C, Huang H. An enhanced backtracking search algorithm for the flight planning of a multi-drones-assisted commercial parcel delivery system. IEEE trans Intell Transp Syst 2023;24:11396–409. https://doi.org/10.1109/TITS.2023.3281522.

[29] Nama S, Saha AK, Chakraborty S, Gandomi AH, Abualigah L. Boosting particle swarm optimization by backtracking search algorithm for optimization problems. Swarm Evol Comput 2023;79:101304. https://doi.org/10.1016/j.swevo.2023.101304.

[30] Zhang Y, Huang C, Huang H. Backtracking search algorithm with dynamic population for energy consumption problem of a UAV-assisted IoT data collection system. Eng Appl Artif Intell 2023;123:106331. https://doi.org/10.1016/J.ENGAPPAI.2023.106331.

[31] Duan H, Luo Q. Adaptive backtracking search algorithm for induction magnetometer optimization. IEEE Trans Magn 2014;50. https://doi.org/10.1109/TMAG.2014.2342192.

[32] Nama S, Saha AK, Ghosh S. A new ensemble algorithm of differential evolution and backtracking s algorithm with adaptive control parameter for function optimization. Int J Ind Eng Comput 2016;7:323–38. https://doi.org/10.5267/j.ijiec.2015.9.003.

[33] Yu K, Liang JJ, Qu BY, Cheng Z, Wang H. Multiple learning backtracking search algorithm for estimating parameters of photovoltaic models. Appl Energy 2018; 226:408–22. https://doi.org/10.1016/j.apenergy.2018.06.010.

[34] Wolpert DH, Macready WG. No free lunch theorems for optimization. IEEE Trans Evol Comput 1997;1:67–82. https://doi.org/10.1109/4235.585893.

[35] Rahnamayan S, Jesuthasan J, Bourennani F, Naterer GF, Salehinejad H, Rahnamayan S, Jesuthasan J, Bourennani F, Naterer GF, Salehinejad H. Centroid opposition-based differential evolution. Int J Appl Metaheuristic Comput 2014;5: 1–25. https://econpapers.repec.org/RePEc:igg:jamc00:v:5:y:2014:i:4:p:1-25 (accessed August 25, 2021).

[36] T. Si, D. Bhattacharya, Sine cosine algorithm with centroid opposition-based computation, (2021) 119–29. 10.1007/978-981-33-4604-8_9.

[37] Manafi E, Tavakkoli-Moghaddam R, Mahmoodjanloo M. A centroid opposition-based coral reefs algorithm for solving an automated guided vehicle routing problem with a recharging constraint. Appl Soft Comput 2022;128:109504. https://doi.org/10.1016/J.ASOC.2022.109504.

[38] Liao L, Zhou Y. A neighborhood centroid opposition-based grasshopper optimization algorithm. J Phys Conf Ser 2019;1176:032044. https://doi.org/10.1088/1742-6596/1176/3/032044.

[39] Tizhoosh HR. Opposition-based learning: a new scheme for machine intelligence. In: Proceedings of the - international conference on computational intelligence for modelling, control and automation; 2005. p. 695–701. https://doi.org/10.1109/cimca.2005.1631345. CIMCA 2005 Int. Conf. Intell. Agents, Web Technol. Internet.

[40] Nama S. A modification of I-SOS: performance analysis to large scale functions. Appl Intell 2021;51:7881–902. https://doi.org/10.1007/S10489-020-01974-Z.

[41] Nama S, Saha AK. A novel hybrid backtracking search optimization algorithm for continuous function optimization. Decis Sci Lett 2019;8:163–74. https://doi.org/10.5267/j.dsl.2018.7.002.

[42] Lin J. Oppositional backtracking search optimization algorithm for parameter identification of hyperchaotic systems. Nonlinear Dyn 2015;80:209–19. https://doi.org/10.1007/s11071-014-1861-8.

[43] X.S. Yang, A new metaheuristic bat-inspired algorithm, in: Studies in computational intelligence, 2010: pp. 65–74. 10.1007/978-3-642-12538-6_6.

[44] Yang XS, Deb S. Cuckoo search via Lévy flights. In: Proceedings of the world congress on nature & biologically inspired computing NABIC; 2009. p. 210–4. https://doi.org/10.1109/NABIC.2009.5393690. 2009 - Proc.

[45] X.-S.S. Yang, M. Karamanoglu, Nature-inspired metaheuristic algorithms 2nd Edition, 2013. www.luniver.com (accessed August 31, 2021).

[46] Yang XS. Flower pollination algorithm for global optimization. Lecture notes in computer science. Berlin, Heidelberg: Springer; 2012. p. 240–9. https://doi.org/10.1007/978-3-642-32894-7_27 (Including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics).

[47] Kennedy J, Eberhart R. Particle swarm optimization. In: Proceedings of the ICNN'95 - international conference on neural networks. 4; 1995. p. 1942–8. https://doi.org/10.1109/ICNN.1995.488968. n.d.

[48] Storn R, Price K. Differential evolution – a simple and efficient heuristic for global optimization over continuous spaces. J Glob Optim 1997;11:341–59. https://doi.org/10.1023/A:1008202821328. 1997 114.

[49] D. Karaboga, An idea based on honey bee swarm for numerical optimization, Tech. Rep. TR06, Erciyes Univ. (2005) 10. https://www.semanticscholar.org/paper/AN-IDEA-BASED-ON-HONEY-BEE-SWARM-FOR-NUMERICAL-Karaboga/cf20e34a1402a115523910d2a4243929f6704db1 (accessed August 31, 2021).

[50] Mirjalili S. Moth-flame optimization algorithm: a novel nature-inspired heuristic paradigm. Knowl-Based Syst 2015;89:228–49. https://doi.org/10.1016/j.knosys.2015.07.006.

[51] Mirjalili S. SCA: a sine cosine algorithm for solving optimization problems. Knowl-Based Syst 2016;96:120–33. https://doi.org/10.1016/j.knosys.2015.12.022.

[52] Zhao J, Tang D, Liu Z, Cai Y, Dong S. Spherical search optimizer: a simple yet efficient meta-heuristic approach. Neural Comput Appl 2020;32:9777–808. https://doi.org/10.1007/s00521-019-04510-4.

[53] Kaur S, Awasthi LK, Sangal AL, Dhiman G. Tunicate swarm algorithm: a new bio-inspired based metaheuristic paradigm for global optimization. Eng Appl Artif Intell 2020;90:103541. https://doi.org/10.1016/j.engappai.2020.103541.

[54] Mirjalili S, Gandomi AH, Mirjalili SZ, Saremi S, Faris H, Mirjalili SM. Salp swarm algorithm: a bio-inspired optimizer for engineering design problems. Adv Eng Softw 2017;114:163–91. https://doi.org/10.1016/j.advengsoft.2017.07.002.

[55] Dhiman G, Kumar V. Seagull optimization algorithm: theory and its applications for large-scale industrial engineering problems. Knowl-Based Syst 2019;165: 169–96. https://doi.org/10.1016/j.knosys.2018.11.024.

[56] Olorunda O, Engelbrecht AP. Measuring exploration/exploitation in particle swarms using swarm diversity. In: Proceedings of the IEEE congress on evolutionary computation CEC 2008; 2008. p. 1128–34. https://doi.org/10.1109/CEC.2008.4630938.

[57] Bevan JM, Kendall MG. Rank correlation methods. . Stat 1971;20:74. https://doi.org/10.2307/2986801.

[58] C.T. Yue, K.V. Price, P.N. Suganthan, J.J. Liang, M.Z. Ali, B.Y. Qu, N.H. Awad, P.P. Biswas, Problem definitions and evaluation criteria for the CEC 2020 special session and competition on single objective bound constrained numerical optimization, Zhengzhou Univ. Zhengzhou China Nanyang Technol. Univ. Singapore. (2019). https://www.ntu.edu.sg/home/epnsugan/index_files/CEC2020/CEC2020-2.htm.

[59] Wang Y, Liu ZZ, Li J, Li HX, Yen GG. Utilizing cumulative population distribution information in differential evolution. Appl Soft Comput J 2016;48:329–46. https://doi.org/10.1016/j.asoc.2016.07.012.

[60] Nasir M, Das S, Maity D, Sengupta S, Halder U, Suganthan PN. A dynamic neighborhood learning based particle swarm optimizer for global numerical optimization. Inf Sci (Ny). 2012;209:16–36. https://doi.org/10.1016/j.ins.2012.04.028.

[61] Noel MM, Muthiah-Nakarajan V, Amali GB, Trivedi AS. A new biologically inspired global optimization algorithm based on firebug reproductive swarming behaviour [Formula presented]. Expert Syst Appl 2021;183:115408. https://doi.org/10.1016/j.eswa.2021.115408.

[62] Dhiman G, Kumar V. Spotted hyena optimizer: a novel bio-inspired based metaheuristic technique for engineering applications. Adv Eng Softw 2017;114: 48–70. https://doi.org/10.1016/j.advengsoft.2017.05.014.

[63] Nowcki H. Optimization in pre-contract ship design. Comput Appl Autom Shipyard Oper Sh Des 1974;2:327–38. https://trid.trb.org/view/14541. accessed November 19, 2020.

[64] Gandomi AH, Yang XS, Alavi AH. Cuckoo search algorithm: a metaheuristic approach to solve structural optimization problems. Eng Comput 2013;29:17–35. https://doi.org/10.1007/s00366-011-0241-y.

[65] Zhang M, Luo W, Wang X. Differential evolution with dynamic stochastic selection for constrained optimization. Inf Sci (Ny) 2008;178:3043–74. https://doi.org/10.1016/J.INS.2008.02.014.

[66] Sadollah A, Bahreininejad A, Eskandar H, Hamdi M. Mine blast algorithm: a new population based algorithm for solving constrained engineering optimization problems. Appl Soft Comput 2013;13:2592–612. https://doi.org/10.1016/J.ASOC.2012.11.026.

[67] Cheng MY, Prayogo D. Symbiotic organisms search: a new metaheuristic optimization algorithm. Comput Struct 2014;139:98–112. https://doi.org/10.1016/J.COMPSTRUC.2014.03.007.

[68] Ray T, Saini P. Engineering design optimization using a swarm with an intelligent information sharing among individuals. Eng Optim 2001;33:735–48. https://doi.org/10.1080/03052150108940941.

[69] Mirjalili S, Lewis A. The whale optimization algorithm. Adv Eng Softw 2016;95: 51–67. https://doi.org/10.1016/j.advengsoft.2016.01.008.

[70] Kaveh A, Khayatazad M. A new meta-heuristic method: ray optimization. Comput Struct 2012;112–113:283–94. https://doi.org/10.1016/J.COMPSTRUC.2012.09.003.

[71] Rashedi E, Nezamabadi-pour H, Saryazdi S. GSA: a gravitational search algorithm. Inf Sci (Ny) 2009;179:2232–48. https://doi.org/10.1016/J.INS.2009.03.004.

[72] Kaveh A, Mahdavi VR. Colliding bodies optimization: a novel meta-heuristic method. Comput Struct 2014;139:18–27. https://doi.org/10.1016/J.COMPSTRUC.2014.04.005.

[73] Ragsdell KM, Phillips DT. Optimal design of a class of welded structures using geometric programming. J Eng Ind 1976;98:1021–5. https://doi.org/10.1115/1.3438995.

[74] Lee KS, Geem ZW. A new meta-heuristic algorithm for continuous engineering optimization: harmony search theory and practice. Comput Methods Appl Mech Eng 2005;194:3902–33. https://doi.org/10.1016/J.CMA.2004.09.007.

[75] Mahdavi M, Fesanghary M, Damangir E. An improved harmony search algorithm for solving optimization problems. Appl Math Comput 2007;188:1567–79. https://doi.org/10.1016/J.AMC.2006.11.033.

[76] Gupta S, Deep K. A hybrid self-adaptive sine cosine algorithm with opposition based learning. Expert Syst Appl 2019;119:210–30. https://doi.org/10.1016/J.ESWA.2018.10.050.

[77] Shi Y, Eberhart R. Modified particle swarm optimizer. In: Proceedings of the IEEE international conference on evolutionary computation ICEC; 1998. p. 69–73. https://doi.org/10.1109/ICEC.1998.699146.

[78] Mirjalili S, Mirjalili SM, Lewis A. Grey wolf optimizer. Adv Eng Softw 2014;69: 46–61. https://doi.org/10.1016/J.ADVENGSOFT.2013.12.007.

[79] Rodríguez L, Castillo O, Soria J, Melin P, Valdez F, Gonzalez CI, Martinez GE, Soto J. A fuzzy hierarchical operator in the grey wolf optimizer algorithm. Appl Soft Comput 2017;57:315–28. https://doi.org/10.1016/J.ASOC.2017.03.048.

[80] Mezura-Montes E, Coello CAC. An empirical study about the usefulness of evolution strategies to solve constrained optimization problems. Int J Gen Syst 2008;37:443–73. https://doi.org/10.1080/03081070701303470.

[81] Mittal N, Singh U, Sohi BS. Modified grey wolf optimizer for global engineering optimization. Appl Comput Intell Soft Comput 2016;2016. https://doi.org/10.1155/2016/7950348.

[82] Sayed GI, Khoriba G, Haggag MH. A novel chaotic salp swarm algorithm for global optimization and feature selection. Appl Intell 2018;48:3462–81. https://doi.org/10.1007/S10489-018-1158-6/TABLES/15.

[83] Apinantanakon W, Sunat K. OMFO: a new opposition-based moth-flame optimization algorithm for solving unconstrained optimization problems. Adv Intell Syst Comput 2018;566:22–31. https://doi.org/10.1007/978-3-319-60663-7_3.

[84] Sahoo SK, Saha AK, Sharma S, Mirjalili S, Chakraborty S. An enhanced moth flame optimization with mutualism scheme for function optimization. Soft Comput 2022; 26:2855–82. https://doi.org/10.1007/S00500-021-06560-0/TABLES/18.

[85] Arora S, Singh S. Butterfly optimization algorithm: a novel approach for global optimization. Soft Comput 2019;23:715–34. https://doi.org/10.1007/S00500-018-3102-4/METRICS.

[86] Fleury C. Sequential convex programming for structural optimization problems. Optim Large Struct Syst 1993:531–53. https://doi.org/10.1007/978-94-010-9577-8_25.

[87] Wang GG. Adaptive response surface method using inherited Latin hypercube design points. J Mech Des 2003;125:210–20. https://doi.org/10.1115/1.1561044.

[88] Youn BD, Choi KK. A new response surface methodology for reliability-based design optimization. Comput Struct 2004;82:241–56. https://doi.org/10.1016/J.COMPSTRUC.2003.09.002.

[89] Yildiz AR, Abderazek H, Mirjalili S. A comparative study of recent non-traditional methods for mechanical design optimization. Arch Comput Methods Eng 2020;27: 1031–48. https://doi.org/10.1007/S11831-019-09343-X/TABLES/16.

[90] Karaboga D, Basturk B. A powerful and efficient algorithm for numerical function optimization: artificial bee colony (ABC) algorithm. J Glob Optim 2007;39:459–71. https://doi.org/10.1007/S10898-007-9149-X/METRICS.

[91] Mirjalili S. The ant lion optimizer. Adv Eng Softw 2015;83:80–98. https://doi.org/10.1016/J.ADVENGSOFT.2015.01.010.

[92] Veeramani C, Sharanya S. An improved evaporation rate-water cycle algorithm based genetic algorithm for solving generalized ratio problems. Res 2021;55: 461–80. https://doi.org/10.1051/ro/2020045.

[93] Eskandar H, Sadollah A, Bahreininejad A, Hamdi M. Water cycle algorithm – a novel metaheuristic optimization method for solving constrained engineering optimization problems. Comput Struct 2012;110–111:151–66. https://doi.org/10.1016/J.COMPSTRUC.2012.07.010.