Soft Computing
https://doi.org/10.1007/s00500-022-06837-y

OPTIMIZATION

A new modified social engineering optimizer algorithm for engineering applications

Fariba Goodarzian[1] · Peiman Ghasemi[2] · Vikas Kumar[3] · Ajith Abraham[1,4]

Accepted: 25 January 2022

Abstract

Nowadays, a great deal of attention is paid to metaheuristic algorithms to reach the approximate solution in an acceptable computational time. As one of the recent-developed successful metaheuristics, Social Engineering Optimizer (SEO) algorithm is according to the inspiration of the rules of social engineering to solve approximate optimization problems. In this research, a Modified Social Engineering Optimizer algorithm (MSEO) by using an adjustment operator is proposed in which there are some assessment criteria for defender and attacker to determine and calculate the weight simultaneously for the first time. This enhancement comprises adding adjustment operators to improve the performance of SEO in terms of search accuracy and running time. Most notably, this operator is utilized to make a better new generation and improve the interaction between the search phases. The adjustment operator strategy is also applied to a novel division based on the best person. As an extensive comparison, the suggested algorithm is tested on fourteen standard benchmark functions and compared with ten well-established and recent optimization algorithms as well as the main version of the SEO algorithm. This algorithm is also tested for sensitivities on the parameters. In this regard, a set of engineering applications were provided to prove and validate the MSEO algorithm for the first time. The experimental outcomes show that the suggested algorithm produces very accurate results which are better than the SEO and other compared algorithms. Most notably, the MSEO provides a very competitive output and a high convergence rate.

Keywords Metaheuristic algorithms · Modified social engineering optimizer · Benchmark functions · Engineering applications

✉ Fariba Goodarzian
Fariba.Goodarzian@mirlabs.org;
Faribagoodarzian7070@gmail.com

Peiman Ghasemi
Peiman.Ghasemi@gutech.edu.om

Vikas Kumar
Vikas.Kumar@uwe.ac.uk

Ajith Abraham
ajith.abraham@ieee.org

[1] Machine Intelligence Research Labs (MIR Labs), Scientific Network for Innovation and Research Excellence, 11, 3rd Street NW, P.O. Box 2259, Auburn, WA 98071, USA

[2] Department of Logistics, Tourism and Service Management, German University of Technology in Oman (GUtech), Muscat, Oman

[3] Bristol Business School, University of the West of England, Bristol BS16 1QY, UK

[4] Center for Artificial Intelligence, Innopolis University, Innopolis, Russia

1 Introduction

The needs and benefits of optimization techniques based on metaheuristics have motivated many scholars and scientists in the last few decades to apply these algorithms to solve NP-hard and complicated optimization problems (Abed-Alguni et al. 2021; Abed-Alguni and Alkhateeb 2017; Alawad and Abed-Alguni 2021a; Abed-Alguni and Alawad 2021). Today, mathematical optimization techniques have played an important role in the industrial and non-industrial sectors (Alawad and Abed-Alguni 2021b). Optimization methods and algorithms are divided into two categories: exact algorithms and approximate algorithms (Wang et al. 2018). Approximate algorithms fall into two general categories: heuristic and metaheuristic approaches (Sacramento et al. 2019). Also, metaheuristic approaches are divided into two groups: single-solution instead of population-based algorithms (Goodarzian et al. 2021a). An

Published online: 22 February 2022

acceptable computational time with the ability to find the global solution is one of the key reasons for researchers in this field to explore new modifications and hybridizations of recent metaheuristics to better solve many optimization problems in different fields of engineering (Rakshit et al. 2014; Okulewicz and Mańdziuk 2019).

The term "metaheuristic" was first introduced by Glover (Glover 1989) when the Tabu Search (TS) method was introduced as a novel method. Modern heuristic methods include the family of Evolutionary algorithms, swarm intelligence, Ant Colony Optimization (ACO) algorithm, Variable Neighborhood Search (VNS) and Greedy Search (GS) algorithms (Fathollahi-Fard et al. 2019). Briefly, we can say that the metaheuristic algorithms are advanced and general search strategies providing steps and benchmarks that are very effective in fleeing local optimal traps (Zhang et al. 2020).

Generally, to find optimal solutions and to solve optimization problems, metaheuristic algorithms have been used. Then, the metaheuristic algorithms could be categorized into four groups: local search versus global search-based, Single solution versus population-based, and Swarm intelligence-based, versus Nature-inspired-based.

In order to find optimal solutions, several metaheuristic ideas were presented to enhance local search heuristics such as Simulated Annealing (SA), TS, VNS, GS, and SEO. These algorithms could both be categorized as global search or local search-based metaheuristics. Other global search algorithms that are not local search-based are population-based algorithms such as ACO, Particle Swarm Optimization (PSO), Imperialist Competitive Algorithm (ICA), Harmony Search (HS), an evolutionary algorithm based metaheuristics.

In terms of a single solution and population-based searches, single solution algorithms concentrate on improving and modifying a single candidate solution. SA, VNS and SEO offer a single solution. Population-based algorithms improve and maintain multiple candidate solutions that utilize population traits to conduct the search. Population-based algorithms involve Genetic Algorithm (GA) and PSO.

The third group of algorithms is Swarm intelligence based on self-organized agents and collective behavior of the decentralized population in a swarm. ACO, Artificial Bee Colony (ABC), Artificial Natural Network (ANN), and PSO are instances of this group.

An active field of research is nature-inspired metaheuristic algorithm design. Recently, evolutionary computation-based metaheuristics are inspired by natural systems. Nature treats as a source of mechanisms, principles, and concepts for artificial computing systems designed to cope with complex computational problems such as PSO, ICA, ABC, Firefly Algorithm (FA), and ACO algorithm.

These classifications can help us to have a better focus on the properties of metaheuristics which are very useful to develop a new one. In addition to these, the concepts of the main algorithms mentioned above are characterized as follows. For example, the GA is a metaheuristic inspired by the process of natural choice, which depends on the larger group of evolutionary algorithms. In general, it is a repetition-based algorithm, most of which are chosen as random processes that are composed of parts of the fitness function, mutation, crossover, and selection (Tam et al. 2019; Yadegari et al. 2019). Mitchell et al. (1992) introduced GA following the concept of Darwin's theory of evolution. GA is one of the random search algorithms, which is derived from nature (Pech et al. 2019). Note that the crossover and mutation are two search engines of GA to focus on exploration and exploitation phases.

Then, the PSO algorithm was originally introduced by Kennedy and Eberhart (1995). The PSO algorithm is a collective search algorithm, which is according to the social behavior of bird's categories (Hajipour et al. 2020). By updating the agents based on the local optimum and global one makes the PSO algorithm perform both diversification and intensification phases properly (Fakhrzad and Goodarzian 2019). Generally, a trade-off between these two search phases plays a key factor in many earlier metaheuristics (Pech et al. 2019; Kennedy and Eberhart 1995; Hajipour et al. 2020; Fakhrzad and Goodarzian 2019; Balaji et al. 2019).

Ant Colony Optimization (ACO) algorithm is inspired by the swarm behavior of ants. The ACO algorithm introduced by Stützle and Dorigo (1999) which is built on the intelligent behavior of ants to find the shortest path from the nest to a food source, has recently attracted the attention of scientists (Stützle and Dorigo 1999; Luan et al. 2019).

The ABC algorithm is an optimization strategy that simulates the behavior of a bee colony and was first introduced by Karaboga (2007) to optimize the real parameter. ABC algorithm is an optimization algorithm according to the collective intelligence and clever behavior of bee populations. Such an algorithm is very effective in solving real-world problems (Aslan 2019).

The Biogeography-Based Optimization (BBO) algorithm was originally introduced by Simon (Simon 2008). The BBO is a population-based algorithm inspired by the phenomenon of animal migration and birds between the islands. In reality, environmental geography is the study of the geographical distribution of environmental species (Bottani et al. 2020). Basically, in biogeography, two determinants are the value of the Habitat Suitability Index (HSI) and the Suitability Index Variables (SIV) (Goudarzi et al. 2019).

The ICA is another computational approach that is utilized to solve optimization problems of various types, which was introduced by Nazari-Shirkouhi et al. (2010). This algorithm is according to the modeling of the social-political process of the colonial phenomenon. The high popularity of this algorithm, along with its high efficiency, is more of an innovative and new aspect and attractive to optimization experts (Ramírez et al. 2019; Lei et al. 2018).

The FA is another nature-inspired metaheuristic algorithm introduced by Yang (2010) with the use of inspiration of the flashing behavior of fireflies. The basis of this algorithm is the behavior of fireflies in streaming light from itself. Most fireflies can capture partnerships for mating by lighting, warning other fireflies and trapping smaller insects for hunting. The intensity rate of light available for other light source fireflies depends on the distance from the source, the intensity of the light source and the absorption power of light, so the fireflies are generally visible to a limited distance (Shang et al. 2020).

Over the last decade, many such metaheuristic optimization algorithms are based on the animals' inspiration or artificial human roles have been designed to solve optimization problems more efficiently (Shadravan et al. 2019; Castellani et al. 2019). There are also many modifications of BEE algorithms, e.g., BEEs show a great deal of attention during the last decade (Kamaruddin and Latif 2019). The BEEs algorithm was provided by Ghanbarzadeh et al. (Pham et al. 2005) which is a population-based search algorithm. This algorithm performs a local search type with a global search and can be utilized for both combinatorial optimization and continuous optimization. The only condition for using the BBEs is that some measurements of the distance among solutions have been defined. The effectiveness and specific capabilities of the BEEs have been proven in a number of studies.

As one of the earliest metaheuristics inspired by music, the Harmony Search (HS) algorithm developed by Geem et al. (2001) is a successful metaheuristic algorithm for routing in wireless sensor networks and in order to increase the life span of these types of networks (Yi et al. 2019). HS algorithm is one of the easiest and newest metaheuristic algorithms that have been inspired by the simultaneous playback process of the orchestra music stream in the optimal search process in optimization problems. In other words, there is a similarity between finding an optimal solution to the complex problem and the process of performing music (Dhiman and Kaur 2019).

Abualigah (2019) presented an algorithm for solving the Text document (TD) clustering problem. The k-mean clustering method is used to evaluate the performance of the obtained subsets. Finally, 4 krill herd algorithms are proposed to solve the TC problem. For the evaluation process, seven benchmarks are used. Abualigah et al. (2021a) presented a new optimizer inspired by the behavior of Aquila Optimizer (AO). To validate the proposed algorithm, a set of different numerical problems (ten functions from the CEC2019 benchmark, 29 functions from the CEC2017 benchmark and 23 classical benchmark functions) were used. Abualigah et al. (2021b) presented a novel metaheuristic algorithm called Arithmetic Optimization Algorithm (AOA). The performance of the proposed algorithm is evaluated on 29 benchmark functions. The results of the proposed algorithm sufficiently prove its superiority in the ability to avert trapping of the local optima.

Recently, Fathollahi-Fard et al. (2018) developed the Social Engineering Optimization (SEO) algorithm which is inspired by the rules of social engineering, an emerging phenomenon in today's real world. They also reviewed the metaheuristics from 1975 till 2017 and found that there are more than a hundred well-known metaheuristics in the literature. However, there is no similar algorithm like SEO only employing two solutions to search and has fast procedures to find the global solution. The application and development of SEO have been explored recently by a few studies. However, research on SEO is still scarce (Goodarzian et al. 2021b; Elarbi et al. 2018; Meng et al. 2016; Li et al. 2015). Given the popularity and high efficiency of the SEO motivated our attempt to develop another efficient version of this algorithm that is efficient and more intelligent than the original one. The SEO algorithm uses only the values of the objective function to perform the optimization process and does not require additional information such as the function derivative. Due to the simplicity of the search process of SEO, it works very quickly and efficiently. The SEO algorithm is also very flexible and works with all kinds of objective functions and constraints in the search space (Goodarzian et al. 2021c). The significant advantage of the SEO algorithm is a new simple and efficient single-solution metaheuristic. One of the other important advantages of the SEO algorithm is the Social Engineering (SE) phenomenon and its techniques. In comparison with other single-based metaheuristics, the SEO algorithm starts with two initial random solutions that include attackers and defenders. The optimal solution is the attacker. In the process of SE, the attacker requests to defeat the defender using SE attacks' skills. These characteristics cause SEO simple to implement, very proper, and more robust for single-solution-based computation.

One of the variations of the Social Engineering Optimization algorithm is called MSEO_1. In the original version of SEO, the attacker aims to assess the traits of the defender randomly to select an efficient one, but in MSEO_1 a roulette wheel strategy is considered to select an appropriate trait from the defender. Therefore, the chance of the first trait is more than other traits. Another variation of the SEO algorithm is called MSEO_2, which focuses on proposing a new spot for the defender inspired by a recent real technique called reverse social engineering. In this variation, instead of directly contacting the defender, the attacker tries to make the defender believe that they are a trustworthy individual. Another variation of the SEO algorithm is called MSEO_3. The contribution of this variation is to have a dynamic parameter for the number of attacks and the number of attacks in each iteration is not fixed (Fathollahi-Fard et al. 2019).

The significant contribution of this study includes, firstly, a Modified SEO (MSEO) metaheuristic algorithm with a novel adjustment operator to enhance its efficiency into running time and search validity. This feature makes MSEO superior to the SEO algorithm. Another advantage of the MSEO is that algorithm nature works in high iterations based on the logic of convergence, which preserves the best solution. Additionally four engineering applications are stated for the first time including (i) location-allocation problem for earthquake evacuation planning, (ii) pharmaceutical supply chain network design, (iii) truck scheduling problem in a cross-docking system, and (iv) production planning under uncertain seasonal demand.

In this paper, a new conceptual framework for social engineering to develop a MSEO algorithm is described. Besides, the main goal is the development of adjusting operators based on the defender and attacker assessment criteria, to speed up convergence, hence making the method more reliable for a wide range of practical applications while preserving the traits of the original SEO. This new procedure can determine and calculate the weight of each of them. The enhancement comprises adding adjustment operators to improve its performance in terms of search precision and running time and thus this operator is utilized to make a new generation. The adjustment operator strategy is also applied to a novel division based on the best person and other random people steps. Also, the enhancement comprises of the defender and the attacker assessment criteria showing the quality of the solution for a population for the defender as well as the attacker. Based on these suppositions, the MSEO is established and tested on fourteen standard benchmark functions and compared with other aforementioned metaheuristic methods. As an extensive comparison, the experimental results indicate that MSEO is more efficient than original SEO, GA, FA, HS, PSO, ACO, ABC, BEEs, ICA, BBO algorithms. To validate the MSEO, four engineering applications are provided and p values of the Wilcoxon test are used in this research.

The rest of this research is examined as follows: Sect. 2 describes the social engineering optimizer in general. Section 3 describes a modified version of this metaheuristic. Section 4 proposes the computational analyses and comparison among different criteria and other algorithms. The engineering applications for the MSEO algorithm is examined completely in Sect. 5. Finally, Sect. 6 concludes our work and describes some future research issues.

2 SEO algorithm

Despite the fact that in recent years, a large number of metaheuristic algorithms have been provided, scholars nevertheless utilize traditional algorithms to solve problems. In addition, over the past two decades, most of the metaheuristic methods are population-based and involve a large number of steps and parameters that make them difficult to understand and perceive. This paper, therefore, proposes an intelligent algorithm like many of the most recent metaheuristic methods, and yet very simple, which only includes four steps and three parameters for adjustment. The SEO was provided by Fathollahi-Fard et al. (2018) inspired by the rules of social engineering as an

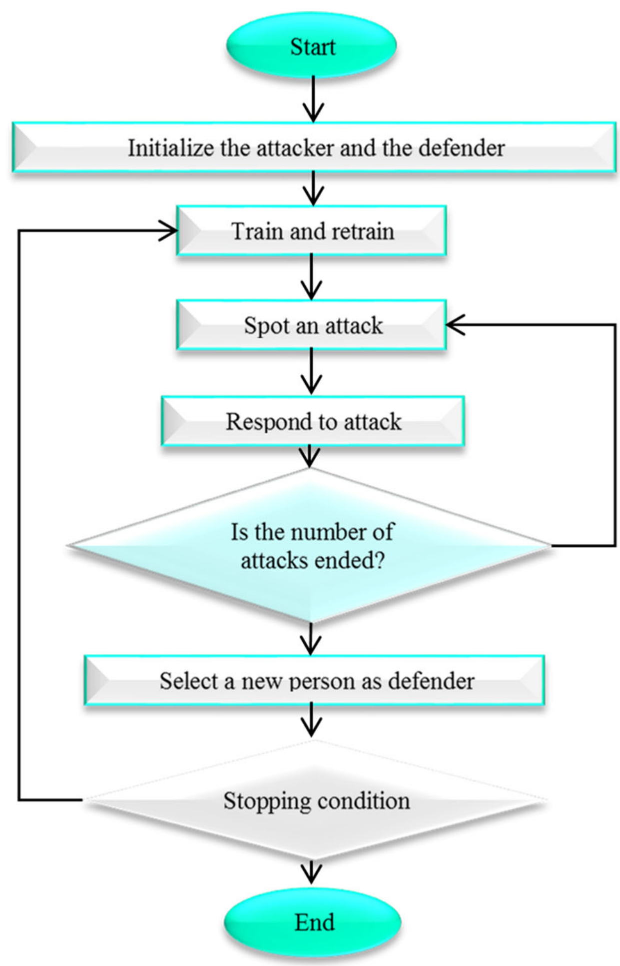


Fig. 1 The flowchart of SEO

```
T1=clock;
Initialize attacker and defender
It=1;
while solving_time < Max_time
    Do training and retraining;
    Num_attack=1;
    while Num_attack < Max_attack
        Spot an attack;
        Check the boundary;
        Respond to attack;
        if the OF of defender is lower than attacker
          Exchange the defender and attacker position;
        endif
        Num_attack= Num_attack+1;
    endwhile
    Create a new solution as defender;
    It=It+1;
    T2=clock;
    Solving_time=T2- T1;
endwhile
Return attacker.
```

Fig. 2 The pseudo-code of SEO

emerging phenomenon in today's real world. Thus, this method starts with two random solutions namely the attacker and the defender. In addition, search phases are according to the rules of social engineering in which the attacker uses certain techniques to obtain the desired aims. More details about SEO can be seen in Fathollahi-Fard et al. (2018). The flowchart and pseudo-code of the SEO are indicated in Figs. 1 and 2. Then, the process of the SEO is explained as follows:

Step 1: Initialize the attacker and the defender.
Step 2: Training and retraining.
Step 3: Spot an attack.
Step 4: Respond to attack.
Step 5: Select a new as a defender.
Step 6: Stop condition.

3 MSEO approach

Since this metaheuristic is recently developed, there are only a few studies to propose new modifications and hybridizations of this algorithm (Fathollahi-Fard et al. 2018; Goodarzian et al. 2021b, c, d, e; Elarbi et al. 2018; Meng et al. 2016; Li et al. 2015). The employed producers in this study differ from similar papers in the literature. In this section, some reformation to the SEO algorithm including (i) the number of attackers and defenders, (ii) defender and attacker evaluation criteria, and (iii) adjustment Operator is provided completely in the next subsections. The process of the MSEO algorithm is stated in the following subsections.

3.1 The number of attackers and defenders

In this algorithm, there are two different search factors that include the attacker and the defender. The number of attackers and defenders as the population is considered in this search space. The number of attackers is randomly selected from 65 to 90% of the total population. The number of attackers is obtained from Eq. (1):

$$N_a = floor[(0.9 - rand \times 0.25) \times N] \tag{1}$$

where *rand* is a random number between [0, 1]. Meanwhile, $floor(0)$ mapping a real number is an integer. The number of defenders (N_d) as complementary between (N) and (N_a) is calculated as Eq. (2):

$$N_d = N - N_a \tag{2}$$

Moreover, the total population of (M) is formed by elements of N and is divided into two subgroups G and Q. Therefore, G and Q sizes are controlled by a predetermined constant ρ ratio. The group of G is a set of attackers $G = \{G_1, G_2, \ldots, G_{N_a}\}$. Meanwhile, the group of Q includes defenders $Q = \{Q_1, Q_2, \ldots, Q_{N_d}\}$. Where in, $M = \{M_1, M_2, \ldots, M_N\}$. So that, $M = m_1 = G_1, M_2 = G_2, \ldots, M_{N_a} = G_{N_a}, M_{N_a+1} = Q_1, M_{N_a+2} = Q_2, \ldots, M_N = Q_{N_d}$.

3.2 Defender and attacker evaluation criteria

In this way, each defender and the attacker have one weight W_a and W_d, which indicates the quality of the solution to the defender d and the attacker a of the population (M). Furthermore, Eqs. (3) and (4) have been used to calculate the weight of each attacker and defender.

$$W_a = \frac{K(M_a) - worst_m}{best_m - worst_m} \tag{3}$$

$$W_d = \frac{K(M_d) - worst_m}{best_m - worst_m} \tag{4}$$

where $K(M_a)$ and $K(M_d)$ capability is obtained by evaluating the attacker's position and defender and according to the objective function $K(0)$. Values $worst_m$ and $best_m$ are defined as Eqs. (5) and (6):

$$best_m = \min_{i \in (1,2,\ldots,M)} (K(M_i)) \tag{5}$$

$$worst_m = \max_{i \in (1,2,\ldots,M)} (K(M_i)) \tag{6}$$

3.3 Adjustment operator

This improved algorithm is introduced with an adjustment operator to enhance its efficiency in terms of search precision and running time. This operator is used to make a novel generation. The size of this part is equal to the size of G and Q. This operator creates a new division according to the best person and other random people from G and Q. Also, we assume that $Y_{o,j}^{t+1}$ the value of the element j is the number of individuals o, then $Y_{o,j}^{t+1}$ generated based on Eq. (7):

$$Y_{o,j}^{t+1} = \begin{cases} Y_{best,j}^{t} & rand \leq \rho \\ Y_{r3,j}^{t} & rand > \rho \end{cases} \tag{7}$$

where r is a random number obtained from Eq. (8), $rand$ is a random number of uniform distribution and δ is a fixed value equal to 1.2 . Also, t is the number of iterations.

$$r = rand \times \delta \tag{8}$$

In Eq. (8), parts of the newly created person are updated according to Eq. (9), if the number of other random numbers created is greater than the adjustment rate. The adjustment rate is shown by the BAR and the set is equal to the fixed partition. In Eq. (10) dy, a local search is represented by the training and retraining of the defender and the attacker in each other in this algorithm. μ an element that controls the penetration of dy in the updating process.

$$Y_{o,j}^{t+1} = Y_{o,j}^{t+1} + \mu\left(dL_y - 0.5\right) \tag{9}$$

$$dy = RT\left(Y_o^t\right) \tag{10}$$

3.4 Computational method of ISEO

The computational method for the proposed algorithm is as follows:

Step 1: Given M as the number of members of the m-dimensional set, the number of defenders M_d and the number of attackers M_a in the total population is defined as:

$$N_a = floor[(0.9 - rand \times 0.25) \times N] \tag{11}$$

$$N_d = N - N_a \tag{12}$$

where $rand$ a random number is between [0,1]. Meanwhile, $floor(0)$ mapping a real number is an integer.

Step 2: Initialization is randomly for the defender Eq. (13), the attacker Eq. (14), and for the set of member Eq. (15). In Fig. 3, an initialized pseudo-code is presented.

$$Q = \{Q_1, Q_2, \ldots, Q_{N_d}\} \tag{13}$$

$$G = \{G_1, G_2, \ldots, G_{N_a}\} \tag{14}$$

$$M = m_1 = G_1,\ M_2 = G_2, \ldots, M_{N_a} = G_{N_a},\ M_{N_a+1} = Q_1, M_{N_a+2} = Q_2, \ldots, M_N = Q_{N_d} \tag{15}$$

Step 3: At this stage, we intend to demonstrate the defender's attacker's training and retraining. In this way, the attacker chooses the most influential trait. For this purpose, α percent of the characteristics are selected randomly and repeated directly in the same characteristic in the defender. The number of traits for training is indicated in Eq. (16).

$$N_{Train} = round\{\alpha,\ nVar\} \tag{16}$$

where α percent is selected traits and $nVar$ nVar nVar is the total number of traits per person. Therefore, N_{Train} NTrain NTrain is the number of characteristics that are randomly experimented with the defender.

Step 4: Calculate the weight of each defender and attacker from the population of N, which is expressed in the pseudo-code in Fig. 4.

Step 5: In order to carry out an attack, this algorithm proposes four various techniques, including obtaining, phishing, diversion theft, and pretext.

Step 6: This improved algorithm is introduced with an adjustment operator to enhance its efficiency in terms of search precision and running time. In the following, we will express its pseudo-code in Fig. 5.

For *(i =1;i< M_d +1;i++)*
  For *(j=1; j<n+1; j + +)*
    $d_{i,j}^0 = p_j^{low} + rand(0,1)\left(p_j^{high} - p_j^{low}\right)$
  End for
End for
For *(k=1;i< M_a +1;k++)*
  For *(j=1; j<n+1; k++)*
    $a_{k,j}^0 = p_j^{low} + rand(0,1)\left(p_j^{high} - p_j^{low}\right)$
  End for
End fore

Fig. 3 The pseudo-code of initialization

For *(a=1; a<M+1; a++)*
  For *(d=1; d<M+1; d++)*
    $W_a = \dfrac{K(M_a) - worst_m}{best_m - worst_m}$
    $W_d = \dfrac{K(M_d) - worst_m}{best_m - worst_m}$
    Where $best_m = \max_{i \in (1,2,\ldots,M)} \left(K(M_i)\right)$ *and* $\min_{i \in (1,2,\ldots,M)} \left(K(M_i)\right)$
  End for
End for

Fig. 4 The pseudo-code of calculating the weight of each attacker and defender

For *i=1 to num Q* **do**
Scale=max Step Size(Iter)
*Step Size=exprnd (2*Max Iter)*
DeltaY=RT(StepSize,Dim)
For *j=1 to Dim* **do**
If *rand* $\geq$ *partition,* **then**
Q(i, j)=Best(j)
Else
r_4*=round (num Q*rand+0.5)*
Q(i,j)=Population(r_4,j)
If *rand>BAR,* **then**
Q(i,j)= Q(i,j)+scale(delta Y(j)-0.5)*
End if
End if
End for
End for

Fig. 5 The pseudo-code of adjustment operator

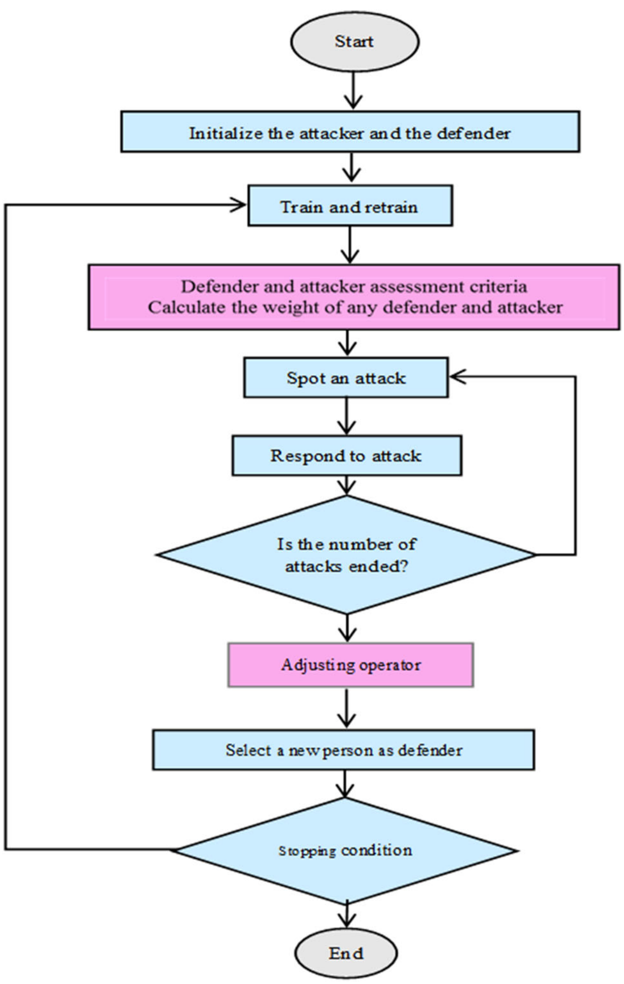


Fig. 6 The flowchart of proposed MSEO

MSEO algorithm

T_1=clock;
Initialize attacker and defender
It=1;
For *(i =1;i< M_d +1;i++)*
For *(j=1; j<n+1; j + +)*
$d^0_{i,j} = p^{low}_j + rand(0,1)(p^{high}_j - p^{low}_j)$
End for
End fore
For *(k=1;i< M_a +1;k++)*
For *(j=1; j<n+1; k++)*
$a^0_{k,j} = p^{low}_j + rand(0,1)(p^{high}_j - p^{low}_j)$
End for
End for
For *(a=1; a<M+1; a++)*
For *(d=1; d<M+1; d++)*
$W_a = \frac{K(M_a) - worst_m}{best_m - worst_m}$
$W_d = \frac{K(M_d) - worst_m}{best_m - worst_m}$
Where $best_m = \max_{i\in(1,2,\ldots,M)}(K(M_i))$ *and* $\min_{i\in(1,2,\ldots,M)}(K(M_i))$
End for
End for
while *solving_time < Max_time*
Do training and retraining;
Num_attack=1;
while *Num_attack < Max_attack*
Spot an attack;
Check the boundary;
Respond to attack;
if the *OF* of defender is lower than attacker
Exchange the defender and attacker position;
End if
Num_attack= Num_attack+1;
End while
For *i=1 to num Q* **do**
Scale=max Step Size(Iter)
*Step Size=exprnd (2*Max Iter)*
DeltaY=RT(StepSize, Dim)
For *j=1 to Dim* **do**
If *rand* $\geq$ *partition,* **then**
Q(i, j)=Best(j)
Else
r_4*=round (num Q*rand+0.5)*
Q(i, j)=Population(r_4,j)
If *rand>BAR,* **then**
Q(i, j)= Q(i,j)+scale(delta Y(j)-0.5)*
End if
End if
End for
End for
Create a new solution as defender;
It=It+1;
T_2=clock;
Solving_time=T_2- T_1;
End while
Return attacker.

Fig. 7 The pseudo-code of proposed MSEO

Table 1 The parameters of the proposed algorithms

Algorithm	Parameter	Value
GA	Crossover probability	0.85
	Mutation probability	0.02
	Selection mechanism	Roulette wheel
FA	Light absorption coefficient	1
	Mutation coefficient	0.2
	Mutation coefficient damping ratio	0.99
HS	Harmony memory size	20
	Number of new harmonies	20
	Harmony memory consideration rate	0.5
	Pitch Adjustment Rate	0.1
PSO	Acceleration constants	[1.5, 2.5]
	Inertia weights	[0.55, 0.85]
	Personal learning coefficient	2
	Global learning coefficient	2
ACO	Intensification factor (selection pressure)	0.5
	Deviation-distance ratio	1
ABC	The number of colony size *NP*	50
	The number of food sources	*NP/2*
	Maximum search time	100
BEEs	Neighborhood radius damp rate	0.99
	Number of scout bees	30
	Recruited bees scale	3
ICA	Number of empires/imperialists	10
	Selection pressure	1
	Assimilation coefficient	2
	Revolution probability	0.1
	Revolution rate	0.05
	Colonies mean cost coefficient	0.1
SEO	Rate of collecting data	0.2
	Rate of connecting attacker	0.08
	Number of connections	50
BBO	Habitat modification probability	1
	Immigration probability	[0, 1]
	Step size	1
	Maximum immigration	1
	Migration rates	1
	Mutation probability	0.06
MSEO	Rate of collecting data	0.2
	Rate of connecting attacker	0.08
	Number of connections	50
	Weight of defender	45
	Weight of attacker	65

Step 7: In this step, the attacker finally defeats the defender and the new defender is randomly replaced.

Step 8: If the stop criteria are met, the process ends, otherwise we will go back to step 3.

The flowchart and pseudo-code of the MSEO algorithm are indicated in Figs. 6 and 7.

4 Analysis and experimental outcomes

In this section, the experimental results of the efficiency of the MSEO have been compared with other metaheuristic methods containing SEO, GA, FA, HS, PSO, ACO, ABC, BEE, ICA, and BBO on 15 benchmark functions.

Table 2 Description of unimodal benchmark functions

No	Name	Function	Dim	Range	$f_{\min}$
F1	*Griewank*	$f(x) = f(x_1, ..., x_n) = 1 + \sum_{i=1}^{n} \frac{x_i^2}{4000} - \prod_{i=1}^{n} \cos\left(\frac{x_i}{\sqrt{i}}\right)$	25	[−600, 600]	0
F2	*Sphere*	$f(x) = f(x_1, ..., x_n) = \sum_{i=1}^{n} x_i^2$	25	[−100, 100]	0
F3	*SumSquares*	$f(x) = f(x_1, ..., x_n) = \sum_{i=1}^{n} i x_i^2$	25	[−10, 10]	0
F4	*Schwefel22.2*	$f(x) = f(x_1, ..., x_n) = \sum_{i=1}^{n} \lvert x_i \rvert + \prod_{i=1}^{n} \lvert x_i \rvert$	25	[−100, 100]	0
F5	*RotatedDiscusFunction*	$F(x) = f(M(x - o)) + F^*$	25	[−100, 100]	0

Table 3 Description of multimodal benchmark functions

No.	Name	Function	Dim	Range	$f_{\min}$
F6	*Rastrigin*	$f(x, y) = 10n + \sum_{i=1}^{n} \left(x_i^2 - 10\cos(2\pi x_i)\right)$	25	[−5.12, 5.12]	0
F7	*Shubert3*	$f(x) = f(x_1, ..., x_n) = \sum_{i=1}^{n} \sum_{j=1}^{5} j \sin((j+1)x_i + j)$	25	[−10, 10]	− 29.673
F8	*Xin-She Yang N. 4*	$f(x) = f(x_1, ..., x_n) = \left(\sum_{i=1}^{n} \sin^2(x_i) - e^{-\sum_{i=1}^{n} x_i^2}\right) e^{-\sum_{i=1}^{n} \sin^2 \sqrt{\lvert x_i \rvert}}$	25	[−10, 10]	0
F9	*Quartic*	$f(x) = f(x_1, ..., x_n) = \sum_{i=1}^{n} i x_i^4 + random[0, 1)$	25	[−1.28, 1.28]	0
F10	*Periodic*	$f(x) = f(x_1, ..., x_n) = 1 + \sum_{i=1}^{n} \sin^2(x_i) - 0.1 e^{-\sum_{i=1}^{n} x_i^2}$	25	[−10, 10]	0
F11	*Qing*	$f(x) = f(x_1, ..., x_n) = \sum_{i=1}^{n} (x^2 - i)^2$	25	[−500, 500]	0

The proposed algorithms parameters in all the experiments are shown in Table 1. The population size and the maximum number of iterations were set to 50 and 25, respectively. Also, all algorithms were run 25 times to achieve statistically significant results. Accordingly, the values of the algorithm's parameters are generated based on random data that are reported in Table 1. To compare based on running time, all the tests are performed on a Laptop with 2.50 GHz and 6.00 GB of RAM. In addition, MATLAB R2020b v9.9 software is used for the implementation of the proposed metaheuristic algorithms.

We used 15 classical benchmark functions to assess the efficiency of the MSEO in this experiment. The benchmark functions could be categorized into three categories including unimodal, multimodal, and n-dimension multimodal. Tables 2, 3, and 4 provide the benchmark functions of unimodal, multimodal, and fixed-dimension multimodal, which involve the mathematical equation, range of optimization variables, and the optimal values. To check the performance and efficiency of the proposed metaheuristic algorithms, the benchmark functions with different difficulty levels are used.

The results of the proposed metaheuristic are reported in Tables 5, 6, and 7, in which each table indicates the average of the best solution obtained of each of the proposed metaheuristic algorithms over 25 independent runs. It is clear that the MSEO has the best efficiency and is also efficient than other methods in most of the test functions. The comparative performance of the proposed algorithm on the unimodal function is shown in Table 5. MSEO was performed on all three test functions. Also, as there is no local solution, the unimodal functions are appropriate for testing the convergence. These results indicate that MSEO enhances the convergence rate of SEO. In addition, this convergence outperforms other methods. Figure 5 shows the behavior of the proposed methods on the unimodal functions based on the convergence for better comparison. Hence, the convergence of MSEO is significantly faster than other proposed methods.

Regarding the multimodal function, this trait causes them to benchmark the efficiency of algorithms to avoid

Table 4 Description of n-dimension multimodal benchmark functions

No	Name	Function	Dim	Range	f_{min}
F12	*Rosenbrock*	$f(x) = f(x_1, ..., x_n) = \sum_{i=1}^{n}\left[b\left(x_{i+1} - x_i^2\right)^2 + (a - x_i)^2\right]$	25	[−5, 10]	0
F13	*Salomom*	$f(x) = f(x_1, ..., x_n) = 1 - \cos\left(2\pi\sqrt{\sum_{i=1}^{D} x_i^2}\right) + 0.1\sqrt{\sum_{i=1}^{D} x_i^2}$	25	[−100, 100]	0
F14	*AlpineN.2*	$f(x) = f(x_1, ..., x_n) = \prod_{i=1}^{n} \sqrt{x_i}\sin(x_i)$	25	[0, 10]	2.808
F15	*Ackley*	$f(x) = f(x_1, ..., x_n) = -a.\exp\left(-b\sqrt{\frac{1}{n}\sum_{i=1}^{n} x_i^2}\right) - \exp\left(\frac{1}{n}\sum_{i=1}^{n}\cos(cx_i)\right) + a + \exp(1)$	25	[−32, 32]	0

Table 5 Mean normalized outcomes of unimodal benchmark functions (F1–F5)

F	GA	FA	HS	PSO	ACO	ABC	BEEs	ICA	SEO	BBO	MSEO
F1	3.91	5.58	2.45	4.67	5.47	5.78	3.42	3.25	1.26	4.21	1.12
F2	2.23	6.36	1.56	5.78	6.04	7.25	5.39	4.56	7.34	4.99	**1.00**
F3	10.24	50.46	7.68	19.34	15.45	14.35	7.56	25.01	8.93	9.02	**1.00**
F4	4.56	17.56	7.91	12.4	5.67	7.89	16.77	2.56	1.35	6.23	1.23
F5	3.28	2.76	2.45	4.02	4.21	3.87	3.12	5.21	1.45	5.39	1.19

Table 6 Mean normalized outcomes of multimodal benchmark functions (F6–F11)

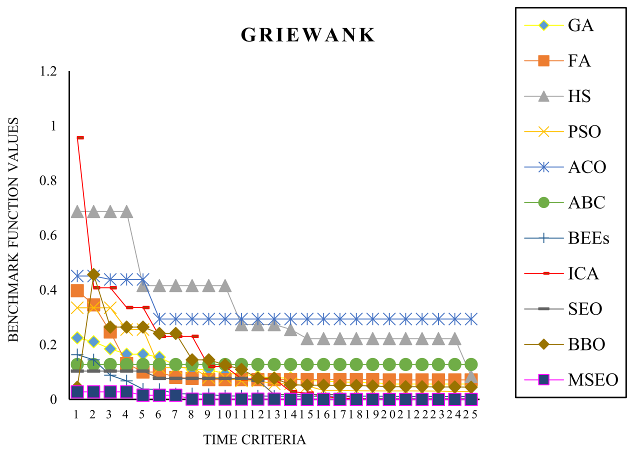
F	GA	FA	HS	PSO	ACO	ABC	BEEs	ICA	SEO	BBO	MSEO
F6	8.45	34.67	4.56	2.45	45.16	3.56	56.34	14.03	122.4	69.21	1.34
F7	18.45	3.05	3.47	11.23	13.23	23.04	5.62	46.12	6.13	5.67	**1.00**
F8	**1.00**	15.36	16.32	29.31	32.02	16.34	67.24	4.56	13.67	14.78	**1.00**
F9	19.23	18.7	56.43	5.05	5.67	76.03	3.12	11.03	4.19	23.01	**1.00**
F10	29.34	13.59	10.78	28.12	31.02	13.09	36.02	8.23	6.23	56.41	1.04
F11	9.23	10.56	5.78	34.7	2.56	34.6	7.89	13.45	2.67	41.8	**1.00**

Table 7 Mean normalized outcomes of n-dimension multimodal benchmark functions (F12–F15)

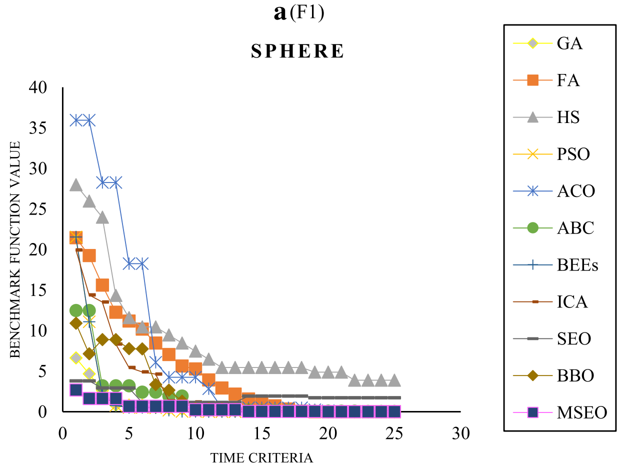
F	GA	FA	HS	PSO	ACO	ABC	BEEs	ICA	SEO	BBO	MSEO
F12	12.01	63.25	27.23	17.26	29.03	45.13	48.23	43.02	8.34	15.67	**1.00**
F13	7.24	42.81	12.03	4.02	3.71	7.57	30.31	6.34	2.45	3.12	**1.00**
F14	16.8	24.1	10.5	6.77	14.6	8.91	48.12	6.78	3.21	9.21	1.12
F15	21.7	56.8	27.1	3.45	5.76	23.5	56.25	3.61	2.54	14.67	1.67

local optima. Tables 6 and 7 indicate that the MSEO algorithm is more reliable than the other algorithms on the multimodal test functions. This indicates that the fast convergence of the MSEO algorithm according to the other proposed mechanisms in this paper does not lead to local solutions. Additionally, the result of the MSEO is better than the SEO and it is better than other methods in the majority of case studies. From Tables 5, 6, and 7, it is clear that MSEO is most effective at the obtained objective function in benchmarks (F1-F15). This proves that the local optima avoidance of the SEO algorithm has been improved by the suggested technique in this paper. Hence, Figs. 8, 9 and 10 indicate the convergence curves of the fifteen test functions, which prove the convergence of MSEO is competitive in the majority of case studies.

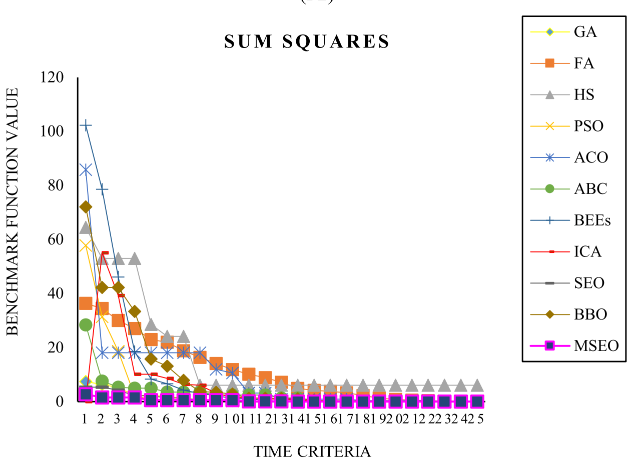
Additionally, convergence graphs of the algorithms including GA, FA, HS, PSO, ICA, ACO, ABC, BEEs, BBO, SEO, and MSEO are indicated in Figs. 8, 9 and 10 in which the process of optimization of the proposed algorithms is indicated. The values of the objective function are



a (F1)

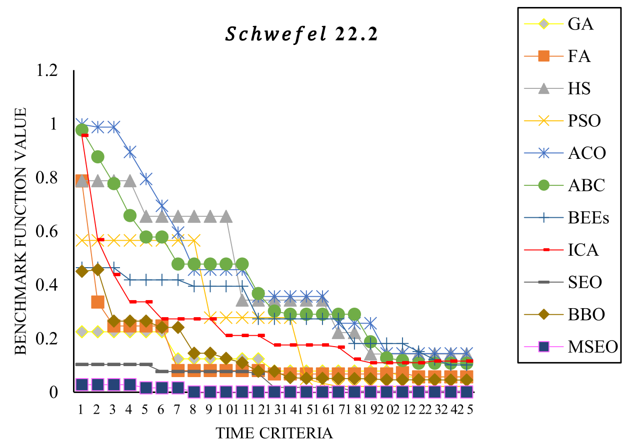


b (F2)

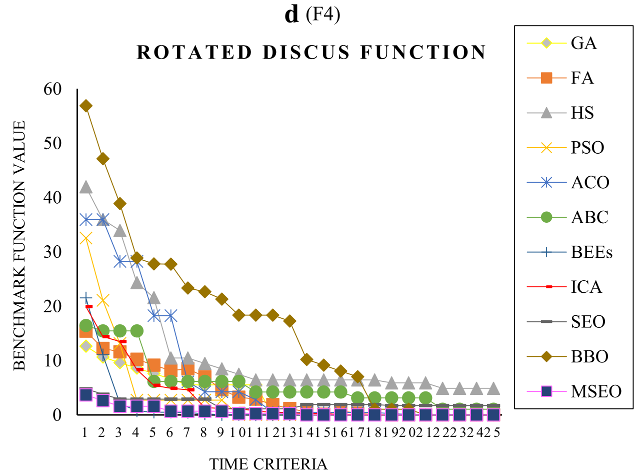


c (F3)

Fig. 8 Comparison of the convergence of the various algorithms based on unimodal functions (F1–F5)



d (F4)



q (F5)

Fig. 8 continued

indicated in Figs. 8, 9 and 10 that shows the best objective function optimum achieved from Monte Carlo simulations that are the real objective function value, not normalized.

Figure 8a (F1) indicates the results achieved for the eleven algorithms when the F1 *Griewank* function is applied. From Fig. 8a (F1), we can conclude that the MSEO during the process of optimization is better than the original SEO. Hence, MSEO showed almost the same fast convergence rate, as well as it outperformed in all generations. MSEO and SEO are the best than the other algorithms during the total process of searching.

Figure 8b (F2) indicates the optimization results for the F2 *Sphere* function. Besides, MSEO emerges as best among all the other algorithms during all processes of optimization.

Figure 8c (F3) illustrates the optimization results for the F3 *Sum Squares* function. Obviously, MSEO and SEO have the same convergence rate during the total optimization process except in the first iteration of 25 iterations in this unimodal function. Eventually, GA, ABC, and FA find the global minimum, respectively.

Figure 8d (F4) displays the optimization outcomes for the F4 *Sum Squares* function. Noticeably, MSEO and SEO

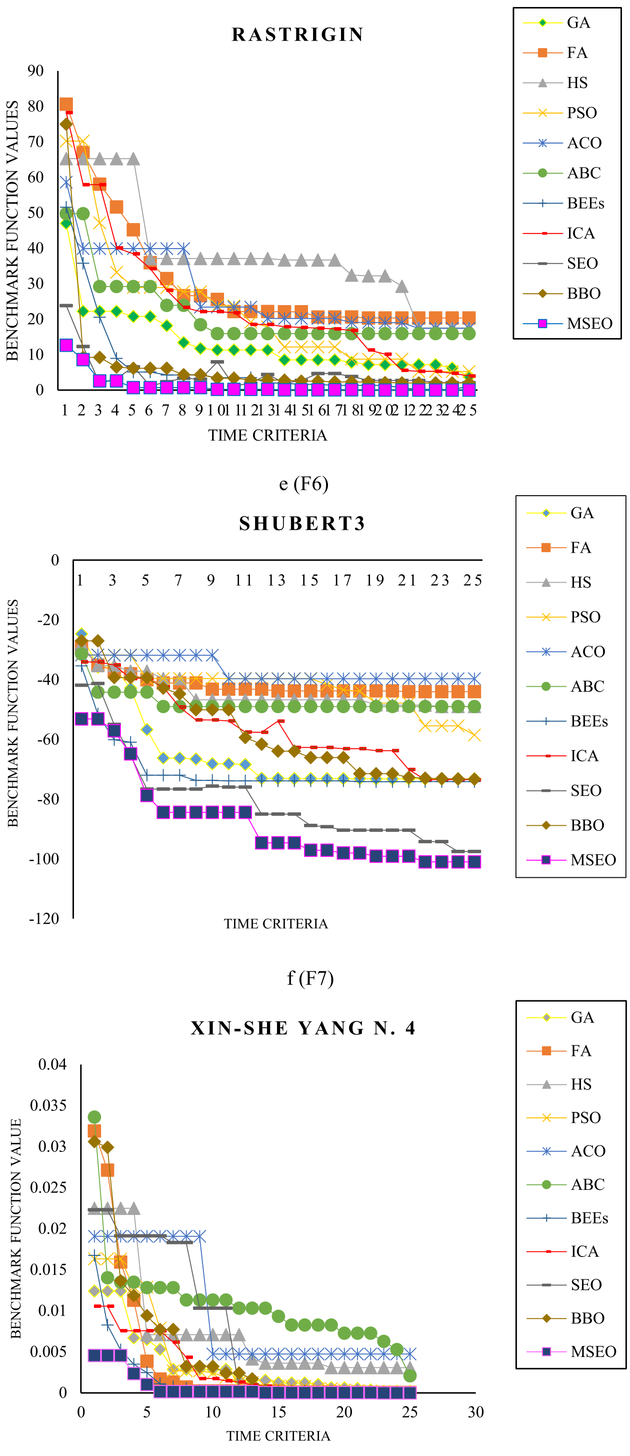


Fig. 9 Comparison of the convergence of the various algorithms based on multimodal functions (F6–F11)

have the same convergence rate during the total optimization process in this unimodal function. Generally, the MSEO algorithm is more robust than other metaheuristic algorithms at the 25 iterations.

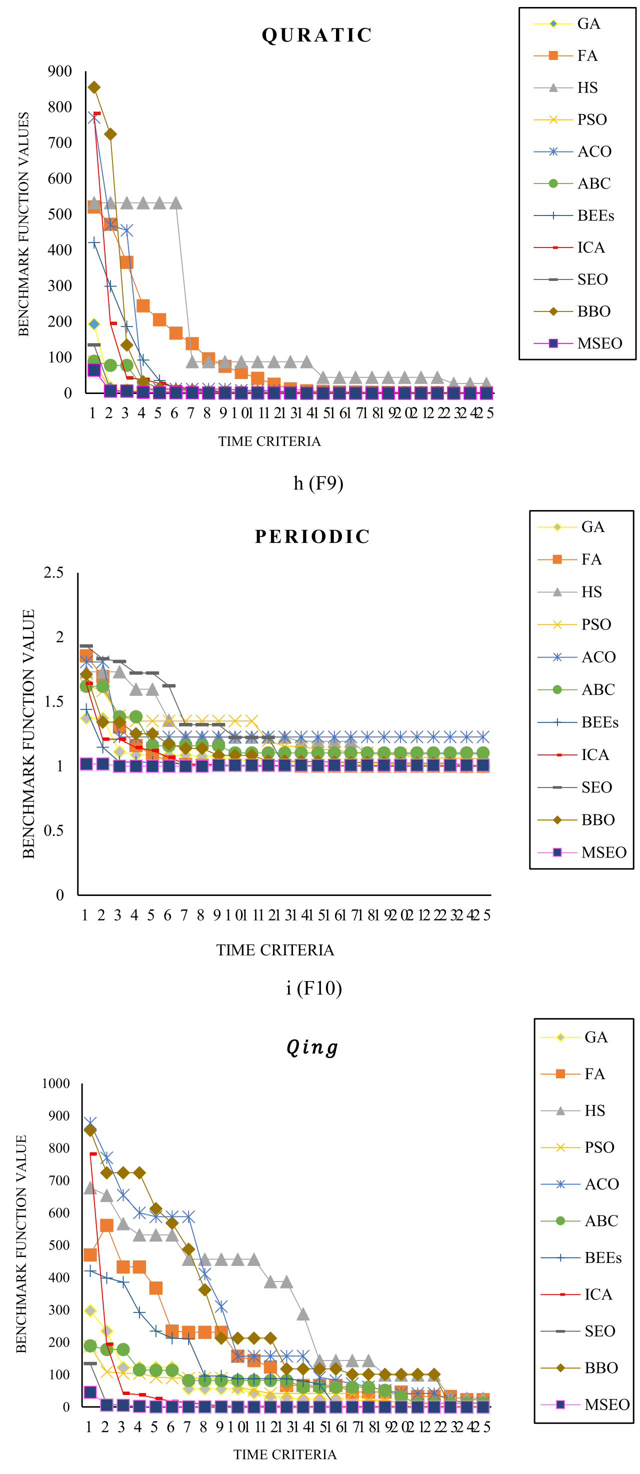
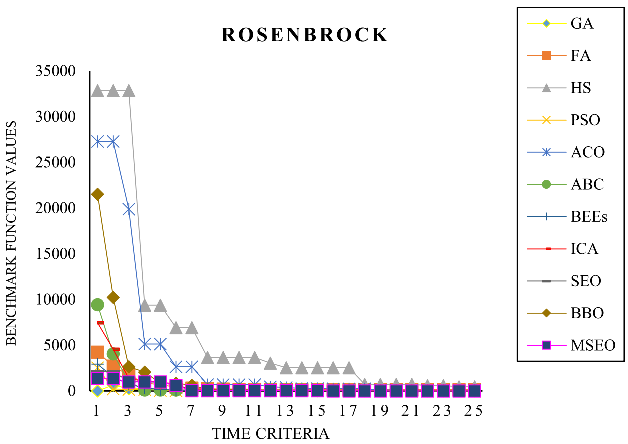
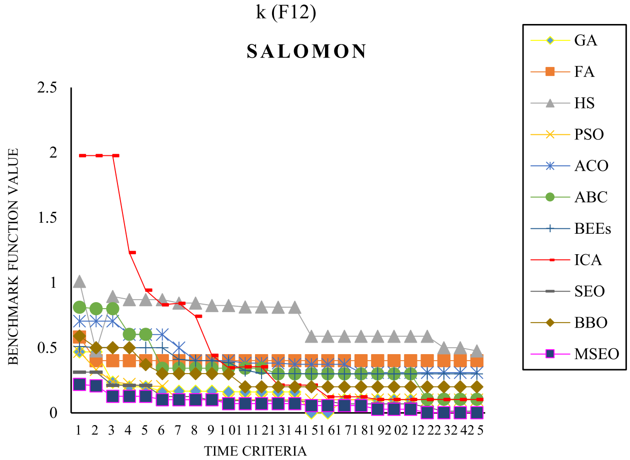


Fig. 9 continued

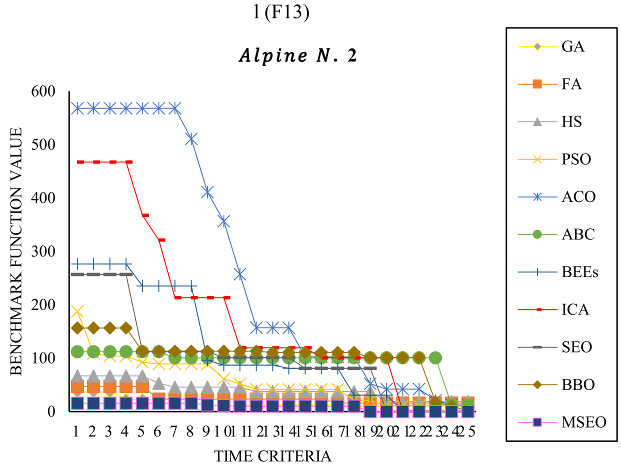
Figure 8q (F5) demonstrates the optimization results for the 'F5 *Rotated Discus Function*' function. Hence, it is clear that MSEO and SEO have the same convergence rate during the total optimization process in this unimodal function. All in all, the MSEO algorithm performs better



k (F12)



l (F13)

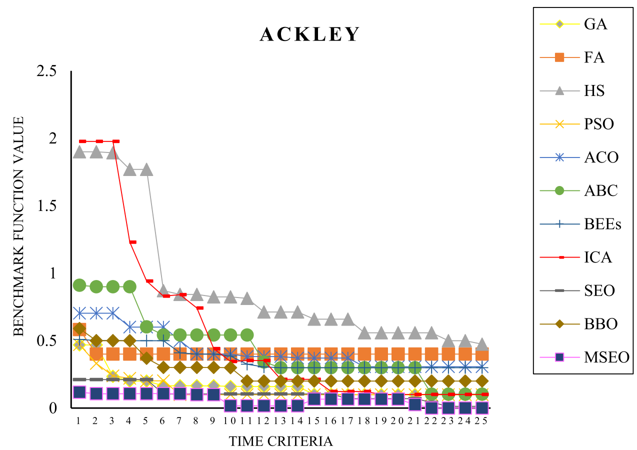


m (F14)

Fig. 10 Comparison of the convergence of the various algorithms based on n-dimension multimodal functions (F12-F15)

than other proposed algorithms at the 25 iterations. Additionally, BBO has the worst efficiency compared to other presented algorithms.

Figure 9e (F6) indicates the optimization results for the F6 *Schwefel*22.2 function. This figure indicates that there is



n (F15)

Fig. 10 continued

a little variance between the performance of FA, HS, PSO, ICA, and BBO for this benchmarking function. As a conclusion of this figure, MSEO is better than basic FA during the optimization process in this multimodal benchmarking function at the 25 iterations.

Figure 9f (F7) indicates the performance obtained for the F7 *Shubert3* function. For this multimodal function, MSEO is the best than the original SEO and the other proposed algorithms. Here, SEO and MSEO illustrate that the convergence rate of these algorithms is lower than other algorithms during all iteration processes. The convergence rate of algorithms is lower than other algorithms. But, the speed of convergence of ICA, FA, PSO, ACO, and HS is almost the same during all optimization process.

Figure 9g (F8) indicates the optimization results obtained for the eleven algorithms in the F8 *Xin-She Yang N. 4* function. Clearly, the convergence rate of all algorithms has been decreasing in all iterations processes, and the MSEO algorithm has a better convergence rate

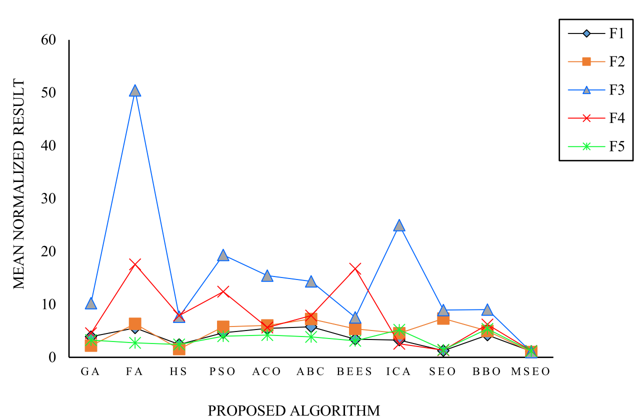


Fig. 11 The mean normalized results of F1–F5 benchmark functions

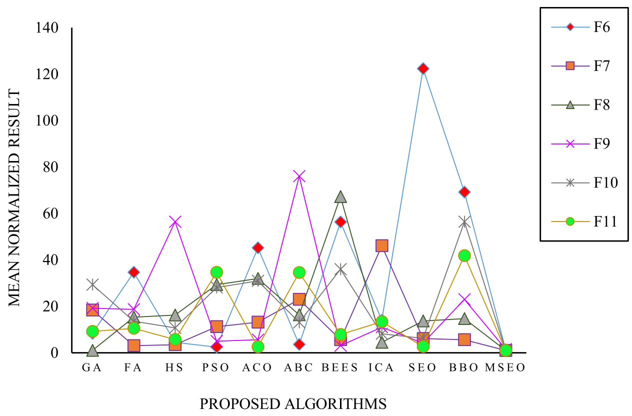


Fig. 12 The mean normalized results of F6–F11 benchmark functions

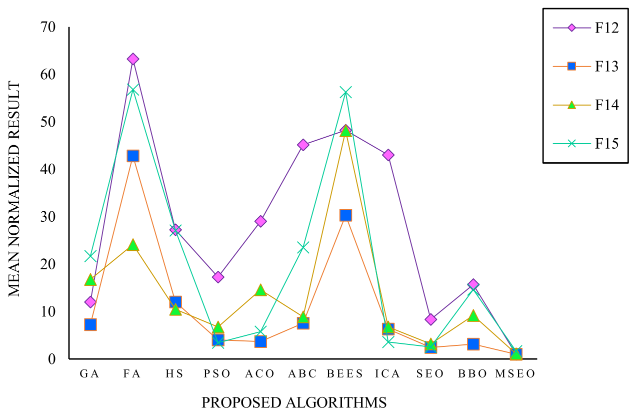


Fig. 13 The mean normalized results of F12–F15 benchmark functions

performance than other algorithms. At last, HS and ACO convergence rate is almost the same value.

Figure 9h (F9) indicates the optimization results for the F9 *Quadratic* function . Obviously, MSEO has the fastest convergence rate for finding the global minimum. BBO ACO, HS, FA, and ICA are not close in the maximum number of iterations at this function, which the process is as follows: first, the convergence rate is strongly descending at the first iterations, and second, the convergence rate is slowly declining at the last iterations. Eventually, MSEO was significantly the best of all other proposed algorithms during all processes.

Figure 9i (F10) shows the optimization results for the F10 *Periodic* function. This figure shows that MSEO is slightly superior to original SEO during the optimization process in the multimodal benchmarking function. Also, the convergence rate of the MSEO algorithm is better than other algorithms. On the other hand, the convergence rate of the ICA, ABC, and PSO algorithms is almost close and the same. It is clear that MSEO outperforms better than the other algorithms during the process of iterations.

Figure 9j (F11) demonstrates the optimization results for the F11 *Qing* function. This figure shows that MSEO is superior to the main SEO during the optimization process in the multimodal benchmarking function. Furthermore, the convergence rate of the MSEO algorithm is more powerful than other algorithms. On the other hand, the convergence rate of the BBO, ACO, HS, and FA algorithms is almost close and the same. It is manifested that MSEO outperforms better than the other algorithms during the process of iterations. As a result, MSEO is more reliable than other proposed algorithms to find optimal solutions.

Figure 10k (F12) indicates the optimization results for the F12 *Rosenbrock* function. From this Figure, we could see that MSEO is slightly superior to the original SEO during the optimization process in this n-dimension multimodal benchmarking function. Clearly, MSEO and SEO outperform all the other proposed algorithms in this benchmarking function. In addition, the convergence rate of GA, FA, BEEs, ICA, SEO, ABC, PSO, and MSEO algorithms is almost the same during all iterations. But, the convergence rate of HS, ACO, and BBO algorithms is sharply declining during the initial iterations and the convergence rate is fixed during the last iterations. Eventually, the MSEO algorithm is better than the other algorithms in terms of convergence rate during the optimization process in this benchmark function.

Figure 10l (F13) indicates the optimization results for the F13 *Salomon* function. In this Figure, MSEO is the slightest superior to the SEO during the optimization process in this n-dimension multimodal benchmarking function. The convergence rate of all algorithms is almost the same except for the ICA. Eventually, the MSEO finds the global minimum during all iterations.

Figure 10m (F14) illustrates the optimization outcomes for the F14 $AlpineN.2$ function. In this Figure, the MSEO algorithm is more robust and the best compared to the other presented algorithms. Also, the ACO is very weak as opposed to the other suggested algorithms, while the convergence of the ABC, PSO, HS, FA and SEO algorithms is very close.

Figure 10n (F15) indicates the optimization results for the F15 *Ackley* function. The behavior of algorithms is the same except for in this Figure. However, the MSEO algorithm shows that it is more efficient during 25 iterations.

As such, Figs. 11, 12 and 13 display the mean normalized results of the proposed algorithms. From Figs. 11, 12 and 13, it is clear that the MSEO algorithm is better than the other proposed algorithms at finding objective function minimum on the ten test benchmark functions. Besides, SEO is the second most effective, performing best on the

Table 8 The outcomes of methods in benchmark during 25 run times (B = best, W = worst, M = mean, SD = standard deviation, D = dimension, R = rank)

Function	D		GA	FA	HS	PSO	ACO	ABC	BEEs	ICA	SEO	BBO	MSEO
F1	25	W	0.22483	0.39711	0.68706	0.33521	0.45136	0.12772	0.16381	0.9564	0.10343	0.04503	0.02783
		M	0.10156	0.10822	0.34516	0.10534	0.32400	0.12772	0.03138	0.14736	0.046327	0.12172	0.006592
		B	0.0565	0.0716	0.0838	0.0285	0.29407	0.12772	0.0106	0.00116	0.002935	0.04503	0.000251
		SD	0.05253	0.08741	0.17557	0.10636	0.06117	2.83279E−17	0.04167	0.21778	0.041621	0.10741	0.010703
		R	4	6	11	5	10	8	2	9	3	7	**1**
F2	25	W	6.6405	21.44	23.9754	21.5401	35.9578	12.464	21.5401	19.9432	3.8124	10.9138	2.6783
		M	0.7589	5.17179	8.13069	1.48597	5.2235	1.80563	1.54597	3.09479	3.858016	2.43153	0.485005
		B	0.0309	0.00273	4.8931	0.00307	0.03077	0.02331	0.00307	0.00716	8.1262	0.00289	0.000342
		SD	1.59510	6.45081	6.35481	4.72884	9.74322	3.41128	4.72884	5.39911	2.580504	3.66356	0.698964
		R	2	9	11	3	10	5	4	7	8	6	**1**
F3	25	W	7.3588	36.2939	64.4675	57.7401	85.7818	28.3936	102.206	55.0187	5.35678	72.0458	2.67833
		M	1.14623	11.2369	16.3565	4.59028	9.88651	3.03060	10.9026	6.04128	0.94958179	9.69499	0.424471
		B	0.00507	0.03156	6.0506	0.01483	0.05698	0.04650	0.00628	0.01465	0.00003489	0.00844	0.000141
		SD	2.08211	11.6359	18.8296	13.1379	17.6495	5.67990	26.0289	13.0551	1.65995342	18.2995	0.685968
		R	3	10	11	5	8	4	9	6	2	7	**1**
F4	25	W	0.21241	0.25691	0.42239	0.51843	0.63146	0.12319	0.01676	0.25655	0.126714	0.03061	0.004543
		M	0.30457	0.60384	0.37760	0.20381	0.23488	0.21043	0.26929	0.35286	0.2567859	0.00513	0.0237323
		B	0.00517	0.12559	0.12326	0.14051	0.03173	0.02245	0.23441	0.56720	0.670211	0.12331	0.0034632
		SD	0.13459	0.03862	0.16684	0.23580	0.05312	0.16553	0.09372	0.23173	0.141516	0.15854	0.0215451
		R	5	3	9	11	2	8	4	10	6	7	**1**
F5	25	W	0.4682	17.124	5.1872	6.10344	0.71238	2.81118	18.2341	1.97664	0.31248	13.5891	0.12302
		M	0.17093	16.561	4.9134	5.15671	0.42851	2.36492	17.9804	0.54946	0.12313	11.2855	0.08123
		B	0.09988	0.01279	0.05643	0.02788	0.03075	0.01036	0.07865	0.10222	0.02782	0.23411	0.000127
		SD	0.10846	0.23037	0.11128	0.16965	0.19853	0.18068	0.24781	0.62398	0.079823	0.14993	0.042981
		R	3	10	4	6	8	7	11	9	2	5	**1**
F6	25	W	47.044	80.6069	65.2467	70.2204	58.6524	49.772	51.5811	78.267	23.8124	75.0101	12.6783
		M	13.2843	30.5370	38.6471	22.0893	27.1389	21.5183	6.60728	23.8042	4.338016	6.65684	1.245005
		B	3.8782	20.3127	17.514	5.2534	17.4363	15.9548	1.1462	3.9756	2.1262	2.0033	0.00034
		SD	9.06943	16.7419	15.2297	18.0933	11.2999	9.89273	12.1051	18.5859	4.715309	14.4077	2.976578
		R	5	10	11	8	9	6	3	7	2	4	**1**
F7	25	W	− 24.723	− 28.831	− 28.060	− 31.761	− 31.737	− 31.440	− 35.363	− 34.091	− 41.8124	− 27.022	− 53.1458
		M	− 64.946	− 41.797	− 44.300	− 42.423	− 36.880	− 47.525	− 70.203	− 56.362	− 80.1935	− 57.449	− 87.5408
		B	− 73.218	− 44.021	− 49.023	− 58.547	− 39.732	− 49.003	− 74.124	− 73.462	− 97.5126	− 73.277	− 101.001
		SD	14.2171	3.63729	5.67600	7.62995	3.88131	3.80698	9.17394	12.8511	15.39355	15.1525	15.30392
		R	4	10	8	9	11	7	3	5	2	6	**1**

Table 8 (continued)

Function	D		GA	FA	HS	PSO	ACO	ABC	BEEs	ICA	SEO	BBO	MSEO
F8	25	W	0.01241	0.03191	0.02249	0.01633	0.01906	0.03359	0.01676	0.01055	0.012314	0.03061	0.004556
		M	0.00327	0.00384	0.00761	0.00411	0.00988	0.01066	0.00159	0.00286	0.009359	0.00513	0.0007304
		B	0.00017	5.75E−05	0.00306	8.6E−05	0.00473	0.00206	1.08E−05	1.86E−06	1.12E−05	3.35E−05	0.0000032
		SD	0.00389	0.00862	0.00684	0.00580	0.00702	0.00553	0.00372	0.00363	0.011516	0.00854	0.0015223
		R	4	5	8	6	10	11	2	3	9	7	**1**
F9	25	W	192.990	520.425	532.004	87.7845	770.224	89.4591	420.736	782.369	134.8124	855.437	64.6783
		M	9.93390	97.9624	173.051	5.41325	72.4739	12.3063	43.2415	46.2033	6.735537	72.1818	3.38158
		B	0.58081	0.89379	27.2352	1.1432	0.91713	1.8525	0.87245	1.1105	0.017365	1.168	0.000122
		SD	38.2965	152.400	207.032	17.2389	192.739	26.3516	104.980	158.293	26.75777	218.465	12.86541
		R	4	10	11	2	8	5	6	7	3	9	**1**
F10	25	W	1.3712	1.8563	1.7314	1.6294	1.8087	1.6193	1.4409	1.6416	1.1124	1.7128	1.006231
		M	1.06858	1.08884	1.27435	1.23603	1.27307	1.17844	1.03009	1.05930	1.066016	1.11892	1.015763
		B	1.0097	1.0012	1.0962	1.0548	1.2265	1.10434	1.00311	1.00013	1.001755	1.0038	1.004345
		SD	0.09634	0.21862	0.21786	0.16711	0.16120	0.15310	0.09072	0.13735	0.181685	0.15988	0.026148
		R	5	6	11	9	10	8	2	3	4	7	**1**
F11	25	W	162.231	470.421	367.124	92.7818	856.224	231.421	378.721	877.321	254.8674	931.317	78.6124
		M	154.439	121.934	181.211	32.4221	215.132	65.3356	103.262	112.793	29.73247	178.183	16.32458
		B	0.34681	0.56881	19.2282	2.1766	0.82331	1.8785	2.08765	1.0455	0.012685	1.6321	0.006743
		SD	45.4332	231.215	167.211	19.2345	245.677	34.3236	78.233	133.873	18.75247	212.455	18.83211
		R	5	10	8	3	11	4	6	7	2	9	**1**
F112	25	W	624.008	4254.07	32,846.7	2521.8	27,284.0	9435.52	2931.68	7455.94	2134.334	21,516.2	1357.678
		M	88.4305	558.601	6649.69	128.067	3804.98	639.170	235.674	666.597	322.8058	1579.39	253.012
		B	3.1196	156.501	451.835	2.2669	32.6642	36.2161	1.1472	0.00716	1.029348	3.3772	0.000342
		SD	136.327	970.247	10,206.8	501.745	8143.1	2004.53	649.323	1703.98	637.8187	4647.17	469.1506
		R	**1**	6	11	2	10	7	3	8	5	9	4
F13	25	W	0.4682	0.58178	1.0097	0.47006	0.7038	0.81118	0.50938	1.97664	0.3124	0.5891	0.21783
		M	0.17093	0.40717	0.71032	0.14263	0.42851	0.36492	0.36551	0.54946	0.11139	0.27858	0.074728
		B	0.09988	0.39987	0.47471	0.09988	0.30785	0.1036	0.29987	0.10222	0.01262	0.19993	0.000342
		SD	0.10846	0.03637	0.16084	0.09135	0.13853	0.20468	0.08606	0.62398	0.080995	0.11993	0.057608
		R	3	6	10	4	8	9	5	11	2	7	**1**
F14	25	W	10.3421	42.3449	72.4877	69.7421	101.458	67.2331	156.233	78.0234	9.27878	96.2341	5.54323
		M	1.65341	7.3211	8.3788	5.23312	11.8352	4.45330	14.3045	17.8128	0.565813	13.6799	0.345472
		B	0.04561	0.45156	0.6706	0.45883	0.82398	0.81632	0.45213	0.67426	0.005681	0.68044	0.003451
		SD	3.56112	12.4559	16.8577	7.1567	16.6345	7.63210	23.7829	15.3251	2.3445342	21.2385	0.365918
		R	3	6	9	4	8	5	11	7	2	10	**1**

Table 8 (continued)

Function	D		GA	FA	HS	PSO	ACO	ABC	BEEs	ICA	SEO	BBO	MSEO
F15	25	W	126.321	156.789	212.457	98.7421	276.421	176.561	321.025	138.531	56.78809	278.251	26.50212
		M	23.6211	19.2891	34.4531	15.3412	35.6512	13.2173	56.0988	16.3211	4.450912	29.5671	2.32086
		B	2.12451	7.42316	14.8701	11.4533	10.4568	26.3231	21.4543	12.4502	0.560981	17.6831	0.23011
		SD	6.34450	15.7651	21.3201	6.1217	4.2137	7.63210	18.781	5.4390	3.56332	16.2321	1.05672
		R	6	8	11	5	4	7	10	3	2	9	**1**
Avg. of rank			3.8	7.67	9.6	5.47	8.47	6.73	5.4	6.2	3.6	7.3	**1.2**

other ten of the ten benchmarks when multiple runs are made.

In this regard, first of all, the metaheuristic algorithms have been tuned in this comparison. Hence, the methods have been run for 25 iterations and the best (B), the worst (W), the average (M), and also the standard deviation (SD) are mentioned and shown in Table 8. Therefore, the dimension ($D = 25$) for each benchmarked function is considered. This comparison is according to the equal number of fitness evaluations. All comparative algorithms have been run for a maximum of 25 iterations (i.e. GA, FA, HS, ABC, PSO, ICA, ACO, BBO, BEEs, SEO, and MSEO). From Table 8, it is clear that the improved version of SEO has better ranks than the other methods. In all of the test problems, MSEO algorithm indicates the best value. Eventually, the MSEO is the best in the total ranking by 1.2 on average of rank as indicated in Table 8.

Moreover, to highlight the performance of the algorithms, statistical analyses have been done. The means plots and Least Significant Difference (LSD) for all methods for benchmark functions (F1–F15), has been provided as seen in Fig. 14. All statistical results prove that our MSEO not only performs better than the original version but is also stronger and more robust than the other algorithms. Additionally, the Standard Deviation (SD) for the proposed algorithms for benchmark functions (F1–F15) is provided in Fig. 15 which shows that MSEO algorithm has a better performance in all benchmark functions except for the F7 algorithm. Our findings show that the MSEO is more capable of solving the problem, takes less running time, and has better convergence than other algorithms.

4.1 The parameter sensitivity test for MSEO

In this section, sensitivity analyses have been performed on the parameters of the MSEO to investigate the behavior of the presented 15 benchmark function models. The MSEO algorithm is recognized as the most robust and the most efficient metaheuristic in this study. A set of changes including the rate of collecting data, rate of the connecting attacker, number of connections, the weight of defender, and weight of attacker for proposed 15 benchmark function models are analyzed. The analysis is divided into five instances, namely, I1–I5 as shown in Table 9. Furthermore, all outcomes mean for 25 iterations are indicated in Table 10 and Fig. 16.

According to Table 10 and Fig. 16, it can be concluded that the benchmark functions (F1–F15) of the MSEO algorithm increase by increasing the amount of these parameters.

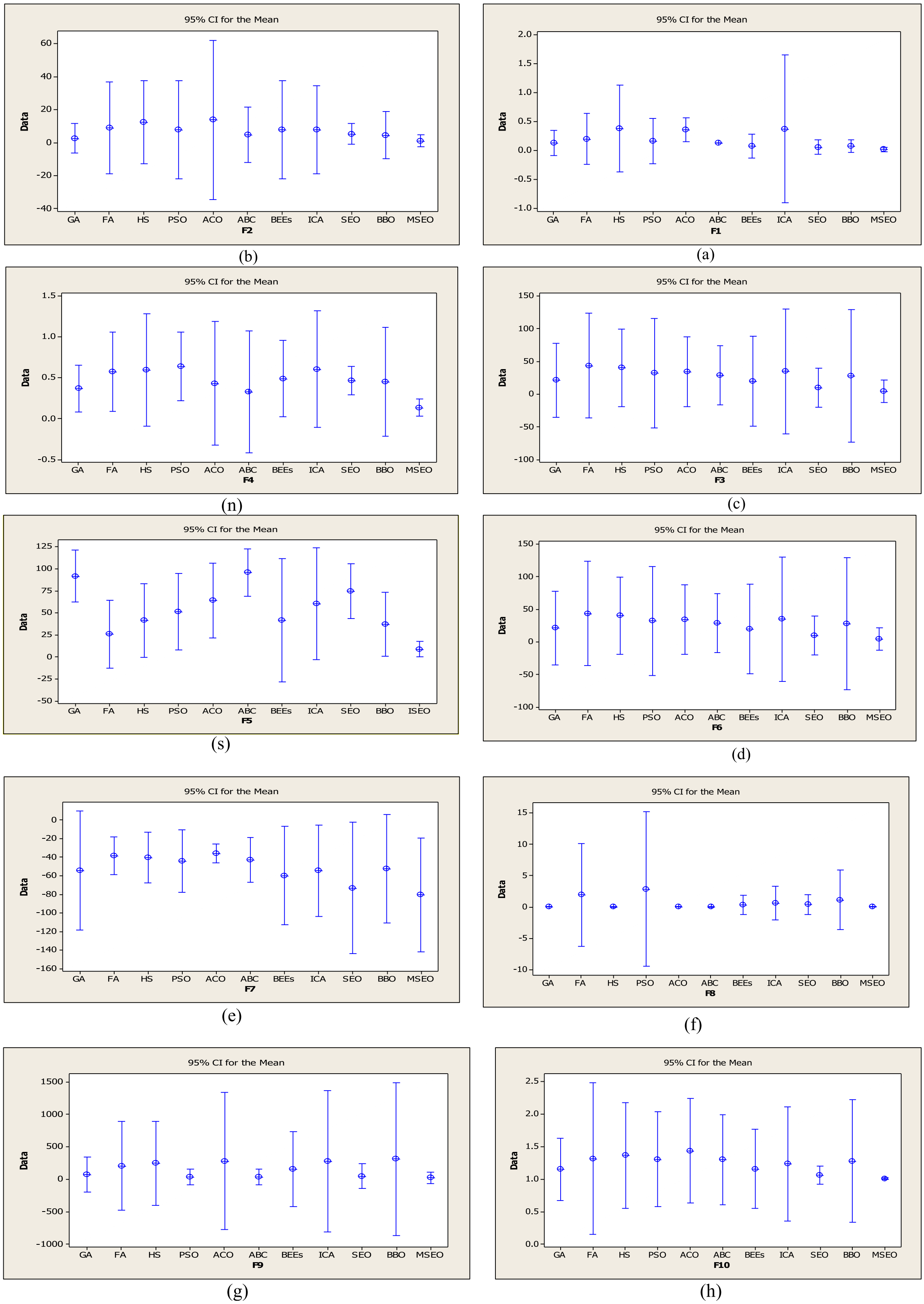


Fig. 14 The Means plot and LSD intervals for the proposed methods in equal number of fitness evaluations for benchmark functions (F1–F15)

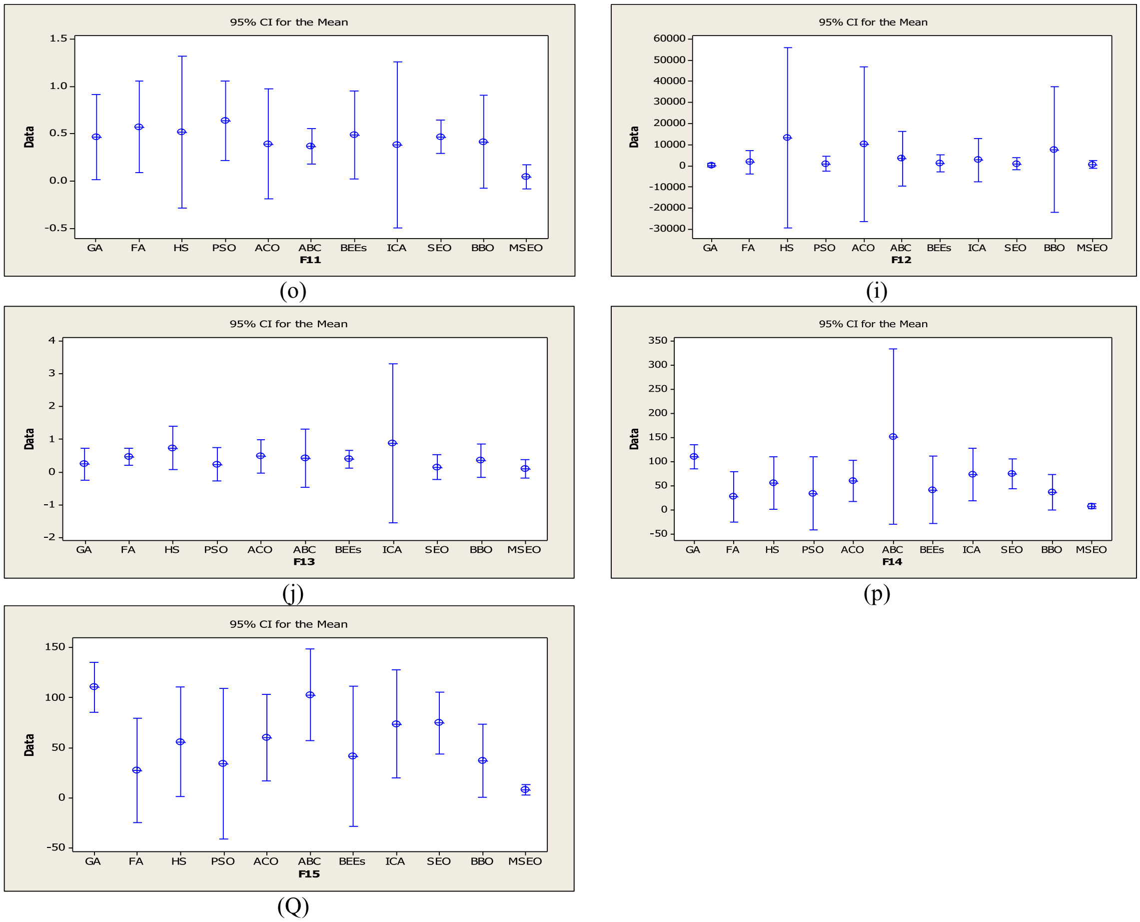


Fig. 14 continued

5 Engineering applications

In this section, four engineering application instances are presented to prove the proper performance of the proposed algorithm. The first case is taken from Ghasemi et al. (2019) who presented a mathematical model according to the location and allocation problems of the shelters for earthquake evacuation planning. The second case is taken from Goodarzian et al. (2020) who designed a new pharmaceutical supply chain network under uncertainty. The next case is from Fathollahi-Fard et al. (2019) who formulated a scheduling problem of trucks in a cross-docking system. Finally, the last example is taken from Goli et al. (2019) who considered a production planning problem under uncertainty. Before introducing the cases, the assessment metrics are as follows:

5.1 Assessment metrics

- *Mean ideal distance (MID)* (Goodarzian et al. 2021d; Goodarzian et al. 2021e): the goal of the MID is the distance between the Pareto optimal solutions. This metric is formulated based on Eq. (17).

$$MID = \frac{\sum_i^n \sqrt{\left(\frac{f_{1i}-f_1^{best}}{f_{1,total}^{max}-f_{1,total}^{min}}\right)^2+\left(\frac{f_{2i}-f_2^{best}}{f_{2,total}^{max}-f_{2,total}^{min}}\right)^2}}{n} \quad (17)$$

where f_{ji} indicates the value of jth objective for the ith solution in Pareto frontier and $f_{j,total}^{min}$ and $f_{j,total}^{max}$ illustrate the minimum and maximum amounts of the ith objective between solutions in the Pareto frontier. In addition, n represents the number of Pareto solutions. Low values of this metric indicate high performance and quality.

- *Spacing metric (SM)* (Goodarzian et al. 2020): the SM demonstrates the uniformity of the spread of the non-dominated set of solutions. The SM metric is computed according to Eq. (18).

$$SM = \frac{\sum_{i=1}^{n-1}\left|d_i - \bar{d}\right|}{(n-1)\bar{d}} \quad (18)$$

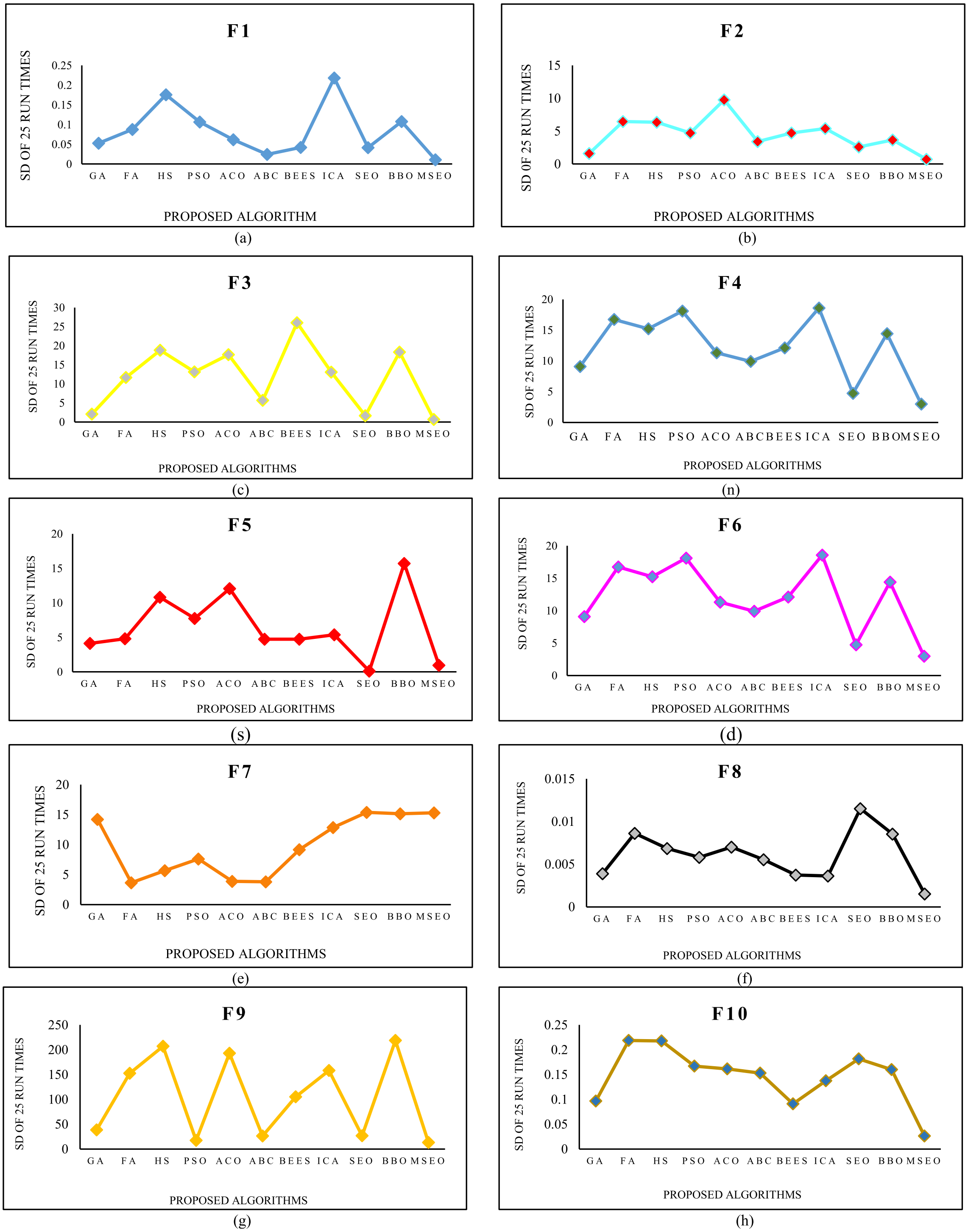


Fig. 15 The SD for the proposed methods in an equal number of fitness evaluations for benchmark functions (F1–F15)

where $\overline{d}$ indicates the average Euclidean distance and d_i is the Euclidean distance between two adjacent Pareto solutions. Lower values of SM indicate higher efficiency. Hence, when SM is close to zero, the distance among all the adjacent solutions will be equal.

– *Spread of non-dominance solution (SNS)* (Goodarzian et al. 2020): the higher value of these metrics brings the better performance. This metric is formulated by the following equation:

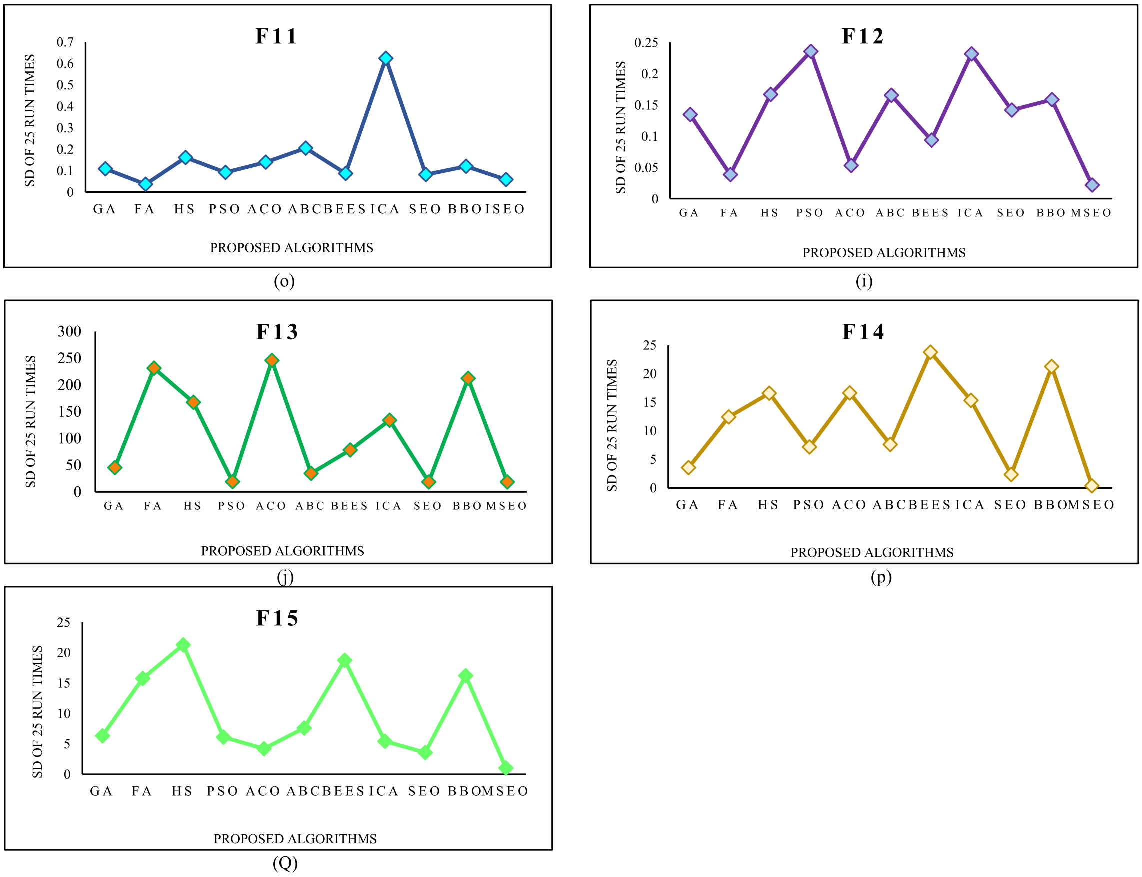


Fig. 15 continued

Table 9 The sensitivity analysis on the parameters of MSEO

Parameters	I1	I2	I3	I4	I5
Rate of collecting data	0.2	0.3	0.4	0.5	0.6
Rate of connecting attacker	0.08	0.15	0.2	0.25	0.3
Number of connections	50	60	70	80	90
Weight of defender	45	55	65	75	85
Weight of attacker	65	75	85	95	100

$$SNS = \sqrt{\frac{\sum_{i=1}^{NPS}(MID - \sum_{j=1}^{\rho_{obj}} f_i^j)^2}{NPS - 1}} \tag{19}$$

It should be noted that NPS is the number of Pareto solutions for the algorithm. f_i^j is the ith solution and jth objective function and the ρ_{obj} is the number of the objective function.

Table 10 The results of the benchmark functions of the MSEO algorithm

Instances	F1	F2	F3	F4	F5	F6	F7	F8
I1	0.006592	0.485005	0.424471	0.0237323	1.34214	1.245005	− 87.5408	0.0007304
I2	0.01244	0.96771	0.56032	0.0765662	2.45617	2.54412	− 68.76	0.005662
I3	0.34512	1.56321	0.96543	0.234065	2.98071	4.63211	− 54.12	0.078943
I4	1.56091	2.34098	1.45076	0.791203	3.40912	6.67912	− 47.54	0.560912
I5	2.7812	4.45712	3.78923	2.543219	7.29817	9.40981	− 31.15	1.542301

Instances	F9	F10	F11	F12	F13	F14	F15
I1	3.38158	1.015763	16.32458	253.012	0.074728	0.345472	2.32086
I2	5.67731	2.34401	18.5778	344.2331	0.23042	0.788091	3.577021
I3	7.04512	4.21309	23.1209	344.2331	1.43891	1.054109	5.345501
I4	9.45326	6.45098	25.54309	408.45671	3.43891	4.64312	7.892341
I5	12.87023	8.78923	37.5678	510.5431	7.76541	8.89212	11.678031

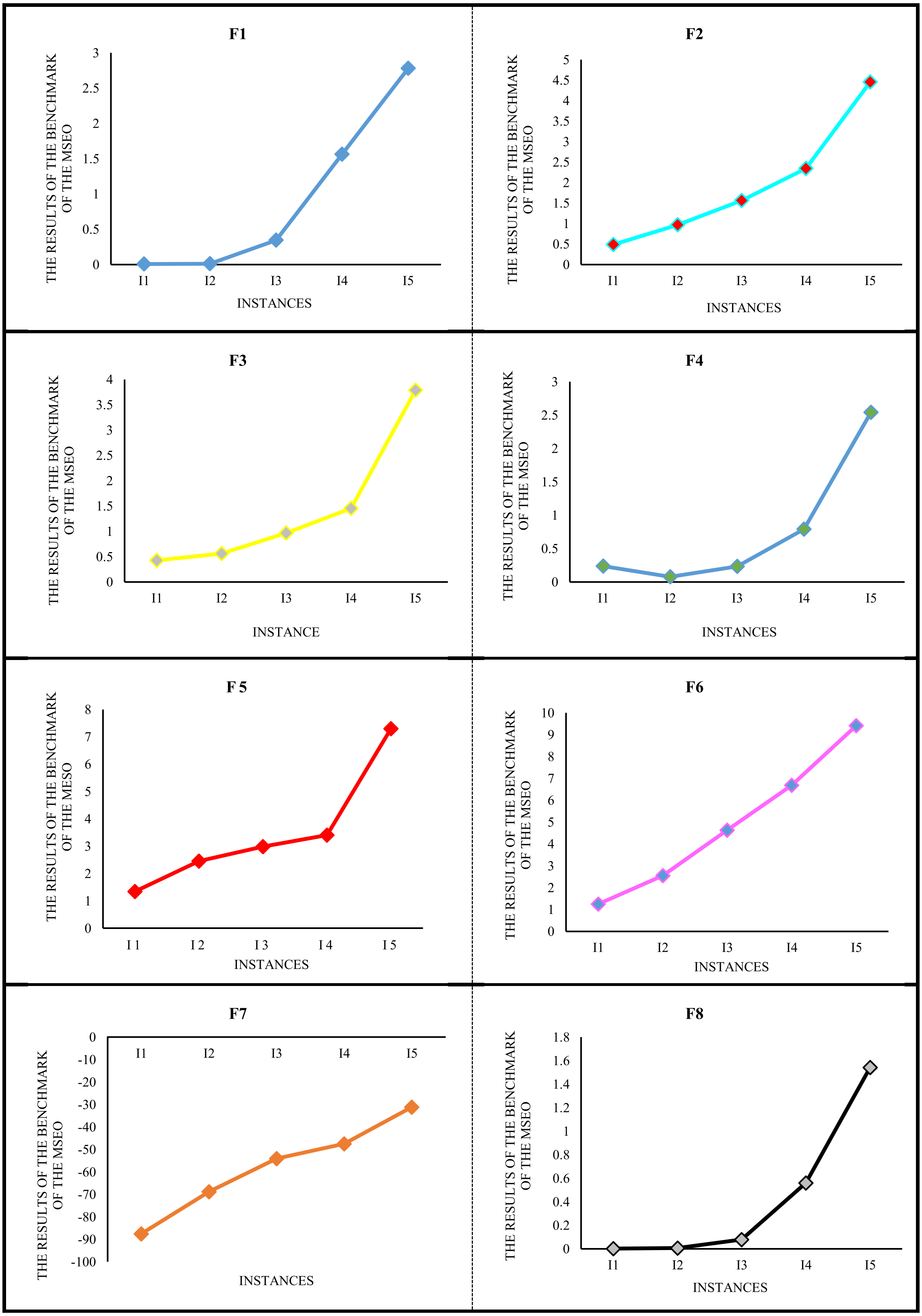


Fig. 16 The behavior outcomes of the MSEO algorithm in benchmark functions (F1–F15)

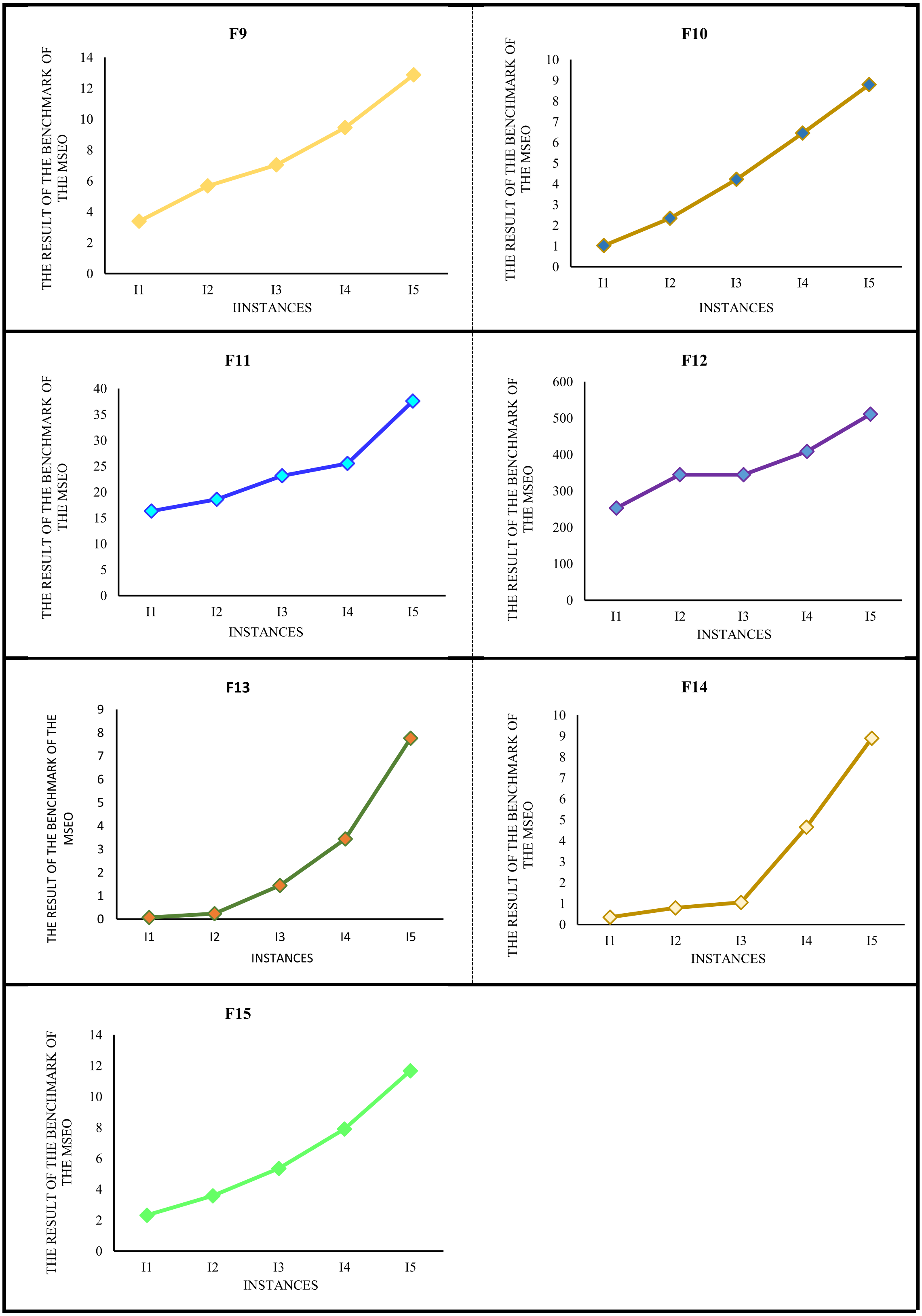


Fig. 16 continued

Table 11 Metrics obtained for each algorithm (Case1)

No.	NSGA-II			MMOPSO			MSEO		
	MID	SM	Time (s)	MID	SM	Time (s)	MID	SM	Time (s)
1	2.259	0.192	3	2.257	0.188	3	2.255	0.187	3
2	2.392	0.198	5	2.349	0.192	5	2.345	0.190	5
3	2.672	0.225	7	2.610	0.219	6	2.606	0.217	6
4	2.989	0.238	15	2.933	0.232	9	2.929	0.231	8
5	3.426	0.250	19	3.386	0.245	12	3.382	0.245	11
6	4.933	0.420	25	4.875	0.416	16	4.825	0.415	14
7	4.995	0.450	31	4.903	0.443	21	4.891	0.440	18
8	5.326	0.467	40	5.100	0.450	25	5.080	0.447	23
9	5.521	0.486	66	5.289	0.479	32	5.246	0.475	28
10	5.615	0.528	79	5.511	0.518	38	5.459	0.515	35
Ave	4.012	0.345	29	3.921	0.338	16.7	3.901	0.336	15.1

- *The relative percentage deviation (RPD)* (Goodarzian et al. 2020, 2021b; e): the RPD is shown as follows:

$$RPD = \frac{|Alg_{sol} - Min_{sol}|}{Min_{sol}} \quad (20)$$

  where Alg_{sol} represents the value of objective in individual trials and also Min_{sol} shows the best solution among all trials.
- Number of partial solutions (NPS) (Goodarzian et al. 2021b): NPS shows the number of solutions in Pareto front for each algorithm and the bigger NPS is more appropriate.

5.2 Case 1: location-allocation problem for earthquake evacuation planning

Ghasemi et al. (2019) proposed a multi-objective, multi-echelon, multi-commodity, and multi-period model for earthquake evacuation planning. Their main goals have been to minimize the cost of location and allocation of facilities to distribution centers and to minimize the shortage of relief commodities. Their proposed model is solved by utilizing modified multiple-objective particle swarm optimization (MMOPSO) and Non-Dominated Sorting Genetic Algorithm-II (NSGA-II) algorithms. The case study was considered in region 1 of Tehran/Iran. Then, to assess the efficiency of their suggested model, two assessment metrics, MID (Mean Ideal Distance) and SM (Spacing metric) have been used.

Table 11 illustrates the outcomes comparing the NSGA-II and MMOPSO methods with the proposed MSEO algorithm. As can be seen, the results of MID and SM metrics are reported for 10 Pareto points. The average MID metric for NSGA-II, MMOPSO, and MSEO algorithms are 4.012, 3.921, and 3.901, respectively, which shows the superiority of the MSEO over the other two algorithms. Also, the average SM metric for NSGA-II, MMOPSO, and MSEO algorithms are 0.345, 0.338, and 0.336, respectively, which shows more efficiency of the MSEO method than the other two methods. Therefore, the computational (CPU) time of the MSEO with an average of 15.1 s is better than the other two algorithms.

5.3 Case 2: pharmaceutical supply chain network design

Goodarzian et al. (2020) designed a multi-objective, multi-period, and multi-product mathematical model for the pharmaceutical supply chain network. A production–distribution-purchasing-ordering-inventory holding-allocation-routing is considered under uncertainty. Their main aim has been to minimize supply chain costs, minimize pharmaceutical delivery time, and maximize route reliability. To solve their proposed model, multi-objective social engineering optimization (MOSEO), multi-objective simulated annealing (MOSA), multi-objective Keshtel algorithm (MOKA), and multi-objective firefly algorithm (MOFFA) algorithms have been utilized. MID (Mean Ideal Distance) and (SNS) spread of non-dominance solution metrics were used to prove the performance of the meta-heuristic algorithms.

Moreover, a comparison of the efficiency assessment metrics of the proposed methods is shown in Table 12. The considered problem for 10 Pareto points is solved. The first five was related to small-scale problems and the second five were relevant to large-scale problems. As it is known, the CPU time to solve the model with the MSEO algorithm for all cases was less than other algorithms. Likewise, the MSEO is better than other methods in all cases in terms of MID and SNS metrics. Therefore, it can be said that, in general, the MSEO algorithm has performed better than MOSEO, MOSA, MOKA, and MOFFA algorithms.

Table 12 Metrics obtained for each algorithm (Case 2)

No.	MOSEO			MOSA			MOKA			MOFFA			MSEO		
	MID	SNS	Time (s)	MID	SNS	Time (s)	MID	SNS	Time (s)	MID	SNS	Time (s)	MID	SNS	Time (s)
1	910.2	828,520.4	18.3	1235.1	1,236,566	25.2	917.3	841,554.7	22.4	881.8	777,587.5	15.6	875.3	777,432.4	15.5
2	693.1	480,500.3	25.1	1581.9	144,671.8	38.5	850.4	723,212.9	29.3	837.2	700,957.7	19.4	686.1	140,214.6	17.2
3	784.7	615,758.1	31.1	1879.8	167,433.2	48.8	629.1	395,844.5	38.3	732.3	536,283.8	28.1	612.6	165,507.9	25.3
4	826.2	682,759.9	67.3	2153.0	1,902,322	86.7	840.2	536,454.2	61.5	749.3	561,513.7	50.4	735.4	531,438.5	46.9
5	902.5	814,553.8	94.2	2489.2	2,256,635	103.2	840.2	705,946.6	86.3	866.8	751,509.4	80.1	812.9	700,981.4	71.3
6	1724.5	2,974,066	877.2	2623.7	5,445,627	1146.4	1408.0	1,982,711	843.0	1985.9	3,944,102	653.2	1401.7	1,893,685	624.0
7	2104.2	4,427,908	1224.3	2891.7	7,145,901	1443.3	1874.1	3,512,375	1153.2	2059.0	4,239,581	732.7	1837.4	3,485,102	711.7
8	2104.2	4,427,908	1659.2	3231.0	8,943,403	2091.7	2331.1	5,434,424	1336.1	2008.6	4,034,569	1142.2	1998.8	4,007,198	1125.5
9	2132.4	4,547,205	1966.4	3742.6	9,312,340	2744.4	2243.7	5,034,456	1778.8	2111.8	4,459,874	1209.2	2075.5	4,445,120	1193.4
10	1969.6	3,879,616	2423.2	3972.3	9,753,098	3277.3	2007.2	4,029,147	2261.2	1946.7	3,789,905	1383.1	1922.4	3,742,158	1347.6

5.4 Case 3: Truck scheduling problem in a cross-docking system

Fathollahi-Fard et al. (2019) formulated a truck scheduling problem in a cross-docking system. Their proposed model was to determine the scheduling of the truck sequence in the receipt and delivery of commodities. They presented four different versions of the SEO algorithm to solve their proposed model. Each version of SEO includes a change in weights on the SEO features using changes in search strategies. Accordingly, firstly, the MSEO-1 algorithm is presented and changes are made in the training and retraining phase. Thereafter, MSEO-2 algorithm is considered as a new spot for the defender. Then, the MSEO-3 algorithm is proposed as a dynamic parameter for the number of attacks. The proposed algorithms are then combined to generate a new algorithm. For example, the hybridization of MSEO-1 and MSEO-2 algorithms leads to the MSEO-12 algorithm. The hybridization of MSEO-1 and MSEO-3 algorithms leads to the MSEO-13 algorithm. Finally, the hybridization of MSEO-1, MSEO-2, and MSEO-2 algorithms leads to the MSEO-123 algorithm.

Table 13 demonstrates a comparison of the assessment metrics of the proposed algorithms. The lower the Relative Percentage Deviation (RPD) metric, the better the algorithm. The Gap metric is also calculated based on the equation $\mathrm{Gap} = (\mathrm{Z}_{\mathrm{al}} - \mathrm{Z}_{\mathrm{best}})/\mathrm{Z}_{\mathrm{best}}$. As can be seen, their model has been solved by 10 problems with different sizes as well as the results of RPD, Gap, and CPU time criteria have been reported. The CPU time of the MSEO algorithm has been better for all cases than other suggested algorithms. For instance, the first CPU time for the MSEO approach was 1608.5 s and for the MSEO-13, MSEO-12, MSEO-123, and SEO-2 approaches were 1693.2, 1616.6, 1643.0 and 1670.9 s, respectively. In this regard, GAP and RPD values in MSEO show better performance than other algorithms and the outcomes show the superiority of this method over other algorithms.

5.5 Case 4: Production planning under uncertain seasonal demand

Goli et al. (2019) suggested a multi-objective and multi-period model of integrated production planning considering seasonal demand. The purpose of their model was to decrease the costs of outsourcing production, maintenance, and shortages along with maximizing customer satisfaction. The NSGA-II and multi-objective invasive weed optimization algorithm (MOIWO) was used to solve their proposed model. In order to increase the effectiveness of their suggested algorithms, their parameters were estimated by the Taguchi approach. Mean Ideal Distance (MID),

Table 13 Metrics obtained for each algorithm (Case 3)

No.	MSEO-13			MSEO-12			MSEO-123			SEO-2			MSEO		
	RPD	GAP	Time (s)	RPD	GAP	Time (s)	RPD	GAP	Time (s)	RPD	GAP	Time (s)	RPD	GAP	Time (s)
1	0.061	0.08751	1693.2	0.047	0.03833	1616.6	0.034	0.055287	1643.0	0.072	0.07321	1670.9	0.029	0.03819	1608.5
2	0.055	0.02746	1620.3	0.053	0.07123	1689.3	0.030	0.051233	1657.7	0.078	0.09485	1726.5	0.024	0.02713	1614.4
3	0.059	0.06413	1459.9	0.045	0.03728	1423.1	0.033	0.052827	1444.4	0.068	0.02303	1403.6	0.027	0.02224	1394.7
4	0.057	0.02645	1795.2	0.039	0.02631	1795.0	0.029	0.023178	1789.5	0.081	0.09152	1909.0	0.025	0.02309	1775.5
5	0.055	0.02794	1623.1	0.042	0.03865	1640.0	0.037	0.024892	1618.3	0.086	0.06293	1678.3	0.030	0.02598	1610.6
6	0.060	0.03199	1595.4	0.040	0.03139	1594.5	0.035	0.031237	1594.2	0.084	0.03348	1597.7	0.032	0.03125	1590.0
7	0.072	0.05662	1621.9	0.048	0.05093	1613.1	0.040	0.051507	1614.0	0.075	0.06206	1630.2	0.036	0.05011	1607.1
8	0.071	0.05623	1610.7	0.050	0.05065	1602.2	0.048	0.024155	1561.8	0.068	0.10109	1679.1	0.045	0.02245	1521.4
9	0.076	0.08758	1602.0	0.059	0.07964	1590.3	0.043	0.023209	1507.1	0.078	0.10190	1623.1	0.039	0.02197	1506.9
10	0.080	0.10709	1607.5	0.064	0.11814	1623.5	0.038	0.032734	1499.5	0.082	0.10145	1599.3	0.035	0.03063	1385.3

Table 14 Metrics obtained for each algorithm (Case 4)

No.	NSGA-II			MOIWO			MSEO		
	MID	NPS	RAS	MID	NPS	RAS	MID	NPS	RAS
1	2128.40	99.00	0.45	2392.87	29.00	0.50	2115.67	107.00	0.39
2	9901.84	97.00	0.34	10,025.83	7.00	0.23	9983.64	114.00	0.21
3	14,960.24	97.00	0.18	17,064.71	12.00	0.20	13,584.15	110.00	0.16
4	26,614.19	100.00	0.22	29,887.93	11.00	0.02	26,178.54	126.00	0.02
5	43,885.55	95.00	0.27	43,253.99	19.00	0.12	43,189.12	103.00	0.12
6	65,925.99	98.00	0.03	65,007.11	13.00	0.10	64,989.14	109.00	0.04
7	170,150.20	98.00	0.16	172,745.85	35.00	0.15	170,089.55	105.00	0.12
8	252,032.80	99.00	0.11	256,509.70	27.00	0.11	251,975.90	115.00	0.11
9	284,951.50	95.00	0.21	273,177.90	30.00	0.09	271,256.50	113.00	0.08
10	381,924.15	96.00	0.08	367,442.31	37.00	0.08	351,588.65	111.00	0.08
Ave	125,247.5	97.4	0.20	123,750.8	22.00	0.16	120,495.10	111.30	0.13

Number of Partial Solution (NPS), and Rate of Achievement to two objectives Simultaneously (RAS) metrics were also used to assess the efficiency and performance of their proposed algorithms.

Table 14 displays the outcomes comparing the NSGA-II and MOIWO with the developed MSEO algorithm for 10 Pareto points resulting from model solving. The average MID metric for NSGA-II, MOIWO, and MSEO algorithms are 125,247.5, 123,750.8, and 120,495.10, respectively. The average NPS metric for NSGA-II, MOIWO, and MSEO algorithms are 97.4, 22.00, and 111.30, respectively. Finally, the average RAS metric for NSGA-II, MOIWO, and MSEO algorithms are 0.20, 0.16, and 0.13, respectively. It is clear that the MSEO shows more quality than the other two algorithms in terms of MID, NPS, and RAS assessment metrics.

5.6 Statistical test

The Wilcoxon signed-rank test is a nonparametric statistical test that is employed to assess the similarity of two samples related to the ranking scale. This test indicates whether the achieved modification by MSEO is statistically significant or not. This test is conducted based on comparing the outcomes of the MSEO algorithm and other algorithms at a significance level of 5%. Table 15 illustrates the calculated p values by this test. Values of p values less than 0.05 show that the zero hypothesis is rejected. This means that at the 5% level there is a significant difference between the MSEO algorithm and other algorithms. As can be seen, the calculated p values were below 0.05. Therefore, the calculated values statistically confirm the achieved modification by MSEO. Then, it can be said that the results of solving the MSEO approach have

Table 15 P values of the Wilcoxon test of MSEO function approximation outcomes versus other algorithms ($p \geq 0.05$ are underlined)

No.	Case 1		Case 2				Case 3				Case 4	
	NSGA-II	MMOPSO	MOSEO	MOSA	MOKA	MOFFA	MSEO-13	MSEO-12	MSEO-123	SEO-2	NSGA-II	MOIWO
1	3.22E−11	3.39E−11	1.25E−12	1.77E−4	1.42E−3	2.66E−4	6.75E−05	5.29E−05	4.61E−10	6.79E−05	1.86E−03	2.99E−04
2	3.42E−11	3.40E−11	1.20E−12	1.95E−4	1.46E−3	2.60E−4	6.79E−05	5.23E−05	4.31E−10	6.76E−05	1.79E−03	2.68E−04
3	3.61E−11	3.62E−11	1.44E−12	1.55E−4	1.12E−3	2.43E−4	6.70E−05	5.23E−05	4.67E−10	6.75E−05	1.92E−03	2.76E−04
4	3.54E−11	3.76E−11	1.52E−12	1.60E−4	1.74E−3	2.74E−4	6.72E−05	5.29E−05	4.75E−10	6.76E−05	1.65E−03	2.60E−04
5	3.31E−11	3.44E−11	1.63E−12	1.57E−4	1.15E−3	2.67E−4	6.49E−05	5.24E−05	4.63E−10	6.76E−05	1.44E−03	2.37E−04
6	3.16E−11	3.19E−11	1.75E−12	1.54E−4	1.12E−3	2.37E−4	6.85E−05	5.27E−05	4.50E−10	6.77E−05	1.55E−03	2.94E−04
7	3.34E−11	3.84E−11	1.44E−12	1.84E−4	1.15E−3	2.85E−4	6.94E−05	5.36E−05	4.26E−10	6.75E−05	1.79E−03	2.17E−04
8	3.84E−11	3.45E−11	1.24E−12	1.20E−4	1.19E−3	2.79E−4	6.37E−05	5.27E−05	4.91E−10	6.76E−05	1.92E−03	2.19E−04
9	3.62E−11	3.63E−11	1.35E−12	1.41E−4	1.13E−3	2.91E−4	6.71E−05	5.20E−05	4.47E−10	6.74E−05	1.82E−03	2.93E−04
10	3.01E−11	3.28E−11	1.41E−12	1.36E−4	1.10E−3	2.86E−4	6.94E−05	5.26E−05	4.56E−10	6.74E−05	1.47E−03	2.94E−04

superior and more robust than other proposed algorithms in this paper.

6 Conclusion, limitation, and future works

This paper developed a modified metaheuristic method called MSEO for a set of benchmarked optimization problems by using a new adjust operator based on a novel defender and attacker assessment criteria. This idea determined the weight of each defender and attacker. The main goal was to better balance the search accuracy, running time, convergence speed as well as to develop a new generation of the algorithm. In addition to the development of a novel adjust operator, MSEO was compared with a set of famous and recent algorithms including GA, FA, HS, PSO, ACO, ABC, BEEs, ICA, SEO, and BBO algorithms based on 14 benchmark functions covering the characteristics of unimodal, multimodal, and n-dimension multimodal. To prove the efficiency and performance of MSEO, a series of analyses were conducted. Four engineering application instances were presented to prove the proper performance of the proposed algorithm. The first case was according to the location and allocation problems of the shelters for earthquake evacuation planning. In this case, the computational (CPU) time of the MSEO with an average of 15.1 s performs better than the other two algorithms (NSGA-II and MMOPSO). The second case designed a new pharmaceutical supply chain network under uncertainty. In this case, the MSEO algorithm has performed better than MOSEO, MOSA, MOKA, and MOFFA algorithms. Next a scheduling problem of trucks in a cross-docking system was formulated. In this case, GAP and RPD values in MSEO showed better performance than other algorithms and the outcomes showed the superiority of this method over other algorithms. Finally, the last example considered a production planning problem under uncertainty. In this case, the MSEO showed more quality than the NSGA-II and MOIWO in terms of MID, NPS, and RAS assessment metrics. The outcomes were discussed and analyzed in terms of mean normalized results, convergence rate, and standard deviation. The experimental results illustrate that this approach is a feasible and effective way of resolving global numerical optimization problems. Most notably, it was observed that the MSEO algorithm performed better than all versions of proposed algorithms in the majority of case studies.

The main bounds and limitations of the SEO algorithm are summarized as follows: the SEO algorithm is not able to calculate the global optimum and the local optimum. The SEO algorithm also requires access to a computer system equipped with features such as high RAM and CPU. For future works, the proposed MSEO algorithm can be applied to solve other optimization problems, such as the optimization of the pharmaceutical supply chain network, vehicle routing, and scheduling problems. Additionally, hybridization of proposed algorithms with other evolutionary mechanisms such as crossover and mutation operators are possible. Considering the robust measures to propose a robust version of the proposed algorithm is another suggestion for future study.

Acknowledgements This research has been financially supported by The Analytical Center for the Government of the Russian Federation (Agreement No. 70-2021-00143 dd. 01.11.2021, IGK000000D730321P5Q0002).

Funding The authors have not disclosed any funding.

Data availability Enquiries about data availability should be directed to the authors.

Declarations

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

Human and animal rights This article does not contain any studies with human or animal subjects performed by any of the authors.

Informed consent Informed consent was not required as no humans or animals were involved.

References

Abed-Alguni BH, Alawad NA (2021) Distributed Grey Wolf Optimizer for scheduling of workflow applications in cloud environments. Appl Soft Comput 102:107113

Abed-Alguni BH, Alawad NA, Barhoush M, Hammad R (2021) Exploratory cuckoo search for solving single-objective optimization problems. Soft Comput 25(15):10167–10180

Abed-Alguni BH, Alkhateeb F (2017) Novel selection schemes for cuckoo search. Arab J Sci Eng 42(8):3635–3654

Abualigah LMQ (2019) Feature selection and enhanced krill herd algorithm for text document clustering. Springer, Berlin, pp 1–165

Abualigah L, Yousri D, Abd Elaziz M, Ewees AA, Al-Qaness MA, Gandomi AH (2021a) Aquila optimizer: a novel meta-heuristic optimization Algorithm. Comput Ind Eng 157:107250

Abualigah L, Diabat A, Mirjalili S, Abd Elaziz M, Gandomi AH (2021b) The arithmetic optimization algorithm. Comput Methods Appl Mech Eng 376:113609

Alawad NA, Abed-Alguni BH (2021a) Discrete Jaya with refraction learning and three mutation methods for the permutation flow shop scheduling problem. J Supercomput 1–22

Alawad NA, Abed-Alguni BH (2021b) Discrete island-based cuckoo search with highly disruptive polynomial mutation and opposition-based learning strategy for scheduling of workflow applications in cloud environments. Arab J Sci Eng 46(4):3213–3233

Aslan S (2019) A transition control mechanism for artificial bee colony (ABC) algorithm. Comput Intell Neurosci. https://doi.org/10.1155/2019/5012313

Balaji AN, Porselvi S, Jawahar N (2019) Particle swarm optimization algorithm and multi-start simulated annealing algorithm for scheduling batches of parts in multi-cell flexible manufacturing system. Int J Serv Oper Manag 32(1):83–129

Bottani E, Murino T, Schiavo M, Akkerman R (2020) Resilient food supply chain design: modelling framework and metaheuristic solution approach. Comput Ind Eng 157:274–289

Castellani M, Otri S, Pham DT (2019) Printed circuit board assembly time minimization using a novel Bees algorithm. Comput Ind Eng 133:186–194

Dhiman G, Kaur A (2019) STOA: a bio-inspired based optimization algorithm for industrial engineering problems. Eng Appl Artif Intell 82:148–174

Elarbi S, Bechikh A, Gupta LB, Said YS (2018) Ong, a new decomposition-based NSGA-II for many-objective optimization. IEEE Trans Syst, Man, Cybern: Syst 48(7):1191–1210

Fakhrzad MB, Goodarzian F (2019) A fuzzy multi-objective programming approach to develop a green closed-loop supply chain network design problem under uncertainty: modifications of imperialist competitive algorithm. RAIRO-Oper Res 53(3):963–990

Fathollahi-Fard AM, Hajiaghaei-Keshteli M, Tavakkoli-Moghaddam R (2018) The social engineering optimizer (SEO). Eng Appl Artif Intell 72:267–293

Fathollahi-Fard AM, Ranjbar-Bourani M, Cheikhrouhou N, Hajiaghaei-Keshteli M (2019) Novel modifications of social engineering optimizer to solve a truck scheduling problem in a cross-docking system. Comput Ind Eng 137:106103

Geem ZW, Kim JH, Loganathan GV (2001) A new heuristic optimization algorithm: harmony search. SIMULATION 76(2):60–68

Ghasemi P, Khalili-Damghani K, Hafezalkotob A, Raissi S (2019) Uncertain multi-objective multi-commodity multi-period multi-vehicle location-allocation model for earthquake evacuation planning. Appl Math Comput 350:105–132

Glover F (1989) Tabu search—part I. *ORSA*. J Comput 1(3):190–206

Goli A, Tirkolaee EB, Malmir B, Bian GB, Sangaiah AK (2019) A multi-objective invasive weed optimization algorithm for robust aggregate production planning under uncertain seasonal demand. Computing 101(6):499–529

Goodarzian F, Hosseini-Nasab H, Muñuzuri J, Fakhrzad MB (2020) A multi-objective pharmaceutical supply chain network based on a robust fuzzy model: a comparison of meta-heuristics. Appl Soft Comput 92:106331

Goodarzian F, Kumar V, Abraham A (2021a) Hybrid meta-heuristic algorithms for a supply chain network considering different carbon emission regulations using big data characteristics. Soft Comput 25(11):7527–7557

Goodarzian F, Kumar V, Ghasemi P (2021b) "A set of efficient heuristics and meta-heuristics to solve a multi-objective pharmaceutical supply chain network. Comput Ind Eng 158:107389

Goodarzian F, Abraham A, Fathollahi-Fard AM (2021c) A biobjective home health care logistics considering the working time and route balancing: a self-adaptive social engineering optimizer. J Comput Des Eng 8(1):452–474

Goodarzian F, Ghasemi P, Gunasekaren A, Taleizadeh AA, Abraham A (2021d) A sustainable-resilience healthcare network for handling COVID-19 pandemic. Ann Oper Res. https://doi.org/10.1007/s10479-021-04238-2

Goodarzian F, Taleizadeh AA, Ghasemi P, Abraham A (2021e) An integrated sustainable medical supply chain network during COVID-19. Eng Appl Artif Intell 100:104188

Goudarzi S, Anisi MH, Abdullah AH, Lloret J, Soleymani SA, Hassan WH (2019) A hybrid intelligent model for network selection in the industrial internet of things. Appl Soft Comput 74:529–546

Hajipour V, Tavana M, Di Caprio D, Akhgar M, Jabbari Y (2020) An optimization model for traceable closed-loop supply chain networks. Appl Math Model 84:265–279

Kamaruddin S, Latif MAHA (2019) Application of the bees algorithm for constrained mechanical design optimization problem. Int J Eng Mater Manuf 4(1):27–32
Karaboga D, Basturk B (2007) Artificial bee colony (ABC) optimization algorithm for solving constrained optimization problems. In: International fuzzy systems association world congress, pp 789–798. Springer, Berlin, Heidelberg
Kennedy J, Eberhart R (1995) Particle swarm optimization (PSO). In: Proc. IEEE international conference on neural networks, Perth, Australia, pp 1942–1948
Lei D, Li M, Wang L (2018) A two-phase meta-heuristic for multi-objective flexible job shop scheduling problem with total energy consumption threshold. IEEE Trans Cybern 49(3):1097–1109
Li J, Zhang JQ, Jiang CJ, Zhou MC (2015) Composite particle swarm optimizer with historical memory for function optimization. IEEE Trans Cybern 45(10):2350–2363
Luan J, Yao Z, Zhao F, Song X (2019) A novel method to solve supplier selection problem: hybrid algorithm of genetic algorithm and ant colony optimization. Math Comput Simul 156:294–309
Meng X, Li J, Zhou M, Dai X, Dou J (2016) Population-based incremental learning algorithm for a serial colored traveling salesman problem. IEEE Trans Syst, Man, Cybern: Syst 48(2):277–288
Mitchell M, Forrest S, Holland JH (1992) The royal road for genetic algorithms: fitness landscapes and GA performance. In: Proceedings of the 1st European conference on artificial life, pp 245–254
Nazari-Shirkouhi S, Eivazy H, Ghodsi R, Rezaie K, Atashpaz-Gargari E (2010) Solving the integrated product mix-outsourcing problem using the imperialist competitive algorithm. Expert Syst Appl 37(12):7615–7626
Okulewicz M, Mańdziuk J (2019) A metaheuristic approach to solve dynamic vehicle routing problem in continuous search space. Swarm Evol Comput 109:236–245
Pech S, Kandler G, Lukacevic M, Füssl J (2019) Metamodel assisted optimization of glued laminated timber beams by using metaheuristic algorithms. Eng Appl Artif Intell 79:129–141
Pham DT, Ghanbarzadeh A, Koc E, Otri S, Rahim S, Zaidi M (2005) The bee's algorithm. Technical note, Manufacturing Engineering Centre, Cardiff University, UK
Rakshit P, Konar A, Das S, Jain LC, Nagar AK (2014) Uncertainty management in differential evolution induced multi-objective optimization in presence of measurement noise. IEEE Trans Syst, Man, Cybern: Syst 44(7):922–937
Ramírez A, Romero JR, García-Martínez C, Ventura S (2019) JCLEC-MO: a Java suite for solving many-objective optimization engineering problems. Eng Appl Artif Intell 81:14–28
Sacramento D, Pisinger D, Ropke S (2019) An adaptive large neighborhood search metaheuristic for the vehicle routing problem with drones. Transp Res Part c: Emerging Technol 102:289–315
Shadravan S, Naji HR, Bardsiri VK (2019) The Sailfish Optimizer: a novel nature-inspired metaheuristic algorithm for solving constrained engineering optimization problems. Eng Appl Artif Intell 80:20–34
Shang Y, Nguyen H, Bui XN, Tran QH, Moayedi H (2020) A novel artificial intelligence approach to predict blast-induced ground vibration in open-pit mines based on the firefly algorithm and artificial neural network. Nat Resour Res 29(2):723–737
Simon D (2008) Biogeography-based optimization. IEEE Trans Evol Comput 12(6):702–713
Stützle T, Dorigo M (1999) A CO algorithms for the traveling salesman problem. Evol Algor Eng Comput Sci 4:163–183
Tam JH, Ong ZC, Ismail Z, Ang BC, Khoo SY (2019) A new hybrid GA−ACO−PSO algorithm for solving various engineering design problems. Int J Comput Math 96(5):883–919
Wang K, Luo H, Liu F, Yue X (2018) Permutation flow shop scheduling with batch delivery to multiple customers in supply chains. IEEE Trans Syst, Man, Cybern: Syst 48(10):1826–1837
Yadegari E, Alem-Tabriz A, Zandieh M (2019) A memetic algorithm with a novel neighbourhood search and modified solution representation for closed-loop supply chain network design. Comput Ind Eng 128:418–436
Yang XS (2010) Firefly algorithm, Levy flights and global optimization. In: Research and development in intelligent systems, vol XXVI, pp 209–218. Springer, London
Yi J, Gao L, Li X, Shoemaker CA, Lu C (2019) An on-line variable-fidelity surrogate-assisted harmony search algorithm with multi-level screening strategy for expensive engineering design optimization. Knowl-Based Syst 170:1–19
Zhang X, Du KJ, Zhan ZH, Kwong S, Gu TL, Zhang J (2020) Cooperative coevolutionary bare-bones particle swarm optimization with function independent decomposition for large-scale supply chain network design with uncertainties. IEEE Trans Cybern. https://doi.org/10.1109/TCYB.2019.2937565

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.