Contents lists available at GrowingScience

International Journal of Industrial Engineering Computations

homepage: www.GrowingScience.com/ijiec

A new hybrid algorithm based on MVO and SA for function optimization

Ömer Yılmaz[a*], Adem Alpaslan Altun[b] and Murat Köklü[b]

[a]*Department of Information Technologies, Tokat Vocational and Technical Anatolian High School, 60100, Tokat, Turkey*
[b]*Department of Computer Engineering, Faculty of Technology, Konya Selcuk University, 42130, Konya, Turkey*

C H R O N I C L E

Article history:
Received May 11 2021
Received in Revised Format
June 28 2021
Accepted October 22 2021
Available online
October, 27 2021

Keywords:
Simulated annealing
Multi-verse optimizer
Hybrid optimization algorithm
Function optimization

A B S T R A C T

Hybrid algorithms are widely used today to increase the performance of existing algorithms. In this paper, a new hybrid algorithm called IMVOSA that is based on multi-verse optimizer (MVO) and simulated annealing (SA) is used. In this model, a new method called the black hole selection (BHS) is proposed, in which exploration and exploitation can be increased. In the BHS method, the acceptance probability feature of the SA algorithm is used to increase exploitation by searching for the best regions found by the MVO algorithm. The proposed IMVOSA algorithm has been tested on 50 benchmark functions. The performance of IMVOSA has been compared with other latest and well-known metaheuristic algorithms. The consequences show that IMVOSA produces highly successful and competitive results.

1. Introduction

Optimization is defined as the process of finding the best solution among alternative solutions in line with the conditions given for a specific problem. The basic goal of the optimization method is to find the necessary parameters for the best result of the fitness function (Murty, 2003). Due to the tremendous recent development of information technology, the use of optimization methods has increased. Many real-world problems can be seen as optimization problems and many algorithms have been used to solve optimization problems. Metaheuristic algorithms are the popular algorithms that are used for solving optimization problems.

Metaheuristic algorithms aim to examine the search space effectively and efficiently in optimization problems where the mathematical model cannot be established or where it is very costly to build a model. Although it is not always possible to find the best global solution with these algorithms, the convenience of their application, their ability to produce fast and effective solutions to large-scale and complex problems, the fact that the metaheuristic method developed for any problem can also be applied to other problems makes these methods very useful (Kaya & Fığlalı 2018; Talbi, 2009). The most important advantage of the metaheuristic algorithm can be said to be the ability to reach the global best without getting stuck with the local best (Laporte et al., 2000). Considering the publications, there are various metaheuristic algorithms that have been used and accepted in many studies. Differential Evolution (DE) (Storn, 1996; Storn & Price, 1997), Ant Colony Optimization (ACO) (Colorni et al., 1991; Jovanovic & Tuba, 2013), Artificial Bee Colony (ABC) (Karaboga, 2005), Gravity Search Algorithm (GSA) (Rashedi et al., 2009), Cat Swarm Optimization (CSO) (Chu et al., 2006), Animal Migration Optimization (AMO) (Li et al., 2014; Luo et al., 2016), Particle Swarm Optimization (PSO) (Kennedy & Eberhart, 1995), Simulated Annealing (SA) (Kirkpatrick et al., 1983), Harris Hawks Optimization (HHO) (Heidari et al., 2019), Multi-verse Optimizer (MVO) (Mirjalili et al., 2016) algorithms can be given as examples of metaheuristic algorithms.

* Corresponding author
E-mail: omerryilmaz@gmail.com (Ö. Yılmaz)

2022 Growing Science Ltd.
doi: 10.5267/j.ijiec.2021.11.001

The common goal of metaheuristic algorithms is to find the best result in current conditions. In order for algorithms to achieve this common goal, they must have two main features. These two main features are exploration and exploitation. Achieving the balance between exploration and exploitation will significantly enhance the success of the algorithm. When researches are examined, it is seen that metaheuristic algorithms are classified as population-based algorithms (e.g., evolutionary algorithms, swarm intelligence) that are exploration-oriented, and single-solution algorithms (e.g., simulated annealing, local search) that are exploitation-oriented (Mafarja & Mirjalili, 2017). Combining optimization algorithms is a way to balance exploration and exploitation capability (Mirjalili & Hashim, 2010; Mafarja & Mirjalili, 2017). Combining a metaheuristic algorithm with at least one algorithm using different and advantageous aspects is defined as a hybrid metaheuristic algorithm. In order to increase the efficiency of optimization algorithms, it is seen that various processes such as hybridization, improvement and modification are developed in many studies (Alizada, 2019). Considering the studies conducted, it is seen that generally good results are obtained when applied in an optimization problem using the advantageous aspects of two or more algorithms. Our aim in this study is to combine MVO and SA algorithms with a new hybrid method to increase performance for function optimization.

In general, hybrid algorithms can be basically detached into two groups as collaborative and integrative (Ting et al., 2015). Collaborative hybrids are a combination of multiple algorithms, operating in sequence or in parallel. In collaborative hybrids, the effects of algorithms on performance are half. Integrative hybrids are created by integrating a secondary algorithm into a main algorithm. This is accomplished by replacing a function in the metaheuristic algorithm with another metaheuristic (Talbi, 2002). In integrative hybrids, the effect of the second algorithm on the results is much less than the main algorithm. In this paper, the integrative hybrid model in which a population-based algorithm (MVO) is hybridized with another single-solution-based algorithm (SA) is used for function optimization. In the proposed model, the SA algorithm will increase the exploitation in the MVO algorithm. Different hybrid models of metaheuristic algorithms have been developed for function optimization so far. However, a hybrid model using MVO and SA algorithms has not been encountered in the studies and this model will be applied for the first time.

The SA algorithm has an important place among metaheuristic algorithms and has been used in many studies. SA is one of the local search algorithms developed for the solution of combinatorial optimization problems and used in the solution of continuous and discrete problems (Alizada, 2019; Henderson et al., 2003). The most remarkable characteristic of the SA algorithm is its capability to avoid the local best. In the algorithm, it is sometimes possible to get rid of local best points by accepting randomly determined aspirant solutions that may cause an increase or decrease in the objective function (Dupanloup et al., 2002). The major shortcoming of SA is that its efficiency is not satisfactory. This is because SA cannot learn enough from its search history while sampling to generate aspirant solutions (Wang et al., 2016).

MVO algorithm is one of the current metaheuristic optimization algorithms suggested in 2015 (Mirjalili et al., 2016). MVO's inspiration is the three basic notions of multiverse theory: wormholes, black holes and white holes. The notions of black hole and white hole are used by MVO to explore search areas. In contrast, wormholes are used for exploitation in the local area achieved through the exploration stage to find the best global solution. Mirjalili, the architect of this algorithm, showed in his study that MVO achieved ambitious consequences compared to other metaheuristic algorithms. However, information exchange is not sufficient in the structure of the MVO algorithm, so it has problems such as low accuracy, slow convergence, changeable system and easily stuck to local minimums (Jia et al., 2019; Song et al., 2020).

GSMVO (Jia et al., 2019), H-MVO (Abasi et al., 2020), QMVO (Sayed et al., 2019), DE-SMVO (Chen et al., 2021), HPSO-MVO (Jangir et al., 2017), WOASA (Mafarja & Mirjalili, 2017), SA-MFO (Sayed & Hassanien, 2018), SA-PSO (Pan et al., 2019), CSA (Alkhateeb & Abed-Alguni, 2019), HHOBSA (Abdel-Basset et al., 2021) are many studies in which MVO and SA are hybridized with other algorithms and are used in subjects such as function optimization, engineering problems, training of artificial neural networks. These hybrid algorithms aim to reduce the likelihood of catching a local best. In this study, a hybridization method was tried to complement the shortcomings of MVO. It was seen that there was no hybrid study conducted with MVO and SA in the literature review. In this study, a simulated annealing method was applied while updating the universes in the standard version of MVO to further increase the success of MVO. IMVOSA has SA's ability to escape local best and MVO's learning mechanism and quick search capability to guide the creation of aspirant solutions. IMVOSA was tested with 50 well-known benchmark functions that are commonly used in the literature for experimental study. Test consequences show that the suggested algorithm can meaningfully improve the success of the MVO. Furthermore, experimental consequences show that IMVOSA produces better results than other algorithms.

In the second part of this study, the original MVO and SA algorithms are briefly explained. In the third chapter, detailed information about the proposed hybrid algorithm (IMVOSA) and its components is given. In the fourth chapter, comparative test results with other optimization algorithms of IMVOSA are given and the success of the proposed algorithm is evaluated. Finally, in the fifth chapter, the consequences obtained from the study were evaluated and suggestions were made for prospective studies.

2. MVO and SA algorithms

2.1. MVO

Multi-verse optimizer was suggested by Seyedali Mirjalili in 2015 (Mirjalili et al., 2016). Inspired by the notions of wormholes, black holes and white holes in multiverse theory and the big bang theory. Wormholes, black holes and white holes are mathematically modeled for local search, exploration and exploitation in this population-based algorithm. The objective function for each search agent is specified by the inflation rate. Each object and each universe in the search agent represent a variable and an aspirant solution.

Object exchange among universes occurs when universes which have high inflation rates try to send objects to universes which have low inflation rates. However, in order to be a stable universe in low inflation rate universes, it takes objects from universes with high inflation rates. In the optimization process, the above steps are initiated in each iteration and then adjusted according to the inflation rates.

During the optimization process, the following rules apply to MVO's universes:

- the high rate of inflation increases the opportunity of having white holes;
- the low rate of inflation increases the opportunity of having black holes;
- universes which have high inflation rates frequently send objects to white holes;
- universes which have low inflation rates are more likely to take objects from black holes;
- objects in every universe can perform a random move approaching the best universe, independent of the rate of inflation with wormholes. There is invariably an opportunity to transfer objects from a universe which has a high inflation rate to a universe which has a low inflation rate.

The conceptual model of the proposed algorithm is shown in Fig. 1.

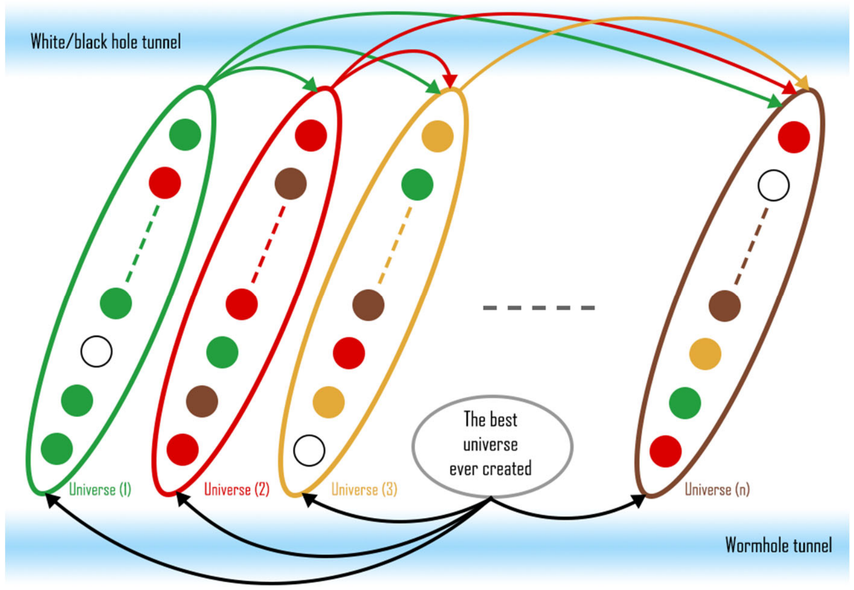

Fig. 1. Conceptual Model of MVO Algorithm (Mirjalili et al., 2016).

The mathematical model of this algorithm is as follows:

$$U = \begin{bmatrix} x_1^1 & x_1^2 & \dots & x_1^d \\ x_2^1 & x_2^2 & \dots & x_2^d \\ \vdots & \vdots & & \vdots \\ x_n^1 & x_n^2 & \dots & x_n^d \end{bmatrix} \quad (1)$$

where n is the number of universes (aspirant solutions), d is the number of parameters (variables)

$$x_i^j = \begin{cases} x_k^j & rnd1 < NR(UN_i) \\ x_i^j & rnd1 \geq NR(UN_i) \end{cases} \quad (2)$$

where x_i^j identifies the jth parameter of ith universe, x_k^j identifies the jth parameter of kth universe chosen by a roulette wheel selection, $rnd1$ is a number obtained randomly in the range [0, 1], $NR(UN_i)$ is normalized inflation rate of the ith universe, UN_i shows the ith universe.

It is seen that wormholes randomly alter the objects of universes without taking into account inflation rates in order to preserve the variety of the universes and to exploit them. Wormhole tunnels are always assumed to be built between a universe and the best universe ever generated. The mathematical model of this structure is as follows:

$$x_i^j = \begin{cases} \begin{cases} X_j + \text{TDR} \times ((ub_j - lb_j) \times rnd4 + lb_j) & rnd3 < 0{,}5 \\ X_j - \text{TDR} \times ((ub_j - lb_j) \times rnd4 + lb_j) & rnd3 \geq 0{,}5 \end{cases} & rnd2 < \text{WEP} \\ x_i^j & rnd2 \geq \text{WEP} \end{cases} \tag{3}$$

where X_j identifies the jth parameter of best universe generated up to now, x_i^j identifies the jth parameter of ith universe, WEP and TDR are the coefficients, ub_j is the upper bound of jth variable, lb_j shows the lower bound of j th variable, and $rnd2$, $rnd3$, $rnd4$ are numbers obtained randomly in the range [0, 1].
From Eq. (3) it can be seen that the MVO has two coefficients: Wormhole existence probability (WEP) and Travelling distance rate (TDR). The mathematical model for both coefficients is as follows:

$$\text{WEP} = \min + l \times \left(\frac{\max - \min}{L}\right) \tag{4}$$

where, L is the maximum iterations, l is the current iteration, max is the maximum value which is 1 the original MVO paper and min is the minimum value which is 0.2 in the original MVO paper.

$$\text{TDR} = 1 - \frac{l^{1/p}}{L^{1/p}} \tag{5}$$

where p identifies the accuracy of exploitation on iterations. Higher value p means earlier and more correct local search - exploitation. The pseudo-code for MVO algorithm is given in Algorithm 1. Details about the algorithm can also be found in (Mirjalili et al., 2016).

Algorithm 1. MVO

```
 1: Generate random universes (UN)
 2: Initialize BestUniverse, WormholeExistenceProbability, TravellingDistanceRate
 3: SUN = Sorted universes
 4: NR = Normalize inflation rates (fitness) of the universes
 5: while Time < Max_time
 6:   Assess the fitness of all universes
 7:   for each universe indexed by i
 8:     Update TravellingDistanceRate, WormholeExistenceProbability
 9:     BlackHoleIndex = i;
10:     for each object indexed by j
11:       rnd1 = random([0, 1]);
12:       if rnd1 < NR(U_i)
13:         WhiteHoleIndex = Roulette_Wheel_Selection(-NR);
14:         UN(BlackHoleIndex, j) = SUN(WhiteHoleIndex,j);
15:       end if
16:       rnd2 = random([0, 1]);
17:       if rnd2 < WormholeExistenceProbability
18:         rnd3 = random([0, 1]);
19:         rnd4 = random([0, 1]);
20:         if rnd3 < 0.5
21:           UN(i,j)= BestUniverse(j)+TravellingDistanceRate ×
                       ((ub(j) - lb(j)) × rnd4 + lb(j));
22:         else
23:           UN(i,j)= BestUniverse(j)-TravellingDistanceRate ×
                       ((ub(j) - lb(j)) × rnd4 + lb(j));
24:         end if
25:       end if
26:     end for
27:   end for
28: end while
```

2.2. SA algorithm

Simulated annealing was suggested by Kirkpatrick et al. (1983), based on the hill climbing method and applied to solve optimization problems. The algorithm is inspirited by the annealing operation, which is based on replacing the properties of a particular material with heat treatment. The creation of fresh solutions in the simulated annealing algorithm is performed randomly or based on predetermined rules. At each iteration, the existing solution is compared with the newly created solution. SA can accept not only new solutions that improve the existing solution, but also worse results that meet certain criteria in order not to stick to the local best and to find the global best. The acceptance probability of new solutions that do not improve the existing solution varies depending on the temperature (Henderson et al., 2003). These criteria are determined by the Boltzmann probability, and this mechanism is defined as follows:

$$\mathrm{Pr} = e^{\left(\frac{-(F(Y)-F(Y_0))}{T_k}\right)} \tag{6}$$

where Pr is identified as the acceptance probability. $F(Y_0)$ represents the objective function for current solution and $F(Y)$ represents the objective function for neighbor solution. When $F(Y_0)$ is better than $F(Y)$, SA uses the acceptance probability mechanism to determine whether the neighbor solution should be considered as the current solution. T_k is the temperature value at time k and its value in each iteration is calculated as follows:

$$T_{k+1} = a \times T_k \tag{7}$$

where a refers to the temperature coefficient. In studies, the a value generally takes a value between 0.8 and 0.99. T_k is the initial temperature value and T_{k+1} is the temperature at time k, r is a number obtained randomly in the range [0, 1]. Starting the simulated annealing algorithm with a relatively high temperature value will prevent it from being attached to the local best (Eglese, 1990). Detailed information about the simulated annealing algorithm can be found in (Kirkpatrick et al., 1983). The pseudo-code for SA algorithm is given in Algorithm 2.

Algorithm 2. SA

```
1:  $T$ = Temperature value
2:  $T_0$ = Final temperature value
3:  $a$ = Temperature coefficient
4:  $S$ = First solution
5:  $f(S)$ = Fitness value of the first solution
6:  while $T > T_0$
7:      $S'$ = A new solution in the $S$ neighborhood
8:      $f(S')$ = Calculate the fitness value of $S'$
9:      if $f(S') < f(S)$
10:         $NS = S'$;
11:         $f(NS) = f(S')$;
12:     else
13:         $\Delta f = f(S') - f(S)$
14:         $r = random[0, 1]$;
15:         if $r > exp(-\Delta f / T)$
16:             $NS = S'$;
17:             $f(NS) = f(S')$;
18:         else
19:             $NS = S$;
20:             $f(NS) = f(S)$;
21:         end if
22:     end if
23:     $S = NS$;
24:     $T = T \times a$
25: end while
```

3. IMVOSA algorithm

MVO is a population-based algorithm with successful consequences in many optimization problems. Although the MVO algorithm has many advantages, the exchange of information in the structure of this algorithm is not enough, so it has disadvantages such as slow convergence, low accuracy and sticking to local minimums (Jia et al., 2019; Song et al., 2020). To overcome these shortcomings, improvements to the original algorithm's structure are proposed. Details of these improvements are described below.

In the MVO algorithm, updating objects in a universe is performed by a simple location update (Jia et al., 2019). Equation 2, used during the exploration phase of the original MVO algorithm, is proposed as follows to increase the exploration and exploitation of the universe near the best universe.

$$x_i^j = \begin{cases} x_k^j & rnd1 < NI(U_i) \\ X_j & rnd1 \geq NI(U_i) \end{cases} \quad (8)$$

where x_i^j identifies the jth parameter of ith universe, X_j identifies the jth parameter of best universe generated up to now, x_k^j identifies the jth parameter of kth universe chosen by a roulette wheel selection , $rnd1$ is a number obtained randomly in the range [0, 1], UN_i shows the ith universe, $NR(UN_i)$ is normalized inflation rate of the ith universe.

Tidal disruption is an astronomic event that happens when a star gets close enough to a supermassive black hole and the black hole shreds by tidal force, resulting in an event called spaghettification (Hawking, 1988; Komossa, 2015). Due to the tidal event, all objects in the black hole's gravitational field are pulled towards the center of the black hole, forming spirals around the black hole. In supermassive black holes, it is possible to cross the event horizon in one piece. However, this is impossible in smaller massive black holes. Today, there is no proven, clear information about the state of objects crossing the event horizon regarding black holes.

Based on the tidal disruption effect of black holes and the phenomenon of spaghettification of objects exposed to this effect, we propose a new method in the MVO algorithm. In this method, in order to achieve global best, objects that cross the event horizon and enter the black hole can be altered by the influence of the black hole, in this way exploration and exploitation can be increased. In this method, we used the SA algorithm in an integrative hybrid model structure to avoid local minimum and increase exploitation.

This method, which we call the black hole selection method (BHS), obtains a new set of solutions consisting of the total number of probabilities, taking into account the possibility that each object is in three positions when the black holes of low inflation-value universes draw objects from high inflation-value universes. Within this solution set, a sub-solution set is created at certain intervals and randomly determined in each iteration. From this sub-solution set, the universe with the best fitness value is selected by the black hole. The mathematical representation of the proposed method is as follows:

$$NU = 3^d \quad (9)$$

NU shows the total number of universes calculated for three locations of objects in a black hole, and d (Equation 1) identifies the number of objects in a universe.

$$R = \frac{NU}{max+1-l} \quad (10)$$

R identifies an integer identifying the number of universes to search, NU (Equation 9) identifies the total number of universes calculated for three locations, max (Equation 4) identifies the total number of iterations, l (Equation 4) identifies the number of valid iterations.

$$rnd2 = [0. R] \quad (11)$$

$rnd2$ is a random integer that takes a value between 0 and R (Equation 10). R identifies the number of universes to search.

$$NSU = rnd2 \times (max + 1 - l) \quad (12)$$

NSU identifies the number of the first universe to start searching, $rnd2$ (Equation11) is a randomly selected integer, max (Equation 4) is the total number of iterations, l (Equation 4) is the number of valid iterations.

$$NEU = rnd2 \times (max + 1 - l) + (max - l) \quad (13)$$

NEU identifies the number of the last universe to complete the search, $rnd2$ (Equation11) is a randomly selected integer, max (Equation 4) the total number of iterations, l (Equation 4) the number of current iterations. In this study, the three location values that each object can be found in the black hole are calculated as follows:

$$x_j = X_j - x_j \quad (14)$$
$$x_j = x_j + TDR \times rnd3 \times (X_j - x_j) \quad (15)$$
$$x_j = x_j - TDR \times rnd3 \times (X_j - x_j) \quad (16)$$

where x_j identifies the jth parameter of thre universe inside the black hole, X_j identifies the jth parameter of best universe generated up to now, TDR (Equation 5) is a coefficient, $rnd3$ is a number obtained randomly in the range [0, 1].

In our proposed BHS method, we used the acceptance probability feature of the SA algorithm to make acceptable not only new solutions that improve the existing solution, but also worse results that meet certain criteria in order to avoid to the local best and to find the global best. In this way, the values of the objective function of the solutions produced will not tend to decrease continuously, and in some cases, the solutions with high objective function will be accepted and the search for the global best will be performed. The main purpose of metaheuristic algorithms is to find the best solution or close to best solutions. For this purpose, the two basic components of metaheuristic algorithms, exploration-exploitation concepts come to the fore (Blum & Roli, 2003). Exploration generally means the ability to visit many and different regions of the search area, while exploitation means the ability to obtain better solutions in areas identified by exploration. A good and balanced combination of these two main components will increase the chances of achieving a global solution (Yang, 2010). In this context, we have made the following changes in the exploitation phase (Equation 3) of the standard MVO algorithm to attain a balance between exploration and exploitation in the IMVOSA algorithm.

$$x_i^j = \begin{cases} \begin{cases} X_j + \text{TDR} \times ((ub_j - lb_j) \times rnd6 + lb_j) & rnd5 < 0{,}5 \\ X_j - \text{TDR} \times ((ub_j - lb_j) \times rnd6 + lb_j) & rnd5 \geq 0{,}5 \end{cases} & rnd4 > \text{WEP} \\ x_i^j & rnd4 \leq \text{WEP} \end{cases} \tag{17}$$

where X_j identifies the jth parameter of best universe generated up to now, WEP and TDR are the two main coefficients, ub_j is the upper bound of jth variable, lb_j is the lower bound of j th variable, x_i^j identifies the jth parameter of ith universe, and $rnd4$, $rnd5$, $rnd6$ are numbers obtained randomly in the range [0, 1]. Unlike the standard MVO algorithm, an update will be made if the $rnd4$ value is greater than the WEP coefficient. This process occurs when $rnd4$ value is less than WEP coefficient in standard MVO algorithm. With the change made in this section, the local search or exploitation near the best solution found up to the current iteration, tends to decrease from the first iteration to the last iteration, as opposed to the standard MVO algorithm. The pseudo-code of the IMVOSA algorithm is as follows:

Algorithm 3. IMVOSA (continued overleaf)

```
1:  Generate random universes (UN)
2:  Initialize BestUniverse, WormholeExistenceProbability, TravellingDistanceRate
3:  SUN = Sorted universes
4:  NR = Normalize inflation rates (fitness) of the universes
5:  while Time < Max_time
6:    Assess the fitness of all universes
7:    for each universe indexed by i
8:      Update TravellingDistanceRate, WormholeExistenceProbability
9:      BlackHoleIndex = i;
10:     for each object indexed by j
11:       rnd1 = random([0, 1]);
12:       if rnd1 < NR(U_i)
13:         WhiteHoleIndex = Roulette_Wheel_Selection(-NR);
14:         UN(BlackHoleIndex, j) = SUN(WhiteHoleIndex,j);
15:       else
16:         UN(BlackHoleIndex, j) = BestUniverse(j);
17:       end if
18:     end for
19:     S = UN(BlackHoleIndex)
20:     f(U) = Calculate fitness value of UN(BlackHoleIndex)
21:     Calculate NU, R, NSU, NEU values according to Equations 9, 10, 12, 13
22:     if NEU > NU
23:       NEU = NU
24:     for each universe in the sub-solution set
25:       for each object indexed by j
26:         m = k % $3^{j+1}$
27:         if m < $3^j$
28:           S(j) = BestUniverse(j) - UN(BlackHoleIndex, j)
29:         else
30:           rnd2=random([0,1]);
31:           if m<$3^j \times 2$
32:             S(j)=UN(BlackHoleIndex,j)+TravellingDistanceRate ×
                    rnd2 ×BestUniverse(j)-UN(BlackHoleIndex, j))
33:           else
34:             S(j)=UN(BlackHoleIndex,j)-TravellingDistanceRate ×
                    rnd2 ×BestUniverse(j)-UN(BlackHoleIndex, j))
35:           end if
36:         end if
37:       end for
38:       f(S) = Calculate the fitness value of S
39:       if f(S) < f(UN(BlackHoleIndex))
```

Algorithm 3. (continued)

```
40:                 $UN(BlackHoleIndex) = S$
41:             else
42:                 $\Delta f = f(S) - f(UN(BlackHoleIndex))$
43:                 $rnd3 = random([0, 1]);$
44:                 if $rnd3 < exp(-\Delta f / T)$
45:                     $UN(BlackHoleIndex) = S$
46:                 end if
47:             end if
48:             $T = T \times a$
49:         end for
50:         for each object indexed by $j$
51:             $rnd4 = random([0,1]);$
52:             if $rnd4 < WormholeExistenceProbability$
53:                 $rnd5 = random([0, 1]);$
54:                 $rnd6 = random([0, 1]);$
55:                 if $rnd5 < 0.5$
56:                     $UN(i,j)= BestUniverse(j)+ TravellingDistanceRate \times$
                            $((ub(j) - lb(j)) \times rnd6 + lb(j));$
57:                 else
58:                     $UN(i,j)= BestUniverse(j)- TravellingDistanceRate \times$
                            $((ub(j) - lb(j)) \times rnd6 + lb(j));$
59:                 end if
60:             end if
61:         end for
62:     end for
63: end while
```

4. Experimental Results

4.1. Test Functions and Comparison Algorithms

In this study, 50 benchmark functions with different characteristics commonly used in publications were used to assess the success of the proposed IMVOSA algorithm. Mathematical formulas and properties of benchmark functions are listed in Table 1. As shown in table 1, functions such as F_1 - F_{20} unimodal and F_{21} - F_{50} multimodal are divided into two groups. Unimodal functions with a global best reveal the exploitation capabilities of algorithms, while multimodal functions demonstrate the algorithms' ability to exploration and avoid local minimums. Unimodal functions with a global best reveal the exploitation capabilities of algorithms, while multimodal functions demonstrate the ability of algorithms to avoid local minimums and to exploration. Bounds, dimensions and global minimum values of functions are listed in Table 2.

Table 1
Unimodal and multimodal benchmark functions (continued overleaves)

Name	Type	Equation
Ackley N.2	Uni	$F_1(x) = -200e^{-0.2\sqrt{x^2+y^2}}$
Booth	Uni	$F_2(x) = (x_1 + 2x_2 - 7)^2 + (2x_1 + x_2 - 5)^2$
Brent	Uni	$F_3(x) = (x_1 + 10)^2 + (x_2 + 10)^2 + e^{-x^2-y^2}$
Brown	Uni	$F_4(x) = \sum_{i=1}^{d-1} (x_i^2)^{(x_{i+1}^2+1)} + (x_{i+1}^2)^{(x_i^2+1)}$
Dixon & Price	Uni	$F_5(x) = (x_1 - 1)^2 + \sum_{i=2}^{d} i(2x_i^2 - x_i - 1)^2$
Drop-Wave	Uni	$F_6(x) = -\dfrac{1+\cos\left(12\sqrt{x_1^2+x_2^2}\right)}{\left(0.5\left(x_1^2+x_2^2\right)+2\right)}$
Leon	Uni	$F_7(x) = 100(x_2 - x_1^3)^2 + (1 - x_1)^2$
Matyas	Uni	$F_8(x) = 0.26(x_1^2 + x_2^2) - 0.48x_1x_2$

Table 1. (continued)

Name	**Type**	**Equation**
Powell Sum	Uni	$F_9(x)=\sum_{i=1}^{d}\lvert x_i\rvert^{i+1}$
Schwefel's Problem 1.2	Uni	$F_{10}(x)=\sum_{i=1}^{d}\left(\sum_{j=1}^{i}x_j\right)^2$
Schwefel's Problem 2.20	Uni	$F_{11}(x)=\sum_{i=1}^{d}\lvert x_i\rvert$
Schwefel's Problem 2.21	Uni	$F_{12}(x)=\max\lvert x_i\rvert,\ 1\le i\le d$
Schwefel's Problem 2.22	Uni	$F_{13}(x)=\sum_{i=1}^{d}\lvert x_i\rvert+\prod_{i=1}^{d}\lvert x_i\rvert$
Schwefel's Problem 2.23	Uni	$F_{14}(x)=\sum_{i=1}^{d}x_i^{10}$
Step	Uni	$F_{15}(x)=\sum_{i=1}^{d}\lfloor\lvert x_i\rvert\rfloor$
Step 2	Uni	$F_{16}(x)=\sum_{i=1}^{d}\lfloor\lvert x_i\rvert+0.5\rfloor$
Sum Squares	Uni	$F_{17}(x)=\sum_{i=1}^{d}ix_i^2$
Trecanni	Uni	$F_{18}(x)=x_1^4-4x_1^3+4x_1+x_2^2$
Wayburn Seader 1	Uni	$F_{19}(x)=\left(x_1^6+x_2^4-17\right)^2+\left(2x_1+x_2-4\right)^2$
Wayburn Seader 2	Uni	$F_{20}(x)=\left[1.613-4\left(x_1-0.3125\right)^2-4\left(x_2-1.625\right)^2\right]^2+\left(x_2-1\right)^2$
Ackley	Multi	$F_{21}(x)=-20\exp\left(-0.2\sqrt{\frac{1}{d}\sum_{i=1}^{d}x_i^2}\right)-\exp(\frac{1}{d}\sum_{i=1}^{d}\cos\left(2\pi x_i\right)) +20+\exp(1)$
Alpine 1	Multi	$F_{22}(x)=\sum_{i=1}^{d}\lvert x_i\sin\left(x_i+0.1x_i\right)\rvert$
Bartels Conn	Multi	$F_{23}(x)=\lvert x_1^2+x_2^2+x_1x_2\rvert+\lvert\sin\left(x_1\right)\rvert+\lvert\cos\left(x_2\right)\rvert$
Beale	Multi	$F_{24}(x)=\left(1.5-x_1+x_1x_2\right)^2+\left(2.25-x_1+x_1x_2^2\right)^2+\left(2.625-x_1+x_1x_2^3\right)^2$
Bohachevsky 1	Multi	$F_{25}(x)=x_1^2+2x_2^2-0.3\cos(3\pi x_1)-0.4\cos(4\pi x_2)+0.7$
Bohachevsky 2	Multi	$F_{26}(x)=x_1^2+2x_2^2-0.3\cos(3\pi x_1)0.4\cos(4\pi x_2)+0.3$
Bohachevsky 3	Multi	$F_{27}(x)=x_1^2+2x_2^2-0.3\cos(3\pi x_1+4\pi x_2)+0.3$
Cross-in-Tray	Multi	$F_{28}(x)=-0.0001\left[\left\lvert\sin\left(x_1\right)\sin\left(x_2\right)e^{\left\lvert 100-\left[\left(x_1^2+x_2^2\right)\right]^{0.5}/\pi\right\rvert}\right\rvert+1\right]^{0.1}$

Table 1. (continued)

Name	Type	Equation
Csendes	Multi	$F_{29}(x)=\sum_{i=1}^{d} x_i^6(2+\sin\frac{1}{x_i})$
Colville	Multi	$F_{30}(x)=100\left(x_1-x_2^2\right)^2+\left(1-x_1\right)^2+90\left(x_4-x_3^2\right)^2+\left(1-x_3\right)^2+10.1\left(\left(x_2-1\right)^2+\left(x_4-1\right)^2\right)$ $+19.8\left(x_2-1\right)\left(x_4-1\right)$
Deckkers-Aarts	Multi	$F_{31}(x)=10^5 x_1^2+x_2^2-\left(x_1^2+x_2^2\right)^2+10^{-5}\left(x_1^2+x_2^2\right)^4$
Griewank	Multi	$F_{32}(x)=\sum_{i=1}^{d}\frac{x_i^2}{4000}-\prod_{i=1}^{d}\cos\left(\frac{x_i}{\sqrt{i}}\right)+1$
Goldstein-Price	Multi	$F_{33}(x)=\left[1+\left(x_1+x_2+1\right)^2\begin{pmatrix}19-14x_1+3x_1{}^2-14x_2\\+6x_1x_2+3x_2{}^2\end{pmatrix}\right].$ $\left[30+(2x_1-3x_2)^2\begin{pmatrix}18-32x_1+12x_1{}^2\\+48x_2-36x_1x_2+27x_2{}^2\end{pmatrix}\right]$
Helical Valley	Multi	$x_1\geq 0\Rightarrow\theta=\frac{\arctan\left(\frac{x_2}{x_1}\right)}{2\pi}$, $x_1<0\Rightarrow\theta=\frac{\arctan\left(\frac{x_2}{x_1}\right)+\pi}{2\pi}$ $F_{34}(x)=100\left[\left(x_3-10\theta\right)^2+(\sqrt{x_1^2+x_2^2}-1)^2\right]+x_3^2$
Himmelblau	Multi	$F_{35}(x)=\left(x_1^2+x_2-11\right)^2+\left(x_1+x_2^2-7\right)^2$
Holder-Table	Multi	$F_{36}(x)=-\left\|\sin\left(x_1\right)\cos\left(x_2\right)e^{\left\|1-\left[\left(x_1^2+x_2^2\right)\right]^{0.5}/\pi\right\|}\right\|$
Keane	Multi	$F_{37}(x)=-\frac{\sin^2\left(x_1-x_2\right)\sin^2\left(x_1+x_2\right)}{\sqrt{x_1^2+x_2^2}}$
Kowalik Problem	Multi	$a=0.1957, 0.1947, 0.1735, 0.16, 0.0844, 0.0627, 0.0456, 0.0342, 0.0323, 0.0235, 0.0246$ $b^{-1}=0.25, 0.5, 1, 2, 4, 6, 8, 10, 12, 14, 16$ $F_{38}(x)=\sum_{i=1}^{11}\left(a_i-(x_1(b_i^2+b_ix_2))/\left(b_i^2+b_ix_3+x_4\right)\right)$
Levy N.13	Multi	$F_{39}(x)=\sin^2\left(3\pi x_1\right)+\left(x_1-1\right)^2\left(1+\sin^2\left(3\pi x_2\right)\right)+\left(x_2-1\right)^2\left(1+\sin^2\left(2\pi x_2\right)\right)$
Pathological	Multi	$F_{40}(x)=\sum_{i=1}^{d-1}\left(0.5+\frac{\sin^2\sqrt{100x_i^2+x_{i+1}^2}-0.5}{1+0.001\left(x_i^2-2x_ix_{i+1}+x_{i+1}^2\right)^2}\right)$
Price 2	Multi	$F_{41}(x)=1+\sin^2 x_1+\sin^2 x_2-0.1e^{-\left(x_1^2+x_2^2\right)}$
Price 3	Multi	$F_{42}(x)=100\left(x_2-x_1^2\right)^2+6\left[6.4\left(x_2-0.5\right)^2-x_1-0.6\right]^2$
Price 4	Multi	$F_{43}(x)=(2x_1^3x_2-x_2^3)^2+\left(6x_1-x_2^2+x_2\right)^2$
Rastrigin	Multi	$F_{44}(x)=10d+\sum_{i=1}^{d}\left[x_i^2-10\cos\cos\left(2\pi x_i\right)\right]$

Table 1. (continued)

Name	Type	Equation
Salomon	Multi	$F_{45}(x)=1-\cos\left(2\pi\sqrt{\sum_{i=1}^{d}x_i^2}\right)+0.1\sqrt{\sum_{i=1}^{d}x_i^2}$
Schaffel N.1	Multi	$F_{46}(x)=0.5+\frac{\sin^2\left(x_1^{\ 2}+x_2^{\ 2}\right)^2-0.5}{\left(1+0.001\left(x_1^{\ 2}+x_2^{\ 2}\right)\right)^2}$
Six Hump Camel	Multi	$F_{47}(x)=\left(4-2.1x_1^2+\frac{x_i^4}{3}\right)x_1^2+x_1x_2+\left(4x_2^2-4\right)x_2^2$
Three Hump Camel	Multi	$F_{48}(x)=2x_1^2-1.05x_1^4+\frac{x_1^6}{6}+x_1x_2+x_2^2$
Weierstrass	Multi	$a=0.5 \quad b=3 \quad kmax=20$ $F_{49}(x)=\sum_{i=1}^{d}\left[\sum_{k=0}^{kmax}a^k\cos\left(2\pi b^k\left(x_i+0.5\right)\right)-d\sum_{k=0}^{kmax}a^k\cos\left(\pi b^k\right)\right]$
Wolfe	Multi	$F_{50}(x)=\frac{4}{3}\left(x_1^{\ 2}+x_2^{\ 2}-x_1x_2\right)^{0.75}+x_3$

MVO (Mirjalili et al., 2016) and six algorithms commonly mentioned in the literature to assess the success of the IMVOSA algorithm: Cuckoo Search (CS) (Yang & Deb, 2009), Differential Evolution (DE) (Storn & Price, 1997), Harris Hawks Optimization (HHO) (Heidari et al., 2019), Moth Flame Algorithm (MFO) (Mirjalili, 2015), Particle Swarm Optimization (PSO) (Kennedy & Eberhart, 1995) and Gravity Search Algorithm (GSA) (Rashedi et al., 2009) were used. The results of the algorithms used to solve the 50 selected benchmark functions were compared using various statistical measurements and methods.

Table 2
Bounds, dimensions and minimum fitness

F_x	Range	D	Fmin	F_x	Range	D	Fmin
F_1	[-32, 32]	2	-200	F_{26}	[-100, 100]	2	0
F_2	[-10, 10]	2	0	F_{27}	[-100, 100]	2	0
F_3	[-20, 0]	2	0	F_{28}	[-10, 10]	2	−2.06261218
F_4	[-1, 4]	20	0	F_{29}	[-1, 1]	20	0
F_5	[-10, 10]	20	0	F_{30}	[-10, 10]	4	0
F_6	[-5.2, 5.2]	2	-1	F_{31}	[-20, 20]	2	−24777
F_7	[0, 10]	2	0	F_{32}	[-600, 600]	20	0
F_8	[-10, 10]	2	0	F_{33}	[-2, 2]	2	3
F_9	[-1, 1]	20	0	F_{34}	[-10, 10]	3	0
F_{10}	[-100, 100]	20	0	F_{35}	[-6, 6]	2	0
F_{11}	[-100, 100]	20	0	F_{36}	[-10, 10]	2	−19.2085
F_{12}	[-100, 100]	20	0	F_{37}	[-10, 10]	2	−0.673668
F_{13}	[-100, 100	20	0	F_{38}	[-5, 5]	4	0.0003075
F_{14}	[-10, 10]	20	0	F_{39}	[-10, 10]	2	0
F_{15}	[-100, 100]	20	0	F_{40}	[-100, 100]	20	0
F_{16}	[-100, 100]	20	0	F_{41}	[-10, 10]	2	0
F_{17}	[-10, 10]	20	0	F_{42}	[-500, 500]	2	0
F_{18}	[-5, 5]	2	0	F_{43}	[-500, 500]	2	0
F_{19}	[-5, 5]	2	0	F_{44}	[-5.12, 5.12]	20	0
F_{20}	[-500, 500]	2	0	F_{45}	[-100, 100]	20	0
F_{21}	[-32, 32]	20	0	F_{46}	[-100, 100]	2	0
F_{22}	[-10, 10]	20	0	F_{47}	[-5, 5]	2	-1.0316
F_{23}	[-500, 500]	2	1	F_{48}	[-5, 5]	2	0
F_{24}	[-4.5, 4.5]	2	0	F_{49}	[-0.5, 0.5]	20	0
F_{25}	[-100, 100]	2	0	F_{50}	[0, 2]	3	0

4.2. Experimental setup

Experiments were carried out using the EvoloPy framework. Evolopy is a framework consisting of classical and up-to-date metaheuristic algorithms written in Python language with an easy-to-use interface. Studies have been conducted on optimization, clustering, feature selection and artificial neural network training using Evolopy (Mirjalili et al., 2017; Khurma et al., 2020; Qaddoura et al., 2021). The suggested algorithm and the others are coded in Python 3.8 language and have been tested on a personal computer with Intel (R) Core (TM) i7-10875H CPU 2.30 GHz, 64 Bit Windows 10 operating system and 16 GB (RAM). The parameter settings for the algorithms used in the comparison are shown in Table 3. For all algorithms and benchmark functions used the number of search agents is set to 30, the number of iterations is set to 100. Each algorithm has been run independently 30 times to achieve balanced performance consequences. Values are normalized in the range [0,1] for better analysis of consequences (Mirjalili et al., 2017). At the stage of evaluating the performance of algorithms, the average value (Ave) and standard deviation (Std) of the values obtained as a result of each experiment were calculated for each function. In addition, non-parametric Wilcoxon rank-sum test was implemented to the results to investigate the relationship among algorithms; p value was considered less than 0.05 (5E-02) in order to evaluate whether there was a statistically distinctive difference. Table 5 contains the results from the Wilcoxon rank-sum test.

Table 3
Parameters and values of IMVOSA and other algorithms (continued overleaf)

Algorithm	Parameter	Description	Value
IMVOSA	*WEP_Max*	Maximum WEP	1
	WEP_Min	Minimum WEP	0.2
	p	Exploitation accuracy	6
	T	Initial temperature	1
	a	Cooling rate	0.88
MVO	*WEP_Max*	Maximum WEP	1
	WEP_Min	Minimum WEP	0.2
	p	Exploitation accuracy	6
CS	p_a	Discovery Rate	0.25
	β	Beta	1.5
DE	mutation_factor	Mutation factor	0.5
	crossover_ratio	Crossover Ratio	0.7
PSO	*Vmax*	Maximum particle velocity	6
	c1	Acceleration coefficient1	2
	c2	Acceleration coefficient2	2
	wMax	Maximum inertia weight coefficient	0.9
	wMin	Minimum inertia weight coefficient	0.2
HHO		*No custom parameters*	
MFO	*b*	Logaritmik spiral	1
GSA	G_0	Gravitational constant	20
	a	User-specified constant	*100*

4.3. Experimental Results and Discussions

In this part, the success of IMVOSA and other metaheuristic algorithms was analyzed and compared based on average and standard deviation values for 30 trials. In addition, the relationship among algorithms using the results obtained from algorithms was investigated by applying Wilcoxon statistical test.

Table 4
Statistical results for 50 benchmark functions (continued overleaves)

F_x		IMVOSA	MVO	CS	DE	HHO	MFO	PSO	GSA
F_1	Ave	0.00E+00	1.00E+00	8.93E-02	8.13E-09	2.31E-09	3.05E-12	5.87E-06	1.54E-07
	Std	0.00E+00	1.00E+00	1.17E-01	1.94E-08	1.75E-08	5.03E-12	1.58E-05	1.41E-07
F_2	Ave	7.32E-03	4.60E-04	2.09E-05	6.60E-15	1.39E-01	0.00E+00	1.45E-11	1.00E+00
	Std	2.82E-03	1.30E-04	1.07E-05	4.52E-15	1.31E-01	0.00E+00	1.24E-11	1.00E+00
F_3	Ave	5.80E-04	4.96E-04	3.96E-06	1.81E-20	1.00E+00	0.00E+00	1.56E-13	5.95E-17
	Std	2.63E-04	1.46E-04	3.29E-06	2.32E-20	1.00E+00	0.00E+00	1.82E-13	2.48E-17
F_4	Ave	0.00E+00	8.80E-04	8.19E-01	7.41E-03	9.44E-23	8.21E-02	1.00E+00	5.16E-01
	Std	0.00E+00	5.06E-04	4.85E-01	4.44E-03	3.00E-22	8.34E-02	1.00E+00	6.68E-01
F_5	Ave	3.17E-05	1.28E-03	2.49E-01	5.90E-03	0.00E+00	1.00E+00	5.79E-03	1.91E-02
	Std	0.00E+00	3.57E-04	4.73E-02	1.18E-03	2.96E-07	1.00E+00	2.83E-03	1.01E-02
F_6	Ave	0.00E+00	2.16E-01	5.17E-01	7.59E-02	0.00E+00	4.27E-01	2.73E-01	1.00E+00
	Std	0.00E+00	7.67E-01	7.86E-01	4.07E-01	0.00E+00	1.00E+00	8.40E-01	8.69E-01
F_7	Ave	8.99E-02	2.65E-01	0.00E+00	2.12E-01	5.94E-02	2.03E-01	1.00E+00	4.58E-01
	Std	1.94E-01	7.44E-01	0.00E+00	5.20E-01	2.62E-01	3.75E-01	1.00E+00	4.39E-01
F_8	Ave	0.00E+00	3.25E-05	7.69E-07	1.85E-14	2.56E-23	2.20E-14	2.12E-10	1.00E+00
	Std	0.00E+00	2.60E-05	9.40E-07	2.83E-14	9.66E-23	8.82E-14	6.19E-10	1.00E+00

Table 4. (continued)

F_x		IMVOSA	MVO	CS	DE	HHO	MFO	PSO	GSA
F_9	Ave	0.00E+00	1.42E-02	2.07E-01	3.24E-03	2.51E-12	1.77E-02	1.56E-02	1.00E+00
	Std	0.00E+00	2.44E-03	3.71E-02	7.88E-04	2.92E-12	7.22E-03	4.33E-03	1.00E+00
F_{10}	Ave	0.00E+00	4.94E-02	4.61E-01	1.00E+00	3.62E-14	7.35E-01	8.80E-03	1.50E-01
	Std	0.00E+00	7.58E-02	3.61E-01	7.64E-01	6.17E-13	1.00E+00	1.46E-02	2.04E-01
F_{11}	Ave	0.00E+00	7.31E-02	1.00E+00	1.77E-01	1.38E-11	3.32E-01	1.18E-02	7.94E-01
	Std	0.00E+00	1.16E-01	7.28E-01	2.13E-01	2.19E-10	1.00E+00	1.93E-02	9.64E-01
F_{12}	Ave	0.00E+00	9.12E-02	8.03E-01	7.34E-01	3.25E-12	1.00E+00	4.09E-02	3.72E-01
	Std	0.00E+00	2.62E-01	5.15E-01	7.58E-01	3.83E-11	1.00E+00	5.61E-02	3.19E-01
F_{13}	Ave	0.00E+00	1.00E+00	2.81E-01	8.10E-15	1.48E-25	2.56E-14	3.23E-15	1.49E-08
	Std	0.00E+00	1.00E+00	4.04E-01	1.07E-15	9.64E-26	4.98E-15	1.28E-15	2.73E-08
F_{14}	Ave	0.00E+00	3.67E-11	5.99E-01	1.20E-03	1.51E-90	1.00E+00	1.46E-05	2.83E-02
	Std	0.00E+00	4.29E-11	3.95E-01	7.10E-04	4.42E-90	1.00E+00	1.80E-05	4.37E-02
F_{15}	Ave	0.00E+00	7.97E-02	1.00E+00	1.80E-01	5.06E-11	2.97E-01	1.09E-02	8.25E-01
	Std	0.00E+00	1.74E-01	1.00E+00	2.82E-01	9.27E-10	9.65E-01	2.85E-02	9.26E-01
F_{16}	Ave	1.19E-03	2.88E-03	1.00E+00	4.54E-02	0.00E+00	1.84E-01	1.92E-04	5.92E-01
	Std	4.10E-04	4.81E-03	1.00E+00	7.21E-02	0.00E+00	3.63E-01	4.75E-04	6.26E-01
F_{17}	Ave	0.00E+00	1.34E-02	1.00E+00	3.97E-02	2.90E-22	4.63E-01	7.66E-02	1.18E-01
	Std	0.00E+00	1.05E-02	3.19E-01	2.03E-02	1.38E-21	1.00E+00	2.23E-01	1.02E-01
F_{18}	Ave	9.49E-14	2.70E-04	1.50E-05	3.53E-13	4.60E-06	0.00E+00	2.33E-12	1.00E+00
	Std	0.00E+00	5.16E-05	4.09E-06	2.36E-13	3.39E-06	2.35E-14	1.48E-12	1.00E+00
F_{19}	Ave	8.47E-03	1.52E-04	7.77E-04	1.26E-08	4.05E-02	8.58E-07	0.00E+00	1.00E+00
	Std	7.51E-03	1.00E-04	6.85E-04	3.30E-08	2.60E-02	1.89E-06	0.00E+00	1.00E+00
F_{20}	Ave	1.40E-08	4.65E-09	8.79E-11	0.00E+00	7.63E-12	4.05E-15	1.22E-13	1.00E+00
	Std	7.17E-09	2.05E-09	3.51E-11	0.00E+00	7.05E-12	4.20E-15	1.70E-13	1.00E+00
F_{21}	Ave	0.00E+00	1.73E-01	1.00E+00	4.28E-01	4.04E-11	8.73E-01	1.22E-01	3.62E-01
	Std	0.00E+00	9.03E-02	2.28E-01	5.77E-01	2.97E-10	1.00E+00	1.10E-01	2.64E-01
F_{22}	Ave	0.00E+00	4.91E-01	1.00E+00	8.09E-01	6.95E-07	3.46E-01	1.46E-01	1.57E-01
	Std	0.00E+00	6.87E-01	4.24E-01	4.79E-01	1.45E-05	1.00E+00	4.39E-01	4.76E-01
F_{23}	Ave	0.00E+00	2.36E-04	8.63E-06	1.03E-14	4.24E-14	0.00E+00	3.15E-10	1.00E+00
	Std	0.00E+00	1.43E-04	5.55E-06	9.41E-15	1.26E-13	0.00E+00	4.13E-10	1.00E+00
F_{24}	Ave	1.00E+00	6.25E-01	5.85E-04	0.00E+00	1.56E-01	1.03E-11	2.36E-09	4.01E-01
	Std	1.00E+00	7.99E-01	6.04E-04	0.00E+00	4.22E-01	2.43E-11	4.09E-09	2.98E-01
F_{25}	Ave	0.00E+00	5.34E-01	2.18E-02	0.00E+00	0.00E+00	0.00E+00	9.41E-12	1.00E+00
	Std	0.00E+00	1.92E-01	1.06E-02	0.00E+00	0.00E+00	0.00E+00	8.23E-12	1.00E+00
F_{26}	Ave	0.00E+00	8.67E-02	2.56E-03	5.07E-16	1.91E-17	0.00E+00	8.10E-13	1.00E+00
	Std	0.00E+00	2.57E-02	1.11E-03	4.32E-16	2.06E-17	0.00E+00	5.08E-13	1.00E+00
F_{27}	Ave	0.00E+00	5.60E-02	2.24E-03	3.38E-12	2.68E-14	6.87E-07	1.39E-09	1.00E+00
	Std	0.00E+00	1.67E-02	8.11E-04	3.30E-12	4.43E-14	1.18E-06	1.16E-09	1.00E+00
F_{28}	Ave	1.00E+00	6.01E-01	2.03E-02	1.00E-05	6.72E-02	0.00E+00	9.49E-08	0.00E+00
	Std	1.00E+00	4.16E-01	1.87E-02	2.56E-05	1.02E-01	0.00E+00	2.69E-07	0.00E+00
F_{29}	Ave	2.28E-11	4.47E-09	2.59E-02	2.75E-04	0.00E+00	5.86E-03	3.25E-03	1.00E+00
	Std	1.61E-11	2.25E-09	5.60E-03	1.23E-04	0.00E+00	3.13E-03	1.56E-03	1.00E+00
F_{30}	Ave	2.41E-02	1.89E-02	1.78E-02	2.84E-03	0.00E+00	1.54E-02	4.35E-03	1.00E+00
	Std	6.82E-03	6.10E-03	1.19E-03	2.96E-05	0.00E+00	6.26E-03	5.94E-04	1.00E+00
F_{31}	Ave	2.15E-06	4.55E-05	3.40E-06	1.05E-14	1.55E-01	0.00E+00	3.06E-15	1.00E+00
	Std	5.16E-06	7.69E-05	6.37E-06	4.15E-14	7.73E-01	0.00E+00	1.36E-14	1.00E+00
F_{32}	Ave	0.00E+00	5.20E-03	1.17E-01	1.03E-02	5.50E-20	2.43E-02	4.71E-02	1.00E+00
	Std	0.00E+00	8.01E-04	1.75E-01	1.50E-02	1.50E-18	6.49E-02	1.25E-01	1.00E+00
F_{33}	Ave	1.00E+00	3.33E-01	1.57E-05	0.00E+00	7.78E-01	8.22E-16	9.22E-14	1.57E-01
	Std	1.00E+00	5.98E-01	8.97E-06	3.24E-16	4.70E-01	0.00E+00	6.73E-14	2.19E-01
F_{34}	Ave	1.31E-01	4.24E-03	1.19E-02	0.00E+00	6.15E-02	1.01E-01	1.45E-03	1.00E+00
	Std	4.11E-01	1.20E-02	1.62E-02	0.00E+00	9.94E-02	3.96E-01	4.89E-03	1.00E+00
F_{35}	Ave	8.30E-04	6.11E-04	7.53E-05	6.12E-08	2.47E-03	0.00E+00	2.15E-12	1.00E+00
	Std	1.84E-04	1.34E-04	1.84E-05	7.18E-08	2.01E-03	0.00E+00	1.13E-12	1.00E+00

Table 4. (continued)

F_x		IMVOSA	MVO	CS	DE	HHO	MFO	PSO	GSA
F_{36}	Ave	7.48E-06	5.93E-06	1.00E+00	8.64E-03	1.11E-02	0.00E+00	1.35E-02	3.64E-02
	Std	6.56E-09	4.27E-09	1.00E+00	1.48E-05	2.79E-05	0.00E+00	1.81E-05	1.85E-05
F_{37}	Ave	4.20E-01	1.00E+00	1.54E-01	1.60E-02	3.63E-02	0.00E+00	9.83E-08	0.00E+00
	Std	1.00E+00	9.76E-01	2.72E-01	6.18E-02	7.34E-02	0.00E+00	3.02E-07	0.00E+00
F_{38}	Ave	2.51E-01	4.63E-01	3.51E-02	1.33E-02	0.00E+00	3.20E-02	3.76E-01	1.00E+00
	Std	5.60E-01	1.00E+00	2.30E-03	7.37E-03	8.86E-03	0.00E+00	6.57E-01	8.69E-01
F_{39}	Ave	5.36E-01	1.58E-01	4.19E-02	3.09E-16	2.67E-02	0.00E+00	1.47E-10	1.00E+00
	Std	4.11E-01	6.55E-02	2.93E-02	3.29E-16	2.03E-02	0.00E+00	1.08E-10	1.00E+00
F_{40}	Ave	0.00E+00	9.95E-01	1.00E+00	4.15E-01	1.51E-04	4.00E-01	8.81E-01	5.45E-01
	Std	0.00E+00	5.12E-01	4.14E-01	9.05E-01	6.85E-03	7.78E-01	8.07E-01	1.00E+00
F_{41}	Ave	1.03E-01	1.98E-01	9.24E-02	0.00E+00	7.67E-02	1.80E-01	1.47E-01	1.00E+00
	Std	4.96E-02	5.67E-03	3.80E-02	4.76E-02	5.24E-02	0.00E+00	3.17E-02	1.00E+00
F_{42}	Ave	8.29E-09	4.79E-09	1.49E-10	8.57E-11	0.00E+00	5.98E-11	8.57E-11	1.00E+00
	Std	2.69E-09	1.48E-09	3.54E-11	2.13E-11	0.00E+00	1.86E-11	2.14E-11	1.00E+00
F_{43}	Ave	0.00E+00	1.00E+00	9.99E-12	1.86E-20	9.35E-18	8.28E-16	4.55E-15	9.81E-02
	Std	0.00E+00	1.00E+00	3.48E-12	5.53E-21	6.01E-18	6.06E-16	1.93E-15	5.04E-02
F_{44}	Ave	0.00E+00	6.39E-01	9.38E-01	1.00E+00	0.00E+00	7.61E-01	7.70E-01	3.83E-01
	Std	0.00E+00	7.44E-01	4.08E-01	4.06E-01	0.00E+00	1.00E+00	8.59E-01	6.43E-01
F_{45}	Ave	4.49E-04	1.42E-01	1.00E+00	3.64E-01	0.00E+00	5.67E-01	7.73E-02	5.38E-01
	Std	1.24E-02	1.61E-01	6.61E-01	3.04E-01	0.00E+00	1.00E+00	7.50E-02	4.77E-01
F_{46}	Ave	0.00E+00	1.23E-05	1.54E-02	7.99E-06	0.00E+00	0.00E+00	7.16E-15	1.00E+00
	Std	0.00E+00	9.02E-06	3.86E-02	3.33E-05	0.00E+00	0.00E+00	3.09E-14	1.00E+00
F_{47}	Ave	7.94E-02	7.53E-02	3.44E-03	4.81E-07	4.98E-03	0.00E+00	8.08E-10	1.00E+00
	Std	2.48E-02	1.27E-02	8.83E-04	3.52E-07	1.96E-03	0.00E+00	6.44E-10	1.00E+00
F_{48}	Ave	0.00E+00	1.19E-05	8.38E-07	8.52E-20	3.19E-20	3.54E-28	2.27E-14	1.00E+00
	Std	0.00E+00	7.68E-06	8.03E-07	1.27E-19	5.12E-20	6.27E-28	2.15E-14	1.00E+00
F_{49}	Ave	0.00E+00	5.87E-01	1.00E+00	3.53E-01	2.18E-08	4.19E-01	4.82E-01	3.51E-01
	Std	0.00E+00	1.00E+00	5.34E-01	3.46E-01	8.63E-07	7.03E-01	8.10E-01	9.18E-01
F_{50}	Ave	0.00E+00	0.00E+00	1.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00
	Std	0.00E+00	0.00E+00	1.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00

Statistical results obtained from 50 comparison functions using IMVOSA, MVO, CS, DE, HHO, MFO, PSO and GSA algorithms are shown in Table 4. As seen in this table, IMVOSA performs better than other algorithms for most functions in unimodal and multimodal groups.

In the unimodal function group, the best results were obtained in 12 functions from 20 functions including F_1, F_4, F_6, $F_8 - F_{15}$, F_{17} with the IMVOSA algorithm. Other best results are MFO algorithm 3 functions, HHO algorithm 3 functions, CS Algorithm 1 function, DE Algorithm 1 function, PSO Algorithm 1 function respectively. With the GSA and MVO algorithms, the best results were not obtained in any of the 20 test functions in the unimodal function group.

With the IMVOSA algorithm, the best results were obtained in 14 functions from 30 functions in the multimodal function group, including $F_{21} - F_{23}$, $F_{25} - F_{27}$, F_{32}, F_{40}, F_{43}, F_{44}, F_{46}, F_{48} - F_{50}. Other best results are 10 functions with MFO algorithm, 8 functions with HHO algorithm, 5 functions with DE algorithm, 3 functions with GSA Algorithm, 1 function with PSO Algorithm, 1 function with CEO algorithm, respectively. The best results were not obtained in any of the 30 test functions in the multimodal function group with the CS algorithm. Fig. 2 shows convergence curves for some functions from the unimodal and multimodal function group to observe the convergence attitude of the IMVOSA algorithm and to better compare the success of the IMVOSA algorithm and other algorithms. As shown in Fig. 2, the proposed IMVOSA algorithm has achieved successful consequences by outperforming other algorithms with regard to performance and stability. The nonparametric Wilcoxon rank sum statistical test was applied to understand the relationship among the MVO, CS, DE, HHO, MFO, PSO, and GSA algorithms used in comparison with the IMVOSA algorithm. This test is used to verify whether solutions are statistically different (Wilcoxon, 1992). The fact that the p value for this test is less than 0.05 (5E-02) identifies a statistically significant difference. Consequences for this test are given in Table 5. In the table, the algorithm with the best average value is compared with other algorithms and gets the N/A value. In cases where there are algorithms or algorithms with the same score as the IMVOSA algorithm for any function, the IMVOSA algorithm was chosen as the best algorithm. According to Table 5, the IMVOSA algorithm performs better than other algorithms. The IMVOSA algorithm has a p value smaller than 0.05 for many benchmark functions and can be concluded to be statistically significant. From these results, it can be said that IMVOSA has a higher ability of exploration and exploitation than MVO and other algorithms.

Table 5
Wilcoxon rank sum test p-values (N/A = not applicable) (continued overleaf)

F_x	IMVOSA	MVO	CS	DE	HHO	MFO	PSO	GSA
F_1	N/A	2.87E-11	2.87E-11	2.87E-11	1.27E-10	2.13E-09	2.87E-11	2.87E-11
F_2	2.87E-11	2.87E-11	2.87E-11	4.73E-11	2.87E-11	N/A	2.87E-11	3.88E-11
F_3	2.87E-11	2.87E-11	2.87E-11	2.87E-11	2.87E-11	N/A	2.87E-11	2.87E-11
F_4	N/A	2.87E-11	2.87E-11	2.87E-11	2.87E-11	2.87E-11	2.87E-11	2.87E-11
F_5	2.87E-11	2.87E-11	2.87E-11	2.87E-11	N/A	2.87E-11	2.87E-11	2.87E-11
F_6	N/A	2.87E-11	2.87E-11	2.87E-11	1.00E+00	7.60E-02	2.87E-11	1.27E-10
F_7	5.12E-04	4.16E-01	N/A	4.25E-01	1.28E-01	2.23E-06	8.50E-03	3.31E-10
F_8	N/A	2.87E-11	2.87E-11	2.87E-11	1.02E-07	2.87E-11	2.87E-11	2.87E-11
F_9	N/A	4.24E-11	4.24E-11	4.24E-11	4.24E-11	4.24E-11	4.24E-11	4.24E-11
F_{10}	N/A	2.87E-11	2.87E-11	2.87E-11	2.87E-11	2.87E-11	2.87E-11	2.87E-11
F_{11}	N/A	2.87E-11	2.87E-11	2.87E-11	2.87E-11	2.87E-11	2.87E-11	2.87E-11
F_{12}	N/A	2.87E-11	2.87E-11	2.87E-11	2.87E-11	2.87E-11	2.87E-11	2.87E-11
F_{13}	N/A	2.87E-11	2.87E-11	2.87E-11	2.87E-11	2.87E-11	2.87E-11	2.87E-11
F_{14}	N/A	2.87E-11	2.87E-11	2.87E-11	1.00E+00	2.87E-11	2.87E-11	2.87E-11
F_{15}	N/A	2.87E-11	2.87E-11	2.87E-11	2.87E-11	2.87E-11	2.87E-11	2.87E-11
F_{16}	2.87E-11	2.87E-11	2.87E-11	2.87E-11	N/A	2.87E-11	2.87E-11	2.87E-11
F_{17}	N/A	2.87E-11	2.87E-11	2.87E-11	2.87E-11	2.87E-11	2.87E-11	2.87E-11
F_{18}	6.57E-01	2.87E-11	2.87E-11	1.72E-03	2.71E-08	N/A	7.73E-10	1.72E-03
F_{19}	2.87E-11	2.87E-11	3.51E-11	1.33E-06	3.51E-11	3.21E-06	N/A	2.87E-11
F_{20}	2.87E-11	2.87E-11	2.87E-11	N/A	5.23E-11	4.25E-01	2.89E-07	2.87E-11
F_{21}	N/A	2.87E-11	2.87E-11	2.87E-11	2.87E-11	2.87E-11	2.87E-11	2.87E-11
F_{22}	N/A	2.87E-11	2.87E-11	2.87E-11	2.87E-11	2.87E-11	2.87E-11	2.87E-11
F_{23}	N/A	2.87E-11	2.87E-11	2.87E-11	1.47E-02	1.00E+00	2.87E-11	2.87E-11
F_{24}	2.87E-11	2.87E-11	2.87E-11	N/A	1.94E-09	5.12E-04	2.87E-11	2.87E-11
F_{25}	N/A	2.87E-11	2.87E-11	1.00E+00	1.00E+00	1.00E+00	2.87E-11	1.83E-01
F_{26}	N/A	2.87E-11	2.87E-11	4.59E-02	6.57E-01	1.00E+00	2.87E-11	9.19E-06
F_{27}	N/A	2.87E-11	2.87E-11	1.27E-10	2.68E-01	2.66E-02	2.87E-11	2.87E-11
F_{28}	2.87E-11	2.87E-11	2.87E-11	1.47E-02	2.87E-11	N/A	7.60E-02	1.00E+00
F_{29}	2.87E-11	2.87E-11	2.87E-11	2.87E-11	N/A	2.87E-11	2.87E-11	2.87E-11
F_{30}	4.01E-10	1.62E-09	4.84E-10	6.80E-08	N/A	1.93E-08	2.49E-08	3.88E-11
F_{31}	2.87E-11	2.87E-11	2.87E-11	2.68E-01	2.87E-11	N/A	2.68E-01	2.87E-11
F_{32}	N/A	2.87E-11	2.87E-11	2.87E-11	6.57E-01	2.87E-11	2.87E-11	2.87E-11
F_{33}	2.87E-11	2.87E-11	2.87E-11	N/A	3.02E-11	3.94E-03	4.29E-11	1.01E-08
F_{34}	1.62E-09	3.18E-11	2.87E-11	N/A	3.18E-11	2.05E-10	8.49E-10	2.87E-11
F_{35}	2.87E-11	2.87E-11	2.87E-11	2.87E-11	2.87E-11	N/A	2.87E-11	2.87E-11
F_{36}	2.87E-11	2.87E-11	2.87E-11	1.90E-03	2.87E-11	N/A	4.59E-02	2.13E-09
F_{37}	2.87E-11	2.87E-11	2.87E-11	2.95E-08	2.87E-11	N/A	1.02E-07	1.00E+00
F_{38}	4.00E-09	2.79E-09	1.63E-08	1.02E-07	N/A	2.71E-08	2.33E-09	3.18E-11
F_{39}	2.87E-11	2.87E-11	2.87E-11	2.87E-11	2.87E-11	N/A	2.87E-11	2.87E-11
F_{40}	N/A	2.87E-11	2.87E-11	2.87E-11	1.27E-10	2.87E-11	2.87E-11	2.87E-11
F_{41}	5.20E-03	3.01E-10	1.05E-02	N/A	1.93E-01	7.79E-03	5.65E-02	1.86E-10
F_{42}	3.88E-11	3.51E-11	4.84E-10	5.95E-01	N/A	1.27E-03	1.04E-01	2.87E-11
F_{43}	N/A	2.87E-11	2.87E-11	2.87E-11	1.07E-06	2.87E-11	2.87E-11	2.87E-11
F_{44}	N/A	2.87E-11	2.87E-11	2.87E-11	1.00E+00	2.87E-11	2.87E-11	2.87E-11
F_{45}	5.32E-10	2.87E-11	2.87E-11	2.87E-11	N/A	2.87E-11	2.87E-11	2.87E-11
F_{46}	N/A	2.87E-11	2.87E-11	2.87E-11	1.00E+00	1.00E+00	2.68E-01	2.87E-11
F_{47}	2.87E-11	2.87E-11	2.87E-11	7.60E-02	2.87E-11	N/A	7.79E-03	8.24E-01
F_{48}	N/A	2.87E-11	2.87E-11	2.87E-11	1.27E-10	1.63E-04	2.87E-11	2.87E-11
F_{49}	N/A	2.87E-11	2.87E-11	2.87E-11	5.32E-10	2.87E-11	2.87E-11	2.87E-11
F_{50}	N/A	1.00E+00	2.87E-11	1.00E+00	1.00E+00	1.00E+00	1.00E+00	1.00E+00

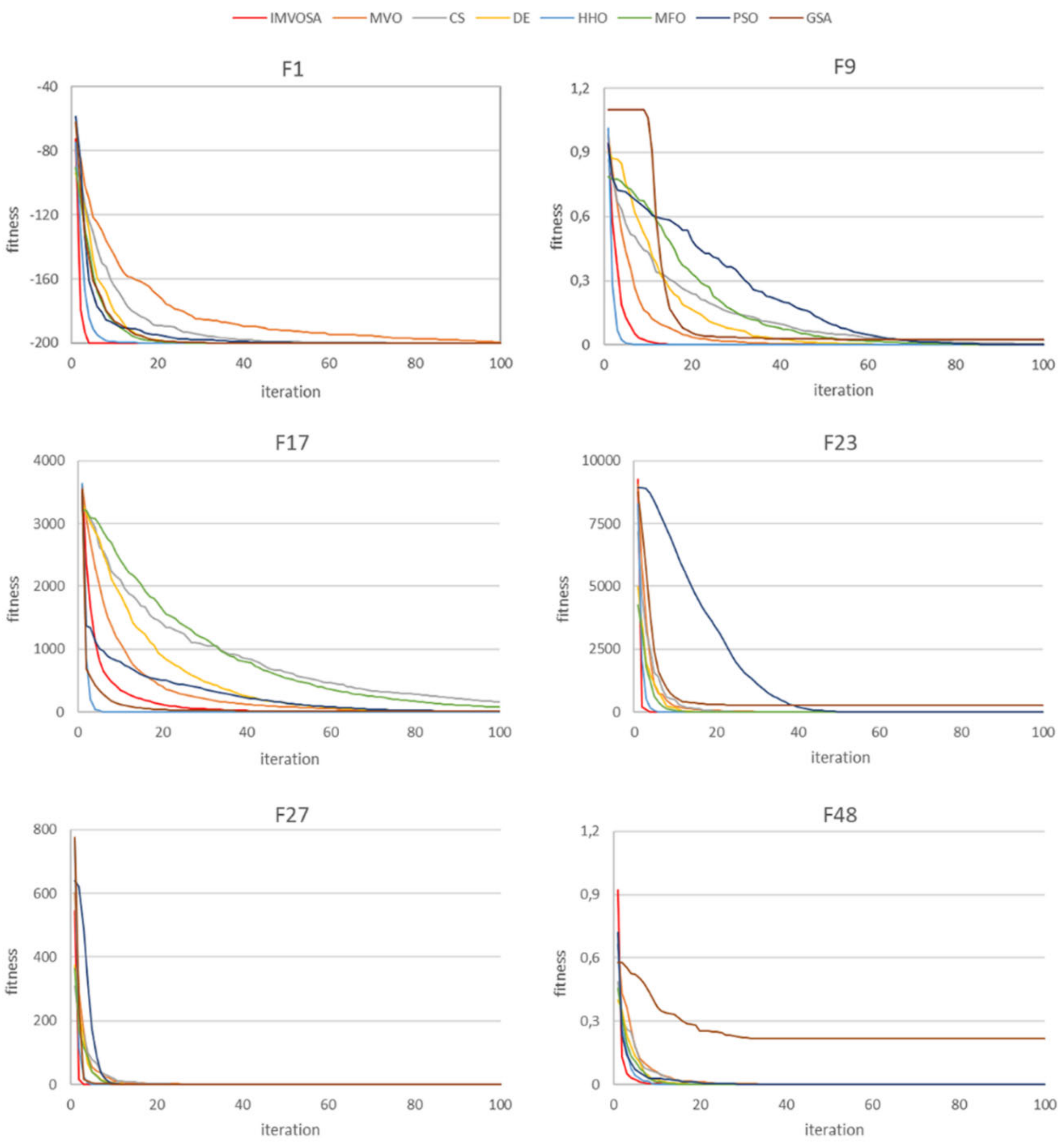

Fig. 2. Convergence curves of six benchmark functions (F1, F9, F17, F23, F27, F48)

5. Conclusion remarks

In metaheuristic algorithms, the aim is to reach the best result in current conditions. Algorithms must have advanced exploration and exploitation abilities to achieve the best results. The balance between exploration and exploitation is significantly influencing the performance of the algorithm. Algorithms can be combined to improve exploration and exploitation capabilities. Combining a metaheuristic algorithm with at least one algorithm using different and advantageous aspects is expressed as a hybrid metaheuristic algorithm.

This article introduces a new hybrid algorithm using MVO and SA algorithms. This new algorithm has been developed using the strengths of MVO and SA. Our main purpose in this study is to combine MVO and SA algorithms with a new hybrid method to increase success for function optimization. The success of the IMVOSA algorithm was tested on 50 benchmark functions and the consequences were compared with the consequences of 7 other metaheuristic algorithms, including MVO algorithm. Comparison consequences show that the IMVOSA algorithm importantly improves the success of MVO algorithm and performs better than other comparison algorithms. In the future, we intend to apply the proposed IMVOSA algorithm for real engineering problems and training of artificial neural networks.

References

Abasi, A. K., Khader, A. T., Al-Betar, M. A., Naim, S., Alyasseri, Z. A. A., & Makhadmeh, S. N. (2020). A novel hybrid multi-verse optimizer with K-means for text documents clustering. *Neural Comput. Appl.*, *32*(23), 17703-17729.

Abdel-Basset, M., Ding, W., & El-Shahat, D. (2021). A hybrid Harris Hawks optimization algorithm with simulated annealing for feature selection. *Artificial Intelligence Review*, *54*(1), 593-637.

Alizada, B. (2019). *Hybridization of swarm-based ant lion and whale optimization algorithms with physics-based algorithms*. (Master's thesis, Erciyes University). Erciyes University Research Information System. https://avesis.erciyes.edu.tr/file?id=8a77bd62-5fea-44cb-a5f3-d61118c8a9ab

Alkhateeb, F., & Abed-Alguni, B. H. (2019). A hybrid cuckoo search and simulated annealing algorithm. *Journal of Intelligent Systems*, *28*(4), 683-698.

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

Chen, L., Li, L., & Kuang, W. (2021). A hybrid multiverse optimisation algorithm based on differential evolution and adaptive mutation. *Journal of Experimental & Theoretical Artificial Intelligence*, *33*(2), 239-261.

Chu, S. C., Tsai, P. W., & Pan, J. S. (2006, August). Cat swarm optimization. In *Pacific Rim international conference on artificial intelligence* (pp. 854-858). Springer, Berlin, Heidelberg.

Colorni, A., Dorigo, M., & Maniezzo, V. (1991, December). Distributed optimization by ant colonies. In *Proceedings of the first European conference on artificial life* (Vol. 142, pp. 134-142).

Dupanloup, I., Schneider, S., & Excoffier, L. (2002). A simulated annealing approach to define the genetic structure of populations. *Molecular ecology*, *11*(12), 2571-2581.

Eglese, R. W. (1990). Simulated annealing: a tool for operational research. *European journal of operational research*, *46*(3), 271-281.

Faris, H., Aljarah, I., Mirjalili, S., Castillo, P. A., & Guervós, J. J. M. (2016, November). EvoloPy: An Open-source Nature-inspired Optimization Framework in Python. In *IJCCI (ECTA)* (pp. 171-177).

Hawking, S. W. (1988). *The Illustrated A Brief History of Time: Updated and Expanded Edition*. Bantam Dell Publishing Group.

Heidari, A. A., Mirjalili, S., Faris, H., Aljarah, I., Mafarja, M., & Chen, H. (2019). Harris hawks optimization: Algorithm and applications. *Future generation computer systems*, *97*, 849-872.

Henderson, D., Jacobson, S. H., & Johnson, A. W. (2003). The theory and practice of simulated annealing. In *Handbook of metaheuristics* (pp. 287-319). Springer, Boston, MA.

Jangir, P., Parmar, S. A., Trivedi, I. N., & Bhesdadiya, R. H. (2017). A novel hybrid particle swarm optimizer with multi verse optimizer for global numerical optimization and optimal reactive power dispatch problem. *Engineering Science and Technology, an International Journal*, *20*(2), 570-586.

Jia, H., Peng, X., Song, W., Lang, C., Xing, Z., & Sun, K. (2019). Hybrid multiverse optimization algorithm with gravitational search algorithm for multithreshold color image segmentation. *IEEE Access*, *7*, 44903-44927.

Jovanovic, R., & Tuba, M. (2013). Ant colony optimization algorithm with pheromone correction strategy for the minimum connected dominating set problem. *Computer Science and Information Systems*, *10*(1), 133-149.

Karaboga, D. (2005). *An idea based on honey bee swarm for numerical optimization* (Vol. 200, pp. 1-10). Technical report-tr06, Erciyes university, engineering faculty, computer engineering department.

Serkan, K. A. Y. A., & FIĞLALI, N. (2018). Çok amaçlı esnek atölye tipi çizelgeleme problemlerinin çözümünde meta sezgisel yöntemlerin kullanımı. *Harran Üniversitesi Mühendislik Dergisi*, *3*(3), 222-233.

Kennedy, J., & Eberhart, R. (1995, November). Particle swarm optimization. In *Proceedings of ICNN'95-international conference on neural networks* (Vol. 4, pp. 1942-1948). IEEE.

Khurma, R. A., Aljarah, I., Sharieh, A., & Mirjalili, S. (2020). Evolopy-fs: An open-source nature-inspired optimization framework in python for feature selection. In *Evolutionary Machine Learning Techniques* (pp. 131-173). Springer, Singapore.

Kirkpatrick, S., Gelatt, C. D., & Vecchi, M. P. (1983). Optimization by simulated annealing. *science*, *220*(4598), 671-680.

Komossa, S. (2015). Tidal disruption of stars by supermassive black holes: Status of observations. *Journal of High Energy Astrophysics*, *7*, 148-157.

Laporte, G., Gendreau, M., Potvin, J. Y., & Semet, F. (2000). Classical and modern heuristics for the vehicle routing problem. *International transactions in operational research*, *7*(4-5), 285-300.

Li, X., Zhang, J., & Yin, M. (2014). Animal migration optimization: an optimization algorithm inspired by animal migration behavior. *Neural Computing and Applications*, *24*(7), 1867-1877.

Luo, Q., Ma, M., & Zhou, Y. (2016). A novel animal migration algorithm for global numerical optimization. *Computer Science and Information Systems*, *13*(1), 259-285.

Mafarja, M. M., & Mirjalili, S. (2017). Hybrid whale optimization algorithm with simulated annealing for feature selection. *Neurocomputing*, *260*, 302-312.

Mirjalili, S., & Hashim, S. Z. M. (2010, December). A new hybrid PSOGSA algorithm for function optimization. In *2010 international conference on computer and information application* (pp. 374-377). IEEE.

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

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

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

Murty, K. G. (2003). Optimization models for decision making: Volume. *University of Michigan, Ann Arbor*.

Pan, X., Xue, L., Lu, Y., & Sun, N. (2019). Hybrid particle swarm optimization with simulated annealing. *Multimedia Tools and Applications*, *78*(21), 29921-29936.

Qaddoura, R., Faris, H., Aljarah, I., & Castillo, P. A. (2021). EvoCluster: An Open-Source Nature-Inspired Optimization Clustering Framework. *SN Computer Science*, *2*(3), 1-12.

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

Sayed, G. I., & Hassanien, A. E. (2018). A hybrid SA-MFO algorithm for function optimization and engineering design problems. *Complex & Intelligent Systems*, *4*(3), 195-212.

Sayed, G. I., Darwish, A., & Hassanien, A. E. (2019). Quantum multiverse optimization algorithm for optimization problems. *Neural Computing and Applications*, *31*(7), 2763-2780.

Song, R., Zeng, X., & Han, R. (2020). An Improved Multi-Verse Optimizer Algorithm For Multi-Source Allocation Problem. *International Journal of Innovative Computing, Information and Control, 16*(6), 1845–1862.

Storn, R. (1996, June). On the usage of differential evolution for function optimization. In *Proceedings of North American Fuzzy Information Processing* (pp. 519-523). IEEE.

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

Talbi, E. G. (2002). A taxonomy of hybrid metaheuristics. *Journal of heuristics*, *8*(5), 541-564.

Talbi, E. G. (2009). *Metaheuristics: from design to implementation* (Vol. 74). John Wiley & Sons.

Ting, T. O., Yang, X. S., Cheng, S., & Huang, K. (2015). Hybrid metaheuristic algorithms: past, present, and future. *Recent advances in swarm intelligence and evolutionary computation*, 71-83.

Wang, C., Lin, M., Zhong, Y., & Zhang, H. (2016). Swarm simulated annealing algorithm with knowledge-based sampling for travelling salesman problem. *International Journal of Intelligent Systems Technologies and Applications*, *15*(1), 74-94.

Wilcoxon, F. (1992). Individual comparisons by ranking methods. In *Breakthroughs in statistics* (pp. 196-202). Springer, New York, NY.

Yang, X. S. (2010). *Nature-inspired metaheuristic algorithms*. Luniver press.

Yang, X. S., & Deb, S. (2009, December). Cuckoo search via Lévy flights. In *2009 World congress on nature & biologically inspired computing (NaBIC)* (pp. 210-214). IEEE.