An Improved Equilibrium Optimizer Algorithm for Tackling Global Optimization Problems

Samia Mandour*[1], Ibrahim el-henawy[1] and Kareem Ahmed[2]
[1]Zagazig University, Zagazig, Egypt, Emails: eng.samia.mandour@gmail.com; ielhenawy@zu.edu.eg
[2]Computer Science department, Beni-Suef University, Egypt, Email: Kareem_ahmed@hotmail.co.uk

Abstract

This paper introduces a new, metaheuristic optimization algorithm, named an Improved Metaheuristic Equilibrium Optimizer (IMEO). The algorithm Equilibrium Optimizer (EO), is inspired by its method of estimating both equilibrium and dynamics, based on mass balance models. Studying the EO closely, we find that EO does not have the potential to get closer to the optimal global solution when it solves certain problems. To improve EO efficiency, this paper suggests using an improvement, called an elite opposition learning-based, that increases the speed and accuracy of EO convergence, and helps the algorithm to get a better solution. Falling into local optima is a big problem, EO suffers from the fact that when we look deeply at the standard EO mathematical formula, we found that the algorithm is trying to get out of the local optima, but sometimes it can't, so we're introducing a new mathematical formula based on using cosine trigonometric function. To validate the proposed algorithm efficiency, The IMEO algorithm is evaluated on 23 benchmarks and compared with other common naturalistic heuristic algorithms. The statistical analysis shows that the results of IMEO achieve better performance compared to the standard EO and several well-known algorithms on several benchmark issues.

Keywords: Meta-heuristic algorithms, Equilibrium Optimizer algorithm, Elite opposition-based learning strategy, Benchmark problems.

1. Introduction

Several prominent naturalistic heuristic algorithms have been sophisticated and applied to global optimization in recent years [1]. Involving those algorithms, the swarm intelligence could effectively tackle the global optimization due to their convergence speed and escaping from local optima [2]. Therefore, several swarm-based optimization algorithms have been recently proposed for solving the global optimization with further improvements, some of which will be surveyed in the rest of this section.

Mirjalili et al. [3] proposed a new algorithm called Grey Wolf Optimizer (GWO) that mimics the behavior of the grey wolves and proves its capability in tackling the competitive problems with unknown search spaces . In [4], the authors have been developed a new variant of GWO for solving the global optimization. This variant was based on integrating a novel weighted distance strategy with GWO to improve its search ability for reaching more intractable regions within the search space as an attempt to escape from local optima, in addition to accelerating convergence rates. Furthermore, Luo, Jun, and Zewei Liu [5] proposed a hybridization algorithm called MDE-GWO based on utilizing the merits of the JADE with opposition-based theory to improve the ability of GWO for avoiding stuck into local minima.

Based on the inspiration of salp swarm, the Salp Swarm Algorithm (SSA) [6] is developed for solving the single and multi-objective optimization problems. SSA can easily trap into local optima as the follower salps blindly follow the leading one. For handling the previous shortcoming of SSA, Nautiyal et al. [7] added the Gaussian, Cauchy, and levy-flight mutation schemes. Also, Salgotra et al. [8] adopted the division of the generations concept in addition to the logarithmic parameter that plays an important role in taking the SSA gradually to the exploitation phase.

Another variant of metaheuristic algorithms is the Whale Optimization Algorithm (WOA) [9] that shows a high superiority compared to other existing algorithms. Chakraborty et al. [10] enhanced the WOA with the Modified mutualism phase borrowed from the symbiotic organisms search for the exploration and

DOI: 10.5281/zenodo.4621285

exploitation purposes. For the same purpose, Fan et al. [11] combined the WOA with the SSA and supported by opposition-based learning strategy.

Mirjalili [12] designed a Sine-Cosine Algorithm (CSA) based on the mathematical sine and cosine functions. CSA suffers from poor convergence for complex and high dimensional problems. For this reason, in [13], a multi-strategy of chaotic local search mechanism, Cauchy mutation, and opposition-based learning strategy to balance the exploration and exploitation principles. Gupta et al. [14] replaced the non-linear transition rule with the linear one for better transition between the exploration and exploitation. Additionally, the mutation operation is used to generate new solutions. Moreover, in [15], a memory matrix of best solutions is stored to guide the candidate solutions during iterations.

Many other algorithms are designed to solve the global optimization problem, such as Particle Swarm Optimization and Gravitational Search Algorithm (PSOGSA) [16], seagull optimization algorithm [17], rat swarm optimizer [18], symbiotic organisms search [19], grey prediction algorithm [20], flower pollination algorithm [21], butterfly optimization algorithm [22], and tunicate swarm algorithm [23]. Although the existence of several algorithms. They may suffer from limitations, including poor convergence and falling into local optima. Recently, a novel meta-heuristic algorithm known as equilibrium optimizer (EO) has been proposed by Mirjalili [24] for tackling the global optimization. The EO algorithm has shown unprecedented success in solving many problems as feature selection [25], parameter estimation [26-28], economic dispatch problem [29], power systems [30], and segmentation [31]. Although of its significant performance, the standard EO still suffers from low accuracy because of local optima and slow convergence rate. This motivates us to propose an Improved Metaheuristic of EO (IMEO) variant with an effective strategy to avoid falling forever into local minima with accelerating convergence speed. The IMEO is integrated with an elite opposition based learning technique. We compared the IMEO with other existing algorithms using 23 benchmarks using many performance measures such as mean, Standard Deviation (SD), best and worst values.

The remaining parts of this paper are categorized as follows: Section (2) sets out the full specification of the EO standard with regard to its nature of inspiration and its mathematical methodology. Section (3) sets out the full specification of elite opposition learning based technology. Section (4) introduces simulation experiments and analysis of results. Section (5) sets out the conclusion and future works.

2. Equilibrium Optimizer algorithm

The inspiration, the mathematical model and the Equilibrium Optimizer (EO) algorithm are introduced in this section.

2.1 Inspiration analysis

The inspiration for the EO strategy is a simplistic excellently-mixed dynamic mass tradeoff of the control volume, in which the mass balance formula has been used to characterize the mental focus of the unreactive constituent in the control volume as a function of its innumerable source and sink mechanisms. The mass balance equation plays an effective role in providing basic physics in order to preserve the mass as it enters, leaves, and then generates it in the volume of control. To express the balance of the general mass, a regular equation is used from the first degree (differential equation), where the change in the mass of time is equal to the amount of mass entering the system and is added to the amount of mass you create inside minus the amount that leaves the system is presented as follows:

$$S\frac{dv}{dt} = Qv_{eq} - Qv + Gr \quad (1)$$

V is the concentricity within the control volume (S), $S\frac{dv}{dt}$ is the mass' change rate in the control volume. Q is the rate that expresses the volume flow in and out of the control volume. v_{eq} represents the concentricity of the balance case, as there is no generation within the control volume. The rate of mass generation within control volume is symbolized asGr. Once $S\frac{dv}{dt}$reaches zero, we can say that a stable equilibrium has been reached. The re-arrangement of equation (1) enables a solution to be made for $\frac{dv}{dt}$as a function of $\frac{Q}{s}$; where

$\frac{Q}{S}$ symbolizes the inverse of the time of residence abbreviated to here as λ (λ=$\frac{Q}{S}$) and means turnover rate. Equation 1 can be rearranged to solve the concentration in control volume (v) as a time function (tm):

$$\frac{dv}{\lambda v_{eq}-\lambda v+\frac{Gr}{S}} = dt \tag{2}$$

When equation 2 is subject to integration, the third equation is produced as follows:

$$\int_{v_0}^{v} \frac{dv}{\lambda v_{eq}-\lambda v+\frac{Gr}{S}} = \int_{tm_0}^{tm} dt \tag{3}$$

This leads to:

$$v = v_{eq} + \left(v_0 - v_{eq}\right)Fn + \frac{Gr}{\lambda S}(1 - Fn) \tag{4}$$

Where,

$$Fn = \exp\left[-\lambda(tm - tm_0)\right] \tag{5}$$

Where, tm_0 symbolized to the premier start time, and in the same manner v_0 represents concentricity, based on the integration period. Either estimates the concentricity of the control volume at a known turnaround rate or calculates the average turnaround rate using a simple mathematical linear regression with a known generation rate and other circumstances, they can be evaluated using equation 4. The EO mathematical framework is designed with the help of the previous equations. Each particle is tantamount to a solution in the search space and each particle position represents concentricity in the optimization of the particle swarm. A particle updating' rules are represented via three terms, shown in equation (4). The concentricity of each particle is updated through these three terms. Balance (equilibrium) concentration is the first term, knew as one of the best-of - the-art solutions selected randomly from the pool, called the Balance Pool. Second terms is related to the difference between the concentricity of the particle and the corresponding equilibrium case acts as a direct search mechanism. This term motivates explorers (particles) to explore the search region effectively. Looking deeply to the generation rate, we find that it is related to the third term, as it plays the exploiter role. Sometimes it gives a share in exploration. Previous three terms and their effect way in the search pattern are manipulated in the following subsections:

2.1.1 Initialization and function evaluations

EO starts the optimization process by generating initial population, analogous to other metaheuristics. Concentrations are created using uniform stochastic initialization with respect to particles' number, and dimensions' number as follows:

$$v_i^{initial} = v_{min} + randi(v_{max} - v_{min}), \qquad i = 1,2,3, \dots\dots\dots\dots, n \tag{6}$$

$v_i^{initial}$represents $i\ th$ particle premier concentration vector. v_{min} , and v_{max} are the values of minimum and maximum for the corresponding dimensions. $randi$ is a stochastic vector generates all its values within the range of [0,1]. Population size is represented by n. According to the fitness function, particles is being evaluated and sorted to decide who will be equilibrium candidates.

2.1.2 Equilibrium pool and candidates (v_{eq})

The final convergence case of this algorithm (balance state) is considered the most globally optimal. At the earliest stage of the optimization process, we do not know the balance (equilibrium) state, and only balance candidates set to supply the particles with the search pattern. Experimental results of different problems cases indicate that four solutions represent the best solutions ever during the optimization process, in addition to another one, which concentrates on the average of the previously mentioned four best

particles. With the help of previously mentioned, four birds are being capable of exploring the search space more effectively, while the arithmetic mean improves the EO' exploitation capability. The candidate's number is specified with respect to the type of optimization problems. Looking deeply into the GWO formula, we find that GWO updates the wolf's position according to the three best wolfs (best wolfs: α, β and gamma). It is available to you to use a different number of candidates according to your goal; means if you want to improve unimodal functions' results, you should decrease candidate's number (i.e., less than four candidates); however, this modification will negatively impact on multimodal functions' results, so according to your aim, you should decide the desired candidate's number. Equilibrium nominated candidates here are five particles, and they are used for equilibrium pool vector construction:

$$\vec{v}_{eq,pool} = (\vec{v}_{eq,1}, \vec{v}_{eq,2}, \vec{v}_{eq,3}, \vec{v}_{eq,4}, \vec{v}_{eq(av)}) \quad (7)$$

Looking at each of the iterations, we find that: the particle concentration is updated by randomly choosing it among the selected candidates with the same probability, and by referring to the process of updating the work of the first particle, We find that the particle updates its concentration based on $\vec{v}_{eq,1}$ previously mentioned in the first iteration, then we see that the same particle updates its focus according to $\vec{v}_{eq(av)}$ in the second iteration, and a comparison will be made between all candidate solutions receiving the same amount of updates with the same likelihood until the end of the update process.

2.1.3 Exponential term (x)

The exponential x term is the axis that contributes to the updating process of concentration, as it is an important factor in creating the balance between exploration and exploitation within the EO algorithm. As the rate in a real control volume can vary with time, λ is supposed to be a random vector in the interval of [0, 1].

$$x^{\rightarrow} = exp^{-\lambda(time-time(0))} \quad (8)$$

As time is a function of frequency, which in turn decreases with the number of iterations

$$time = (1 - \frac{it}{maxit-it})^{(cons_2 \frac{it}{maxit-it})} \quad (9)$$

We find that it and $maxit$ express the current frequency values and the maximum number of iterations in order when examining the previous equation, and by looking, we also find that a constant $cons_2$ is referred to, whose role is summarised in the ability to exploit, And by looking at the following equation, we find that the primitive period $time(0)$ has been referred to in order to maintain the slow speed of search and improve the balance between exploration and exploitation.

$$time(0)^{\rightarrow} = \frac{1}{\lambda^{\rightarrow}} \ln\left(-cons_1 sign(m^{\rightarrow} - 0.5)\left[1 - exp^{-\lambda * time}\right]\right) + time \quad (10)$$

In controlling the ability to exploit, $cons_1$ plays its role as it allows its greater value to enhance the search area's exploration capacity and to review the value of exploitation. The role of $cons_2$ comes in the same pattern, which contradicts the method of operating the first constant in its work, as it increases the ability to exploit and slows down the capacity of exploration, and we find that it creates a balance between exploitation and exploration in this way. With reference to the value of $sign(m^{\rightarrow} - 0.5)$, we find that if the value of m is located as a random vector in periods 0 and 1 and in solving the problems identified in this research paper, the value of $cons_1 = 2$ and the value of $cons_2 = 1$. Equation (11) demonstrates the revised version of Eq. (8) with the replacement of Eq. (10) with Eq. (8).

$$x^{\rightarrow} = \left(cons_1 sign(m^{\rightarrow} - 0.5)\left[exp^{-\lambda * time} - 1\right]\right) \quad (11)$$

2.1.4 Generation rate (Gr)

One of the most prominent terms introduced in Algorithm EO is generation rate, as it provides an accurate solution through exploitation optimization. In many engineering applications, there are many models for expressing the rate of generation as a function of time. For example, the following is defined as one multi-purpose model:

$$Gr^{\rightarrow} = Gr_0{}^{\rightarrow} exp^{-k^{\rightarrow}(time-time(0))} = Gr_0{}^{\rightarrow} x^{\rightarrow} \quad (12)$$

Where $Gr_0{}^{\rightarrow}$is the initial value and a deterioration constant is indicated by k. To have a more systematic and controlled search pattern. In order to limit the number of stochastic variables, this research assumes that K = λ and utilizes the exponential term previously derived. Accordingly, the following is the final set of generation rate equations:

$$Gr^{\rightarrow} = Gr_0{}^{\rightarrow} exp^{-k^{\rightarrow}(time-time(0))} = Gr_0{}^{\rightarrow} x^{\rightarrow} \quad (13)$$

$$Gr_0{}^{\rightarrow} = GRC^{\rightarrow}(v_{eq}{}^{\rightarrow} - \lambda^{\rightarrow} v^{\rightarrow}) \quad (14)$$

$$GRC^{\rightarrow} = \begin{cases} 0.5\, rand1 & rand1 \geq G_{prob} \\ 0 & rand2 < G_{prob} \end{cases} \quad (15)$$

As shown, between range of 0 and 1, rand1 and rand2 express random values. Furthermore, the vector GRC is constructed based on the iteration of the same value obtained from equation 15. G_{prob} includes the possibility of the term contributing to the process of modernization, which can be said to be the control parameter of the generation rate , and it can be said that the number of particles that use the term generation to update their state is through what is called the generation of probability. The mechanism of these contributions is determined by Equations No. 14 and 15. Each particle, on the other hand, is subject to Equation No. 15, and the probability of x is good when it is equal to 0.5, as shown in the research paper on the EO algorithm, as it expresses a good balance between exploration and exploitation:

$$v^{\rightarrow} = v_{eq}^{\rightarrow} + \left(v^{\rightarrow} - v_{eq}^{\rightarrow}\right).x^{\rightarrow} + \frac{Gr^{\rightarrow}}{\lambda^{\rightarrow} v}(1 - x^{\rightarrow}) \quad (16)$$

V is regarded as a unit and the previous equation consists of three parts: the first: the balance concentration indicates, and the other two parts indicate the focus difference. The importance of the second part stems from his duty to search globally in the research area to find an ideal point and the significance of this term is evident in the exploration process, which helps to take advantage of the major differences in focus and when the opportunity arises and finds an ideal point, the third part is more accurate in exploiting that point by taking advantage of small differences in focus Equation (13).

Three criteria make the second and third parts have the same or opposite signs: the concentration of particles and the two equilibrium filters and the rate of rotation (turnover rate:λ) as the mark itself makes the contrast large and this helps to better explore the search area, and on the contrary, the opposite mark's role is to make the contrast small, which helps to improve the local search process in a better way. While the second part, in turn, seeks to find solutions that are relatively far from the equilibrium candidates, and the third part seeks to make these solutions more polished, this does not always work, because small turnover rates (<=0.05) are taken into account in the third term, which increases the contrast and helps exploration ability in certain dimensions.

2.1.5 Memory saving procedures of particle's

It is analogous to the concept of bBest in PSO algorithm, y, and from that point, a memory-preserving feature has been added, which helps each part to track its coordinates in space, meaning that a comparison is made between the appropriate value of each part in the current and previous iteration to determine the

preference between them to work on and this helps to increase the capacity of exploitation [18]. The pseudo code of the EO algorithm along with a memory saving feature is presented in fig (1).

Initialize the population i = 1…n
Assign equilibrium candidates' fitness a large number
Set the parameter values: $cons_1 = 2$, $cons_2 = 1$, $G_{prob} = 0.5$
While it<max-it
For i=1: n of candidates
Calculate each candidate' fitness
If fitness $\vec{v_i}$ < fitness $\vec{v_{veq\ 1}}$
Replace $\vec{v_{eq\ 1}}$ with $\vec{v_i}$ and fitness of $\vec{v_{eq\ 1}}$ with fitness of $\vec{v_i}$
Else if fitness $\vec{v_i}$ > fitness $\vec{v_{eq\ 1}}$ && fitness $\vec{v_i}$ < fitness $\vec{v_{eq\ 2}}$
Replace $\vec{v_{eq\ 2}}$ with $\vec{v_i}$ and fitness of $\vec{v_{eq\ 2}}$ with fitness of $\vec{v_i}$
Else if fitness $\vec{v_i}$ > fitness $\vec{v_{eq\ 1}}$ && fitness $\vec{v_i}$ >fitness $\vec{v_{eq\ 2}}$ && fitness $\vec{v_i}$ <fitness $\vec{v_{eq\ 3}}$
Replace $\vec{v_{veq\ 3}}$ with $\vec{v_i}$ and fitness of $\vec{v_{eq\ 3}}$ with fitness of $\vec{v_i}$
Else if fitness $\vec{v_i}$ > fitness $\vec{v_{eq\ 1}}$ && fitness $\vec{v_i}$ >fitness $\vec{v_{eq\ 2}}$ && fitness $\vec{v_i}$ >fitness $\vec{v_{eq\ 3}}$ && fitness $\vec{v_i}$ <fitness $\vec{v_{eq\ 4}}$
Replace $\vec{v_{eq\ 4}}$ with $\vec{v_i}$ and fitness of $\vec{v_{eq\ 4}}$ with fitness of $\vec{v_i}$
End if
End for
$\vec{v_{av}} = (\vec{v_{veq\ 1}} + \vec{v_{veq\ 2}} + \vec{v_{veq\ 3}} + \vec{v_{veq\ 4}})/4$
Building the equilibrium pool $\underset{v\ eq,pool}{\rightarrow}$ = ($\underset{v\ eq,1}{\rightarrow}$, $\underset{v\ eq,2}{\rightarrow}$, $\underset{v\ eq,3}{\rightarrow}$, $\underset{v\ eq,4}{\rightarrow}$, $\underset{v\ eq(av)}{\rightarrow}$)
Complete memory saving if (it >1)
Set $time = (1 - \frac{it}{maxit-it})^{(cons_2 \frac{it}{maxit-it})}$ Equation (9)
For i= 1: n of candidates
Pick random one candidate from the previously mentioned equilibrium pool
Construct the random vectors: $\lambda^{\rightarrow}, m^{\rightarrow}$ Equation (11)
Create $x^{\rightarrow} = \left(cons_1 sign(m^{\rightarrow} - 0.5)\left[exp^{-\lambda * time} - 1\right]\right)$ Equation (11)
Create $GRC^{\rightarrow} = \begin{cases} 0.5\ rand1 & rand1 \geq G_{prob} \\ 0 & rand2 < G_{prob} \end{cases}$ Equation (15)
Create $Gr_0{}^{\rightarrow} = GRC^{\rightarrow}(v_{eq}{}^{\rightarrow} - \lambda^{\rightarrow} v^{\rightarrow})$ Equation (14)
Create $Gr^{\rightarrow} = Gr_0{}^{\rightarrow} x^{\rightarrow}$ Equation (13)
Update concentrations according to Equation 16 $v^{\rightarrow} = v_{eq}^{\rightarrow} + \left(v^{\rightarrow} - v_{eq}^{\rightarrow}\right).x^{\rightarrow} + \frac{Gr^{\rightarrow}}{\lambda^{\rightarrow} v}(1 - x^{\rightarrow})$
End For
it=it+1
End while

Figure 1 Pseudo code of equilibrium optimizer

3. An improved Metaheuristic equilibrium optimizer algorithm (IMEO)

EO, which tested low dimensional (unimodal) and high dimensional (multimodal) optimization problems, was introduced by Afshin Faramarzi [24] . We had to add the elite opposition technique to the equilibrium optimizer algorithm to increase the exploration power of the methodology explained in section to increase the effectiveness, accuracy of EO convergence, and to cover search space more efficiently (3.1). Section (3.2) illustrates the new methodology to improve the exploitation phase of EO.

3.1 Elite opposition learning based strategy (EOBL)

It appeared to be an elite opposition capability to perform better than an opposition-based one to study and test both elite opposition interrogation techniques and opposition-based learning. This research paper shows that the source food is selected and considered to be the object of EOBL to extract an elite food source one, based on many scientific studies put forward in previous years, so that EOB food source solutions can be presented with the following implications [32].

ceq_j^{it} , and $oppositi_j^{it}$ declare the j_{th} vector of source food and its elite one in the EO at the current it iteration, respectively. Where, $ceq_j^{it} \in [g_j^{it}, e_j^{it}]$ and $oppositi_j^{it}$ signifies the elite opposite point of $FoodS_j^{itt}$ and can be described as follows:

$$oppositi_j^{it} = rand\left(g_j^{it} + e_j^{it}\right) - ceq_1^{it} \tag{17}$$

$rand$ is a fake number that is spread over a uniform distribution in the limited [0, 1] range, and g_j^{it} and e_j^{it} are concluded at the upper bound and lower bound and can be expressed by the mathematical formula that accompanies it:

$$\begin{cases} g_j^{it} = \min(v) \\ e_j^{it} = \max(v) \end{cases} \tag{18}$$

Both min(v) and max (v) point to the minimization and maximization function, respectively. In some cases, op can break the borders and, in this case, by recruiting the EOB concept for this infeasible solution, the reset methodology illustrated as follows, the solution becomes possible.

$$oppositi_j^{it} = \begin{cases} 2 * g_j^{it} - oppositi_j^{it}, & if\ oppositi_j^{it} > \text{ub} \\ 2 * e_j^{it} - oppositi_j^{it}, & if\ oppositi_j^{it} < \text{lb} \end{cases} \tag{19}$$

In the sharing of information between the population and the elite, elite opposition stances play a prominent part. It plays a key role to some extent in expanding the search area for algorithms, increasing population diversity, and addressing the problem of local optimization.

3.2 An Improved Metaheuristic EO based on the new phase of exploitation

The candidate solutions sometimes fall under the shadows of the local optima; hence, we introduce the concept of using Equation 20 with a probability to help get out of that problem with these solutions:

$$\vec{v_{veq}} = \vec{v_{veq}} * \cos(90 * (\text{maxit} - 1/(\text{maxit} + 1)) * (0.5 * \text{rand} * 0.5) \tag{20}$$

The following pseudocode 2 indicates how the proposed IMEO work.

Initialize the population i = 1…n
Assign equilibrium candidates' fitness a large number
Set the parameter values: $cons_1 = 2$, $cons_2 = 1$, $G_{prob} = 0.5$
While it<max-it
    For i=1: n of candidates
    Calculate each candidate' fitness
      If fitness $\vec{v_i}$ < fitness $\vec{v_{veq\,1}}$
      Replace $\vec{v_{eq\,1}}$ with $\vec{v_i}$ and fitness of $\vec{v_{eq\,1}}$ with fitness of $\vec{v_i}$
      Else if fitness $\vec{v_i}$ > fitness $\vec{v_{eq\,1}}$ && fitness $\vec{v_i}$ < fitness $\vec{v_{eq\,2}}$
      Replace $\vec{v_{eq\,2}}$ with $\vec{v_i}$ and fitness of $\vec{v_{eq\,2}}$ with fitness of $\vec{v_i}$

Else if fitness $v_i^{\rightarrow}$ > fitness $v_{eq\,1}^{\rightarrow}$ && fitness $v_i^{\rightarrow}$ >fitness $v_{eq\,2}^{\rightarrow}$ && fitness $v_i^{\rightarrow}$ <fitness $v_{eq\,3}^{\rightarrow}$

Replace $v_{veq\,3}^{\rightarrow}$ with $v_i^{\rightarrow}$ and fitness of $v_{eq\,3}^{\rightarrow}$ with fitness of $v_i^{\rightarrow}$

Else if fitness $v_i^{\rightarrow}$ > fitness $v_{eq\,1}^{\rightarrow}$ && fitness $v_i^{\rightarrow}$ >fitness $v_{eq\,2}^{\rightarrow}$ && fitness $v_i^{\rightarrow}$ >fitness $v_{eq\,3}^{\rightarrow}$ && fitness $v_i^{\rightarrow}$ <fitness $v_{eq\,4}^{\rightarrow}$

Replace $v_{eq\,4}^{\rightarrow}$ with $v_i^{\rightarrow}$ and fitness of $v_{eq\,4}^{\rightarrow}$ with fitness of $v_i^{\rightarrow}$

End if

End for

$v_{av}^{\rightarrow} = (v_{veq\,1}^{\rightarrow} + v_{veq\,2}^{\rightarrow} + v_{veq\,3}^{\rightarrow} + v_{veq\,4}^{\rightarrow})/4$

Building the equilibrium pool $\underset{v\ eq,pool}{\rightarrow} = (\underset{v\ eq,1}{\rightarrow}, \underset{v\ eq,2}{\rightarrow}, \underset{v\ eq,3}{\rightarrow}, \underset{v\ eq,4}{\rightarrow}, \underset{v\ eq(av)}{\rightarrow})$

Complete memory saving if (it >1)

Set $time = (1 - \frac{it}{maxit - it})^{(cons_2 \frac{it}{maxit - it})}$ Equation (9)

For i= 1: n of candidates

Pick random one candidate from the previously mentioned equilibrium pool

If rand< prob

$v_{veq}^{\rightarrow} = v_{veq}^{\rightarrow} * \cos(90 * (\text{maxit} - 1/(\text{maxit} + 1)) * (0.5 * \text{rand} * 0.5)$ Equation(20)

end

Construct the random vectors: $\lambda^{\rightarrow}, m^{\rightarrow}$ Equation (11)

Create $x^{\rightarrow} = \left(cons_1 sign(m^{\rightarrow} - 0.5)\left[exp^{-\lambda * time} - 1\right]\right)$ Equation (11)

Create $GRC^{\rightarrow} = \begin{cases} 0.5\, rand1 & rand1 \geq G_{prob} \\ 0 & rand2 < G_{prob} \end{cases}$ Equation (15)

Create $Gr_0{}^{\rightarrow} = GRC^{\rightarrow}(v_{eq}{}^{\rightarrow} - \lambda^{\rightarrow} v^{\rightarrow})$ Equation (14)

Create $Gr^{\rightarrow} = Gr_0{}^{\rightarrow} x^{\rightarrow}$ Equation (13)

Update concentrations according to Eq: 16 $v^{\rightarrow} = v_{eq}^{\rightarrow} + \left(v^{\rightarrow} - v_{eq}^{\rightarrow}\right).x^{\rightarrow} + \frac{Gr^{\rightarrow}}{\lambda^{\rightarrow} v}(1 - x^{\rightarrow})$

a = min ($v^{\rightarrow}$) Equation (17)

b = max ($v^{\rightarrow}$)

for *j*= 1: dim

$opposit_j^{it} = rand\left(g_j^{it} + e_j^{it}\right) - veq_1^{it}$ Equation (16)

If $opposit_j^{it} > g_j^{it}$

$opposit_j^{it} = 2 * g_j^{it} - opposit_j^{it}$, Equation (18)

End

If $opposit_j^{it} < e_j^{it}$

$opposit_j^{it} = 2 * e_j^{it} - opposit_j^{it}$,

end

end

calculate the fitness of $opposit$

if $(opposit - fitness) < v_{eq1}^{\rightarrow}$- $fitness$

replace the position of $v_{eq1}^{\rightarrow}$ with the $opposit$ one

replace $v_{eq1}^{\rightarrow}$- $fitness$ with $(opposit - fitness$

end

End For

it=it+1

End while

Figure 2 The Improved Metaheuristic of Equilibrium Optimizer pseudocode

4. Simulation and results

This section examines the comparative outcomes of the proposed improved algorithm and most of the well-known meta-heuristic (MH) algorithms. Using MATLAB R2015 (a) on a personal computer running

Windows 7/64-bit / Intel Core (TM) i7-3840QM, 2.80 GHz, and 16 GB RAM, all the algorithms used were coded and run in the same way. A proper assessment of the enhanced algorithm is measured by running a competition on a set of benchmark issues to assess the performance between it and the mentioned algorithms.

4.1. Functions of benchmark assessment

Based on the twenty-three classical benchmark tasks in following tables 1, 2, and 3, respectively, the proposed IMEO approach is affirmed. Indeed, benchmark suits may be classified into three categories. The first category is shown in (Table -1) and this table depicts the typical one dimension of evaluation problems. The second category is appeared in (Table -2), The multimodal dimension of the assessment issues is represented in this table. Static multidimensional screening functions down to the category 3 shown by (Table-3). Challenges levels of different dimensions and benchmark problem landscapes are different, making them appropriate for assessing the performance of exploration and exploitation algorithms.

Table 1 the high unimodal dimension of evaluation problems

Bench-mark function	Dimension	Range	$\mathbf{fun_{min}}$
$fun_{(1)} = \sum_{i=1}^{n} r_i^2$	30	$r_i \in[-100,100]$	0
$fun_{(2)} = \sum_{i=1}^{n} \lvert r_i\rvert + \prod_{i=1}^{n} \lvert r_i\rvert$	30	$r_i \in[-10,10]$	0
$fun_{(3)} = \sum_{i=1}^{n} (\sum_{j=1}^{i} r)^2$	30	$r_i \in[-100,100]$	0
$fun_{(4)} = max_i\lvert r_i\rvert, 1 < i < D$	30	$r_i \in[-100,100]$	0
$fun_{(5)} = \sum_{i=1}^{D-1} [100(r_{i+1} - r_i^{\,2})^2 + (r_i - 1)^2]$	30	$r_i \in[-30,30]$	0
$fun_{(6)} = \sum_{i=1}^{n} (\lvert r_i + 0.5\rvert)^2$	30	$r_i \in[-100,100]$	0
$fun_{(7)} = \sum_{i=1}^{n} r_i^4 + random(0,1)$	30	$r_i \in[-1.28,1.28]$	0

Table 2 the high dimension multi-modal evaluation problems

Bench-mark function	Dimension	Range	$\mathbf{fun_{min}}$
$fun_8 = -\sum_{i=1}^{n} -r_i \sin(\sqrt{\lvert x_i\rvert})$	30	$r_i \in[-500,500]$	-418.9829n
$fun_9 = \sum_{i=1}^{n} [r_i^2 - 10\cos(2\pi r_i) + 10]$	30	$r_i \in[-5.12,5.12]$	0

$fun_{10} = -20\exp\left(-0.2\sqrt{\frac{1}{n}\sum_{i=1}^{n} r_i{}^2}\right) - \exp\left(\frac{1}{n}\sum_{i=1}^{n} cos2\pi r_i\right) + 20 + e$	30	$r_i \in[-32,32]$	0
$fun_{11} = \frac{1}{4000}\sum_{i=1}^{n}(r_i^2) - \prod_{i=1}^{n}\cos\left(\frac{r_i}{\sqrt{i}}\right) + 1$	30	$r_i \in[-600,600]$	0
$fun_{12} = 0.1\left\{sin^2(3\pi r_1) + \sum_{i=1}^{n}(r_i - 1)^2[1 + sin^2(3\pi r_{i+1})] + (r_n - 1)^2[1 + sin^2(2\pi r_n)]\right\} + \sum_{i=1}^{n}\mu(r_i, 10,100,4)$	30	$r_i \in[-50,50]$	0
$fun_{13} = 0.1\left\{sin^2(3\pi r_1) + \sum_{i=1}^{n}(r_i - 1)^2[1 + sin^2(3\pi r_{i+1})] + (r_n - 1)^2[1 + sin^2(2\pi r_n)]\right\} + \sum_{i=1}^{n}\mu(x_i, 5,100,4)$	30	$r_i \in[-50,50]$	0

Table 3 the static dimension multimodal evaluation problems

Benchmark fun	**Dimension**	**range**	**fun_{min}**
$fun_{14} = \left(\frac{1}{500}\sum_{j=1}^{25}\frac{1}{j + \sum_{i=1}^{2}(r_i - a_{ij})^6}\right)^{-1}$	2	$r_i \in[-65,65]$	1
$fun_{15} = \sum_{i=1}^{11}\left[a_i - \frac{r_1(b_i^2 + b_i r_2)}{b_i^2 + r_i x_3 + r_4}\right]^2$	4	$r_i \in[-5,5]$	0.00 0307 5
$fun_{16} = -\frac{1 + \cos\left(12\sqrt{r_1{}^2 + r_2{}^2}\right)}{0.5(r_1{}^2 + r_2{}^2) + 2}$	2	$r_i \in[-5.12,5.12]$	-1
$fun_{17} = 4r_1{}^2 - 2.1r_1{}^4 + \frac{1}{3}r_1{}^6 + r_1 r_2 - 4r_2{}^2 + 4r_2{}^4$	2	$r_i \in[-5,5]$	- 1.03 1628 5

$fun_{18} = [1 + (1 + r_1 + r_2)^2(19 - 14r_1 + 3{r_1}^2 - 14r_2 + 6r_1r_2 + 3{r_2}^2)] * [30 + (2r_1 - 3r_2)^2(18 - 32r_1 + 12{r_1}^2 + 48r_2 - 36r_1r_2 + 27r_2)]$	2	$r_i \epsilon[-5,5]$	3
$fun_{19} = -\sum_{i=1}^{4} c_i \exp\left(-\sum_{j=1}^{3} a_{ij}(r_j - p_{ij})^2\right)$	3	$r_i \epsilon[1,3]$	-3.86
$fun_{20} = -\sum_{i=1}^{4} c_i \exp\left(-\sum_{j=1}^{6} a_{ij}(r_j - p_{ij})^2\right)$	6	$r_i \epsilon[0,1]$	-3.32
$fun_{21} = -\sum_{i=1}^{5}[(r - a_i)(r - a_i)^T + c_i]^{-1}$	4	$r_i \epsilon[0,10]$	-10.1532
$fun_{22} = -\sum_{i=1}^{7}[(r - a_i)(r - a_i)^T + c_i]^{-1}$	4	$r_i \epsilon[0,10]$	-10.4029
$fun_{23} = -\sum_{i=1}^{10}[(r - a_i)(r - a_i)^T + c_i]^{-1}$	4	$r_i \epsilon[0,10]$	-10.5364

4.2. Results and discussion

In order to better confirm the effectiveness of EOBEOA, six well known meta-heuristic algorithms are encouraged: salp swarm algorithm (SSA), whale optimization algorithm (WOA), grey wolf optimization algorithm (GWO), sine cosine algorithm (SCA), population-based hybrid algorithm combining particle swarm optimization and gravity search algorithm (PSOGSA). To make the comparison equitable, the maximum number of iterations for conscience experiments is supposed to be 1000 for all mentioned algorithms run thirty times on each benchmark set, including the standard deviation (SD) throughout the last iteration of the best-estimate iteration.

Table 4 Seven algorithms parameter settings

Algorithm	**Algorithms Parameters values**
PSO-GSA **[16]**	*G*0 =1, *a*=20, *C*1 =0.5, *C*2 =1.5, *w*∈ *[0, 1]*
EO **[33]**	$a_1 = 2$, $a_2 = 1$, $Gp_{prob} = 0.5$
SCA **[12]**	*a*=2
GWO **[3]**	$\alpha^{\rightarrow}$ linearly decreasing from 2 to 0 [7]
WOA **[9]**	α Linearly decreased from 2 to 0 have been used as recommended in [9].
SSA **[6]**	$n_1 = 2e^{-\left(\frac{4itt}{maxitt}\right)^2}$, where itt is the current iteration and maxitt is the maximum number of iterations, n_2 and n_3 are random numbers uniformly distributed in range [0,1]
IMEO	$a_1 = 2$, $a_2 = 1$, $Gp_{prob} = 0.5$

Table 4 summarizes the results of the improved algorithm's evaluation of the problems listed below, as well as its relationship to the algorithms mentioned previously. The table shows that this algorithm outperforms its peers in a number of problems. When we look at the functions 1 and 3 (F1, F3), we can see that the improved algorithm has already found the optimum solution before reaching to maximum number of iterations. When we look at Function No. 2, we can see that the optimized algorithm outperforms the other algorithms significantly, as it has swept all the algorithms to reach the best solution and all its evaluation values are the best among them. When it comes to Function No. 4, we find that the improved algorithm has also been successful in reaching the best solution on its counterparts as the algorithm outperforms its competitors, which are unable to achieve the best results when compared to the proposed algorithm. Looking at the fifth function, we see that the proposed algorithm (IMEO) outperforms all the others based on the Mean values. The sixth function demonstrates that the EO algorithm is effective, and the second place goes to salp swarm algorithm (SSA), and the third place goes to the proposed algorithm (IMEO). When we look at the seventh function, we see that the proposed algorithm was able to find the best solution, while all other algorithms were unable to do so, according to the F6 evaluation.

Table 5 Unimodal benchmark functions results

Fun		GWO	WOA	PSO -GSA	SSA	SCA	EO	IMEO
F1	Best	1.8238e-61	1.8644e-166	1.5772e-19	7.5194e-09	1.5083e-05	3.7503e-90	**0**
	Worst	2.4203e-58	7.3955e-148	1.0000e+04	1.6635e-08	0.1136	3.9146e-84	**0**
	AVG	1.7975e-59	3.7101e-149	2.0000e+03	1.2207e-08	0.0212	2.3919e-85	**0**
	STD	5.3556e-59	1.6534e-148	4.1039e+03	2.4280e-09	0.0338	8.6915e-85	**0**
F2	Best	1.5363e-34	3.6967e-113	1.8251e-09	4.5650e+02	4.3681e-08	5.2772e-49	**6.5896e-242**
	Worst	7.9190e-33	5.5294e-102	127.5683	1.0648e+18	0.0014	5.2014e-47	**5.4099e-236**
	AVG	1.3868e-33	5.0495e-103	12.2510	1.4075e+17	1.2999e-04	8.3220e-48	**3.8455e-237**
	STD	1.7415e-33	1.3364e-102	30.7085	3.3870e+17	3.1692e-04	1.1846e-47	**0**
F3	Best	4.1011e-20	1.6477e+03	712.4705	59.3647	106.0618	5.7850e-28	**0**
	Worst	6.5511e-12	3.9407e+04	4.0081e+04	909.9345	1.0388e+04	5.9848e-22	**0**
	AVG	3.3493e-13	2.2387e+04	1.0205e+04	361.3558	4.2940e+03	8.3382e-23	**0**
	STD	1.4632e-12	1.0928e+04	8.7801e+03	220.3436	3.2530e+03	1.7973e-22	**0**
F4	Best	1.0427e-15	2.9959	25.3828	2.3897	2.8108	2.1314e-24	**1.0847e-223**
	Worst	1.6650e-13	85.8951	89.9957	14.4690	50.8192	7.9271e-21	**9.9226e-216**
	AVG	2.0119e-14	47.7184	62.0984	8.4667	21.2251	1.8151e-21	**5.6702e-217**
	STD	4.2154e-14	32.5033	25.4474	3.6454	13.1247	2.7578e-21	**0**
F5	Best	25.6359	26.3410	15.7772	22.2042	27.9390	24.4147	**24.0796**
	Worst	28.5193	28.7400	8.0032e+07	1.6582e+03	1.1155e+03	25.2220	**24.7565**
	AVG	26.7128	27.3463	1.2000e+07	242.9221	151.8250	24.8063	**24.5063**
	STD	0.7063	0.6574	2.9296e+07	446.1293	253.1963	0.1775	**0.1743**
F6	Best	1.9717e-05	0.5873	1.4035e-19	8.5409e-09	3.9938	**3.2366e-10**	5.0956e-09
	Worst	1.2549	0.3944	1.0100e+04	1.7352e-08	5.5375	**5.2880e-08**	7.7155e-07
	AVG	0.5336	0.0776	2.0000e+03	1.2460e-08	4.5791	**6.4819e-09**	1.4283e-07
	STD	0.3880	0.1309	4.1042e+03	2.5780e-09	0.8959	**1.2385e-08**	2.4026e-07
F7	Best	2.3153e-04	2.0250e-05	0.0382	0.0415	0.0035	3.7536e-04	**4.6102e-06**
	Worst	0.0013	0.0049	0.1487	0.1541	0.1306	0.0016	**3.0062e-04**
	AVG	7.3252e-04	0.0012	0.0798	0.1004	0.0313	7.7244 e-04	**1.1991e-04**
	STD	3.4343e-04	0.0013	0.0295	0.0370	0.0291	3.3244e-04	**8.0394e-05**

When running these algorithms, twenty times, the average CG (convergence curve) representation of EO, IMEO, SSA, PSO-GSA, WOA, GWO, and SCA with unimodal functions is shown in Figs. 1-7. Fig 1 show that IMEO outperforms its peers in terms of convergence rate, as it achieves the best global solution, before reaching to the maximum number of iterations, followed by WOA, and EO, respectively, they converge towards the best, but can't converge towards the global best. Fig 2 also showed the greatest advantage in terms of convergence speed over the others as IMEO converges to the best solution while the other are fell into the abyss of the local optima, WOA placed second, and EO reached thirdly to the best solution. IMEO achieves the optimum solution, and it is the only one, who can reach to optimum. IMEO converges towards the optimum solution from the beginning; thus it is the only one who can reach to optimum, while EO, and GWO try to converge to the optimum, but they can't reach the optimum; so the converge to the best not the optimum. IMEO is the only algorithm can reach to the best, others fall into local optimality. IMEO converge faster than the others from the iteration beginning till end, and EO algorithm reach to optimum secondly as shown in Figure 5. Figure 6 shows that EO converges faster than the others, SSA comes in the second place and IMEO occupies the third place. Fig 7 shows that IMEO is the fastest convergence towards the best solution between others, while GWO and EO reach to the best in order.

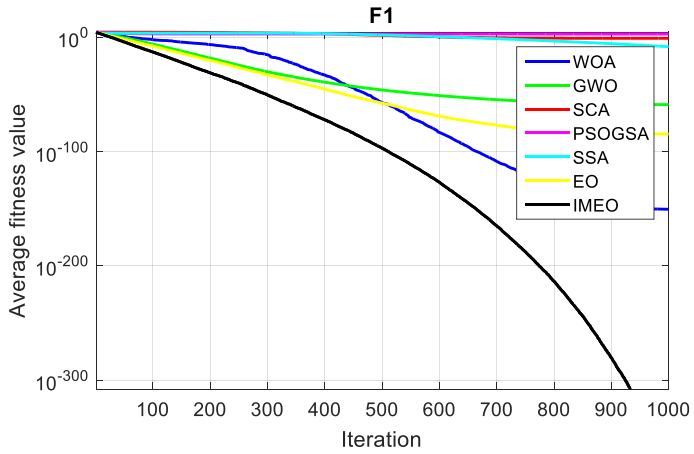

Fig.1: Average fitness value for F1

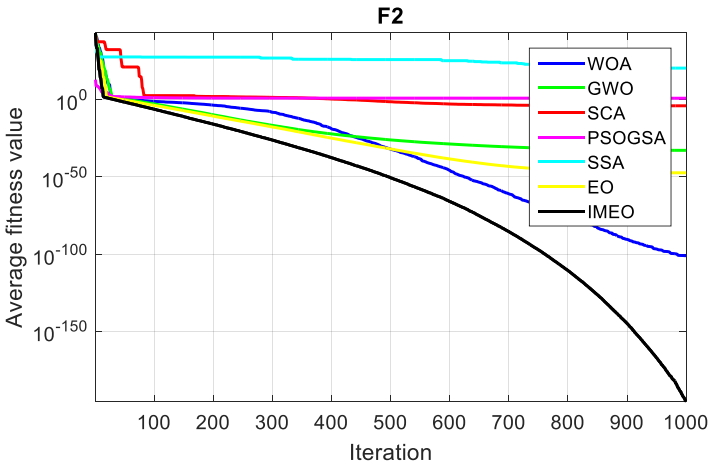

Fig.2: Average fitness value for F2

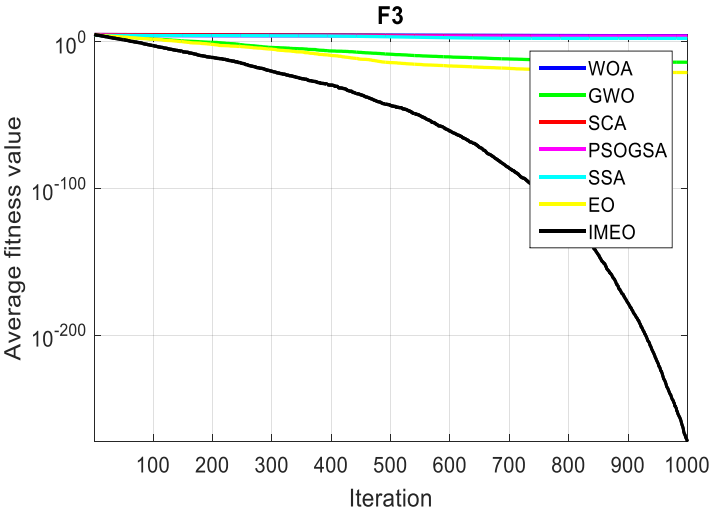

Fig.3: Average fitness value for F3

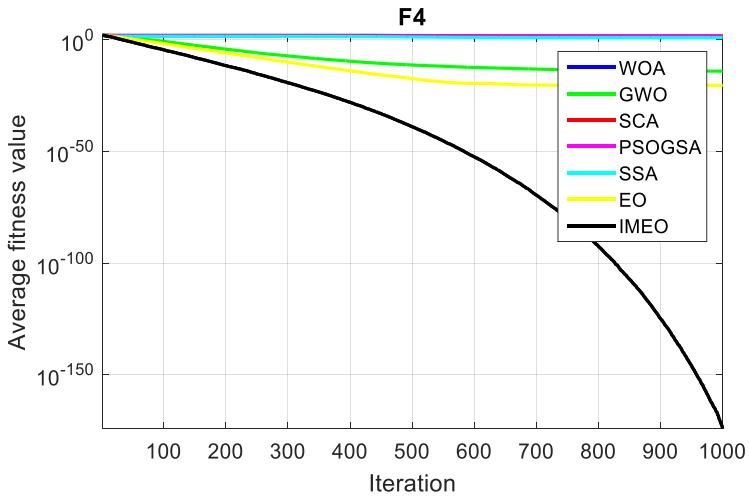

Fig.4: Average fitness value for F4

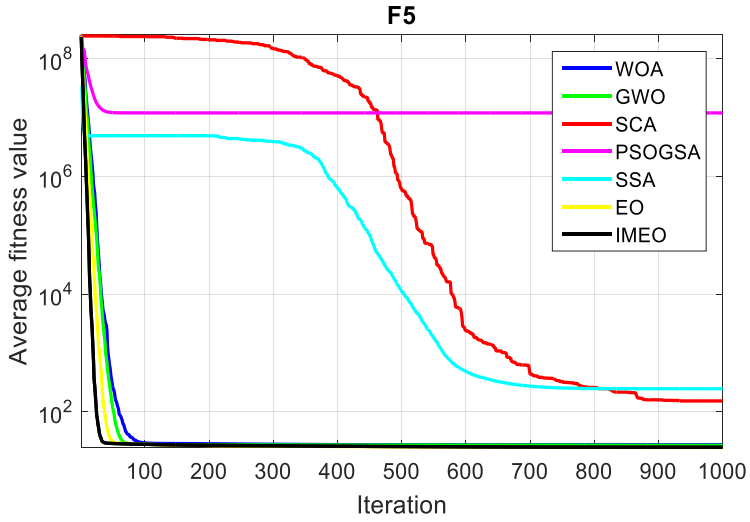

Fig.5: Average fitness value for F5

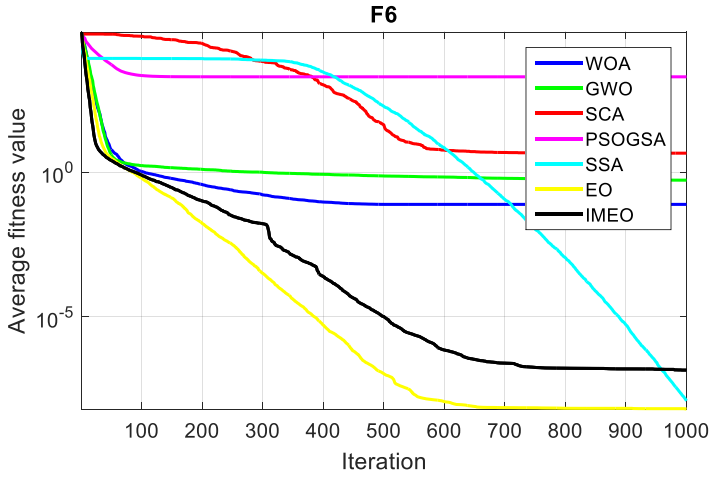

Fig.6: Average fitness value for F6

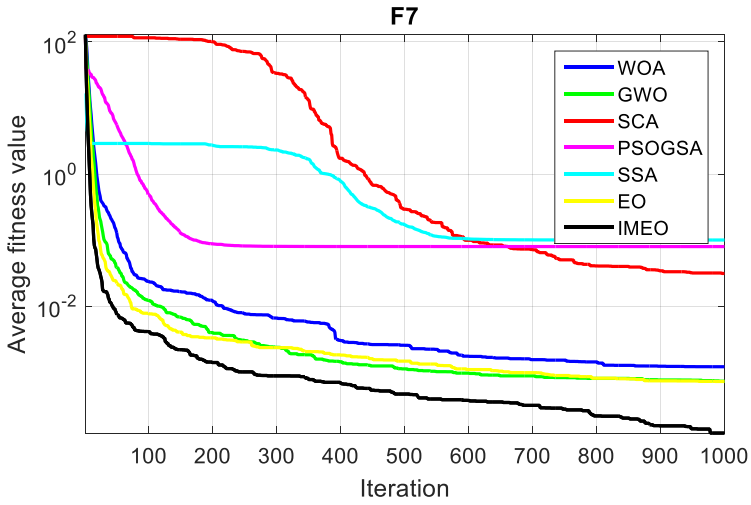

Fig.7: Average fitness value for F7

Looking closely at the results shown in Table 6, from the eighth function to the thirteenth, we find that the improved algorithm has significantly outperformed all its peers on most of functions. Speaking of F8 (best, worst, mean, and standard deviation) evaluation fitness values are shown below in the table, we find that the improved algorithm alone remains on the throne of reaching the best solution in this problem if all the comparative algorithms remain under the shades of the local optima and none of them can go forward in reaching the solution., So, the improved algorithm is the only one that converges on reaching the optimal solution, undisputed. By going through function number nine, we find that the improved algorithm is the fastest algorithm in reaching the optimal solution at the beginning of the iteration and the algorithm EO algorithm comes second in the speed of reaching the solution, followed by the WOA, looking deeply at figs.9, all algorithms of IMEO, EO and WOA reach to the global optimum, but the superiority of IMEO comes from its speed to reach to optimum from the first iteration; thus, it is considered to be the best solution faster than its peers, it is considered to be the best. When it comes to the tenth algorithm, we can see that the improved algorithm prevailed over the others in terms of finding the best solution on its own. Not all algorithms are comparable to IMEO in F11, as a lot of algorithms drop into the local optimality; thus, they don't converge to the best solution, but IMEO, WOA and EO do. Despite the IMEO algorithm's superiority in approaching the answer from the start, the EO algorithm was the only one capable of reaching the best

solution at the end of the F12 and F13 rounds. In the end, we must say that the enhanced algorithm is best for solving most of all these problems, as shown in table 6.

Table 6 Multimodal benchmark evaluations

Fun		GWO	WOA	PSO -GSA	SSA	SCA	EO	IMEO
F8	Best	-7.6072e+03	-1.2569e+04	-8.945e+03	-8.2841e+03	-4.5240e+03	-1.0211e+04	**-7.1349e+241**
	Worst	-4.6065e+03	-8.7377e+03	-5.5040e+03	-6.0416e+03	-3.4682e+03	-7.1334e+03	**-2.6003e+207**
	AVG	-5.8652e+03	-1.2091e+04	-7.2012e+03	-7.5161e+03	-3.9062e+03	-8.9686e+03	**-3.5812e+240**
	STD	7.6126e+02	1.1275e+03	8.8465e+02	5.4920e+02	2.9219e+02	7.8889e+02	**Inf**
F9	Best	0	0	107.4551	23.8790	8.8590e-05	0	**0**
	Worst	5.6843e-14	0	195.0107	107.4552	121.7558	0	**0**
	AVG	8.5265e-15	0	153.0878	62.8315	26.6900	0	**0**
	STD	2.0824e-14	0	25.3416	22.6247	32.4557	0	**0**
F10	Best	1.1546e-15	8.8818e-16	3.6286	1.8551e-05	3.2430e-04	4.4409e-15	**8.8817e-16**
	Worst	1.5099e-14	7.9936e-15	18.9748	2.8871	20.2990	7.9936e-15	**8.8817e-16**
	AVG	1.4744e-14	4.4409e-15	14.0770	1.8222	15.2499	4.7962e-15	**8.8817e-16**
	STD	1.0935e-15	2.5774e-15	5.0115	0.6259	8.3369	1.0935e-15	**0**
F11	Best	0	0	7.8556e-07	4.9129e-08	2.1168e-04	0	**0**
	Worst	0.0293	0	180.1727	0.0489	1.0066	0	**0**
	AVG	0.0025	0	18.0668	0.0121	0.2323	0	**0**
	STD	0.0064	0	47.1423	0.0131	0.2897	0	**0**
F12	Best	0.0146	9.0633e-04	0.4232	0.9935	0.4919	**1.3462e-12**	3.7442e-10
	Worst	0.0865	0.0204	2.5600e+08	8.2201	5.6686	**2.9976e-10**	3.0196e-08
	AVG	0.0446	0.0062	1.2800e+07	5.2150	1.2479	**4.8316e-11**	4.4574e-09
	STD	0.0191	0.0057	5.7243e+07	1.8548	1.2855	**7.0460e-11**	6.9205e-09
F13	Best	0.0529	0.0622	11.5332	1.3505e-09	2.1978	**2.2486e-10**	1.2033e-08
	Worst	0.5489	0.4270	56.4995	15.7395	211.0958	**0.1084**	0.1537
	AVG	0.1948	0.2210	27.7197	0.8971	18.9215	**0.0276**	0.0370
	STD	0.1399	0.1048	11.7789	3.5174	46.8986	**0.0429**	0.0494

Figs.8-13 exhibits the CG (convergence curve) of all algorithms mentioned on multimodal benchmark issues. In Fig 8 IMEO outperforms other compared algorithms in an impressive way; as all others dropped into local optimality. For Fig 9, IMEO reaches the global optimum from the first, followed by EO and WOA in order, but others do not. In Figs 10 IMEO converges towards the best solution from the iterations beginning to the end, while all other algorithms can't. Figs.11 shows the absolute superiority of the optimized algorithm IMEO compared with all others in achieving the best solution from the begging of the iterations, followed by EO and WOA, algorithms, respectively. Both of IMEO and EO algorithms can get the optimum solution, but the other can't. Although IMEO, try to converge to the best solution in the iteration beginning, but EO converges towards the best solution better than IMEO at the end in both F12 and F13.

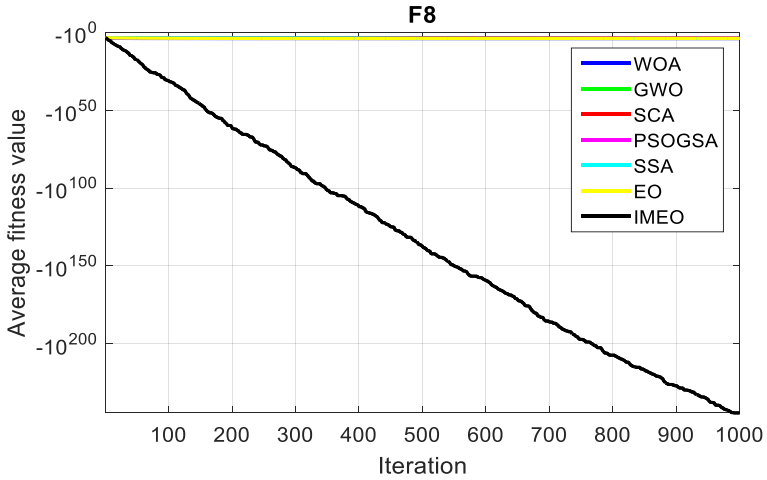

Fig.8: Average fitness value for F8

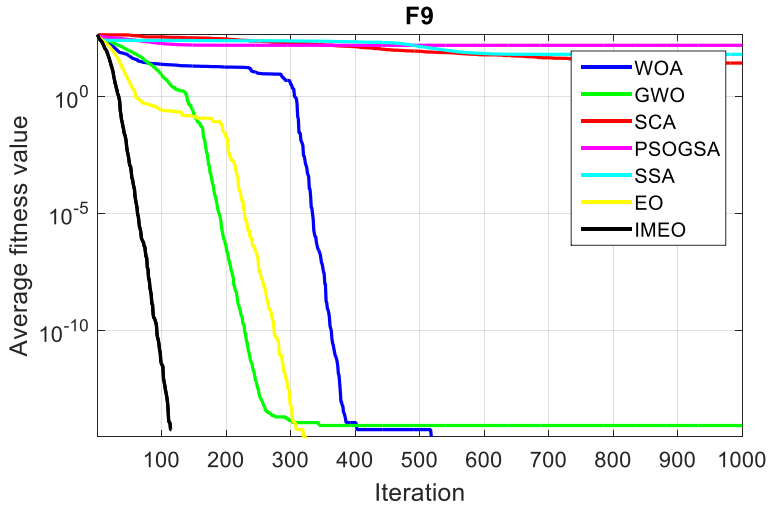

Fig.9: Average fitness value for F9

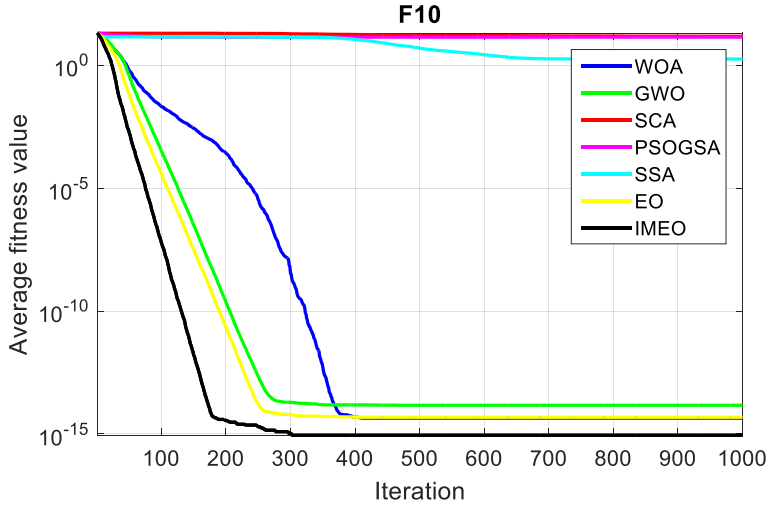

Fig.10: Average fitness value for F10

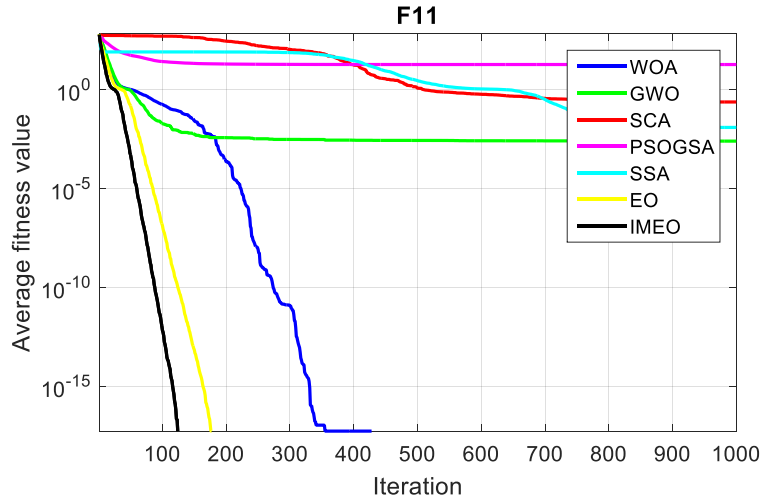

Fig.11: Average fitness value for F11

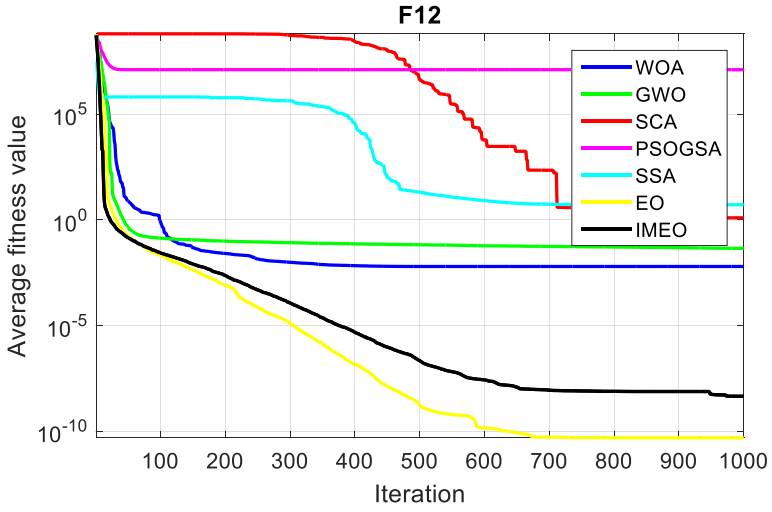

Fig.12: Average fitness value for F12

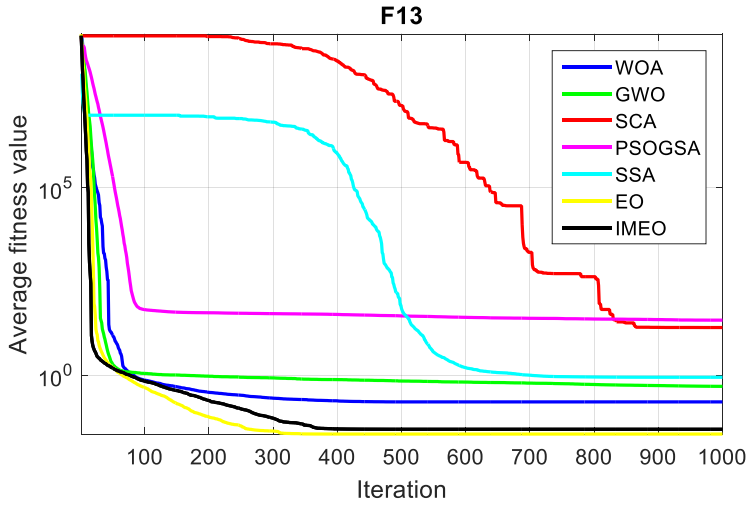

Fig.13: Average fitness value for F13

Table 7 simulates the statistical visualization results of the algorithms mentioned in fixed multimodal issues. Analytical results show that IMEO is better off in many of fixed multimodal problems. In F14, we find that the all algorithms have been approaching the optimum solution, but according to mean value EO is the first followed by IMEO. Although the WOA and SSA accelerate towards the best solution algorithm attempts to enter the best solution at the first evaluation of F15, it fails to do so, while the improved algorithm, is the only one who can find the best solution at all. For F16, although the enhanced algorithm, like its peers, achieves the best solution, it ranks first in favor of statistical results. Moving to F17, we find that three algorithms reached to the optimum solutions, IMEO, EO, and PSO-GSA. Working on the function 18, we showed that all algorithms were able to reach the optimal solution, but the EO algorithm's solution came in first, while the IMEO algorithm came in second, for the standard deviation values, of course. IMEO followed by PSO-GSA in achieving the best solution in F19. IMEO ranks first in favor of statistical results in F20. For F21, EO outperforms its peers according to Best, Worst, Mean, and SD values, followed by IMEO. According to the results, GWO is the best algorithm, reaching to best solution in F22 and F23

Table 7 Fixed multimodal benchmark evaluations

Fun		GWO	WOA	PSO -GSA	SSA	SCA	EO	IMEO
F14	Best	0.9880	0.9880	0.9980	0.9880	0.9880	0.9880	0.9880
	Worst	10.7631	10.7632	18.3034	0.9880	2.9821	0.9880	0.9880
	AVG	2.5204	3.5463	4.8475	0.9880	1.4955	0.9880	0.9880
	STD	2.9296	3.8057	5.2710	1.9060e-16	0.8805	**1.4408e-16**	1.6109e-16
F15	Best	3.0748e-04	3.0916e-04	3.5828e-04	5.5385e-04	4.2929e-04	3.0749e-04	**3.0748e-04**
	Worst	0.0204	0.0016	0.0204	0.0012	0.0015	0.0204	**3.1337e-04**
	AVG	0.0024	7.1081e-04	0.0037	8.8492e-04	0.0010	0.0015	**3.0875e-04**
	STD	0.0061	4.2348e-04	0.0072	2.6698e-04	3.8816e-04	0.0045	**1.4486e-06**
F16	Best	-1.0316	-1.0316	-1.0316	-1.0316	-1.0316	-1.0316	**-1.0316**
	Worst	-1.0316	-1.0316	-1.0316	-1.0316	-1.0316	-1.0316	**-1.0316**
	AVG	-1.0316	-1.0316	-1.0316	-1.0316	-1.0316	-1.0316	**-1.0316**
	STD	5.6753e-09	2.5063e-11	1.8366e-16	1.5459e-14	2.3164e-05	2.1612e-16	**2.1004e-16**
F17	Best	0.3979	0.3979	0.3979	0.3979	0.3979	0.3979	**0.3979**
	Worst	0.3982	0.3979	0.3979	0.3979	0.4018	0.3979	**0.3979**
	AVG	0.3979	0.3979	0.3979	0.3979	0.3990	0.3979	**0.3979**

	STD	7.7697e-05	2.8389e-06	0	2.1194e-14	0.0010	0	**0**
F18	Best	3.0000	3.0000	3.0000	3.0000	3.0000	3.0000	3.0000
	Worst	3.0001	3.0000	3.0000	3.0000	3.0000	3.0000	3.0000
	AVG	3.0000	3.0000	3.0000	3.0000	3.0000	3.0000	3.0000
	STD	5.0249e-05	3.5121e-05	1.3592e-15	7.8572e-14	1.7232e-05	**5.4865e-16**	8.8817e-16
F19	Best	-3.8628	-3.8626	-3.8628	-3.8628	-3.8627	-3.8628	**-3.8628**
	Worst	-3.8582	-3.8493	-3.8628	-3.8628	-3.8532	-3.8628	**-3.8628**
	AVG	-3.8617	-3.8595	-3.8628	-3.8628	-3.8560	-3.8628	**-3.8628**
	STD	0.0018	0.0037	2.1729e-15	2.8341e-14	0.0032	2.2157e-15	**2.1297e-15**
F20	Best	-3.3219	-3.3218	-3.3220	-3.3220	-3.2204	-3.3220	**-3.3220**
	Worst	-2.4317	-3.0152	-3.1345	-3.1951	-2.9950	-3.2	**-3.2031**
	AVG	-3.1493	-3.2582	-3.2591	-3.2313	-3.0472	-3.2736	**-3.2804**
	STD	0.2770	0.0944	0.0662	0.0583	0.0670	0.0700	**0.0582**
F21	Best	-10.1530	-10.1530	-10.1532	-10.1532	-6.6050	**-10.1532**	-10.1532
	Worst	-5.0552	-2.6303	-2.6305	-2.6305	-0.4973	**-5.0552**	-5.0552
	AVG	-9.1400	-8.7344	-5.2608	-8.3931	-3.1572	**-9.8983**	-9.3885
	STD	2.0782	2.5180	3.0845	2.8335	2.1278	**1.1399**	1.8676
F22	Best	**-10.4028**	-10.4027	-10.4029	-10.4029	-5.4310	-10.4029	-10.4029
	Worst	**-10.4019**	-5.0876	-1.8376	-2.7659	-0.9046	-5.0877	-5.0877
	AVG	**-10.4024**	-8.3388	-4.6460	-8.5844	-3.7175	-9.6057	-8.8084
	STD	**2.7248e-04**	2.1808	3.1145	2.9179	1.7681	1.9472	2.4990
F23	Best	**-10.5363**	-10.5364	-10.5364	-10.5364	-8.6596	-10.5364	-10.5364
	Worst	**-10.5357**	-1.6765	-1.8595	-2.8711	-0.9460	-5.1285	-5.1285
	AVG	**-10.5361**	-6.9300	-5.1203	-9.6171	-4.731	-10.2660	-9.7252
	STD	**1.8190e-04**	3.4803	3.6735	2.2864	1.5577	1.2092	1.9812

Looking at Fig.14, we find that both EO and IMEO converges to the best solution from the first iterations followed by for solving F14, but EO is the fastest according to standard deviation value. Fig.15 shows the superiority of IMEO as it can only converges towards best solution, The IMEO algorithm alone can come close to achieving the best solution, whereas others will never get close because they will fall into local optima. Fig.16 and Fig.18 show that all algorithms can achieve the optimum solution, but IMEO and EO converge from the beginning faster than they do. For Fig 17, IMEO reach to the optimum solution as well as EO algorithm. Fig 19 most of algorithms reach to best solution, while IMEO ranked first and PSO-GSA ranked second one. Fig.20 show that, although EO try to converge fast in the first, but IMEO is the best one between them in converging towards the best solution; thus the proposed algorithm ranked first. In F21, IMEO reaches to the optimal solution but it ranks second, while EO algorithm converges towards the best till the end of iteration followed by IMEO, WOA and GWO, respectively. Figs (22-23) show GWO's superiority in converging to the best solution.

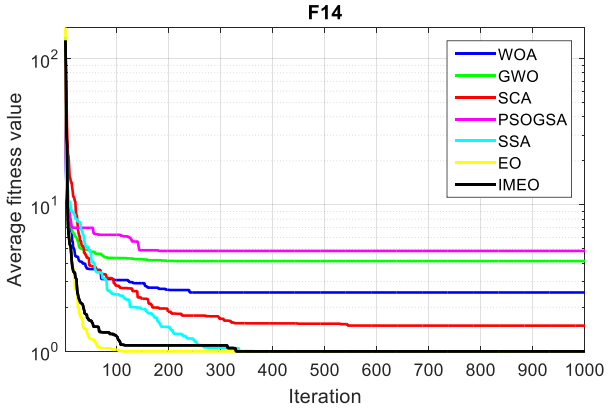

Fig.14: Average fitness value for F14

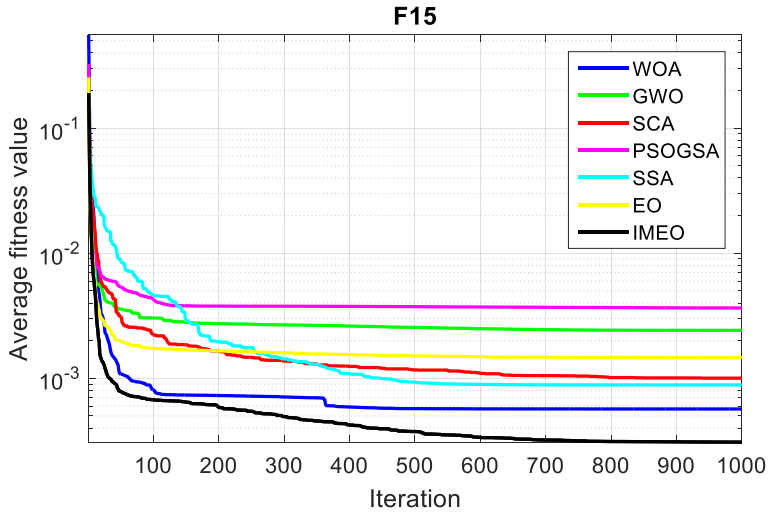

Fig15: Average fitness value for F15

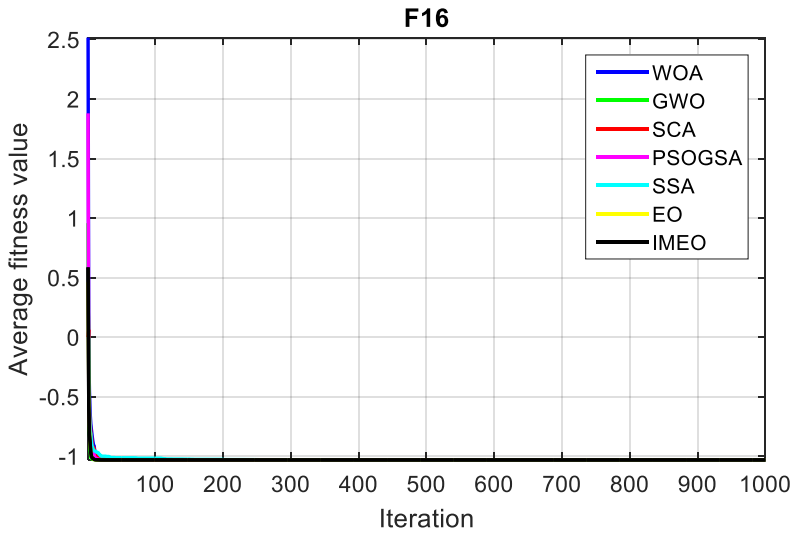

Fig16: Average fitness value for F16

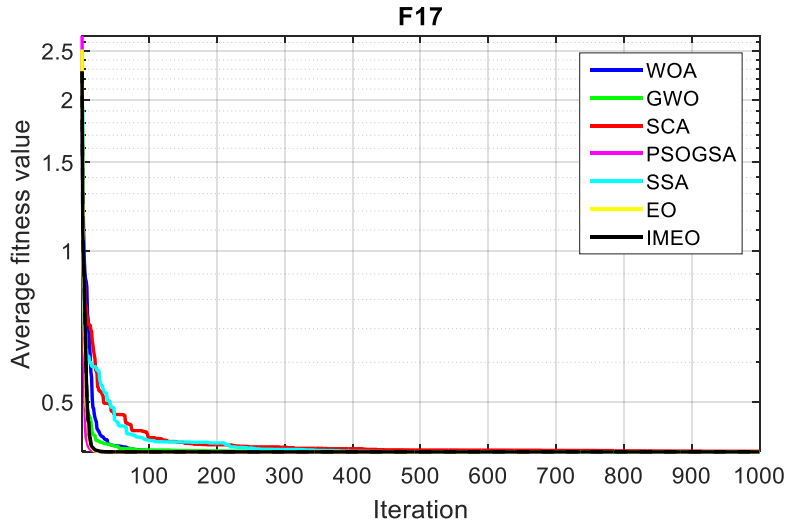

Fig 17: Average fitness value for F17

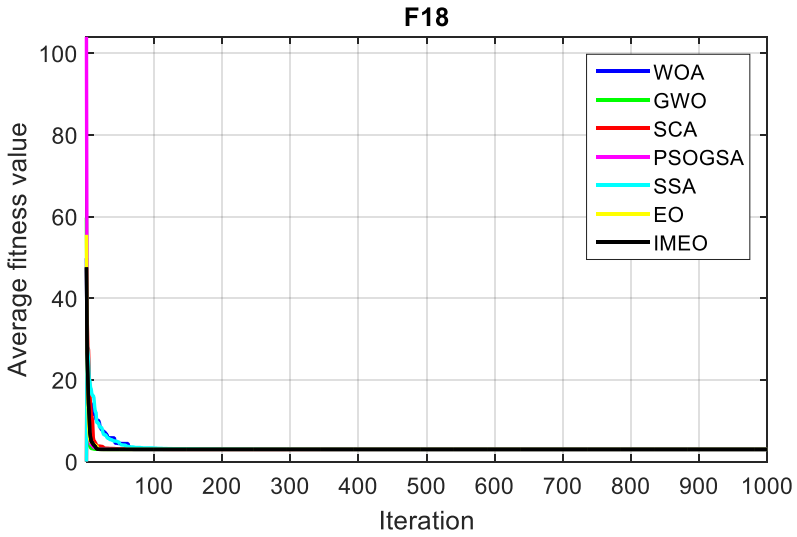

Fig 18: Average fitness value for F18

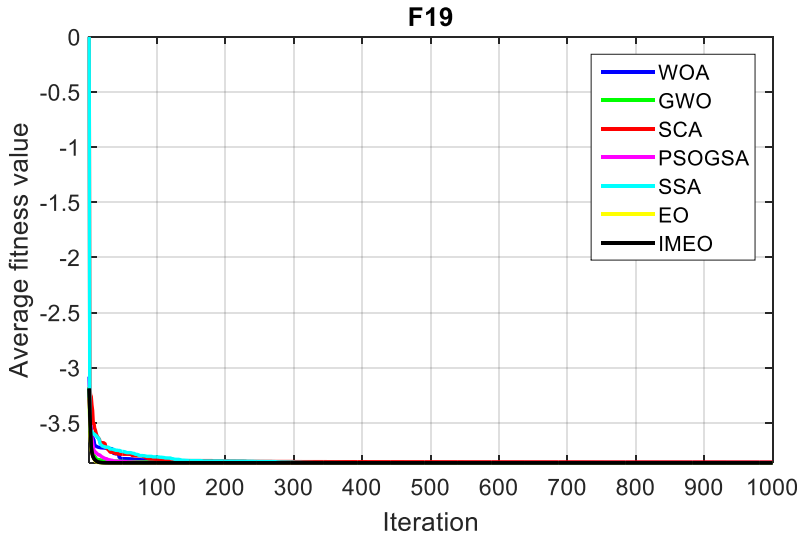

Fig 19: Average fitness value for F19

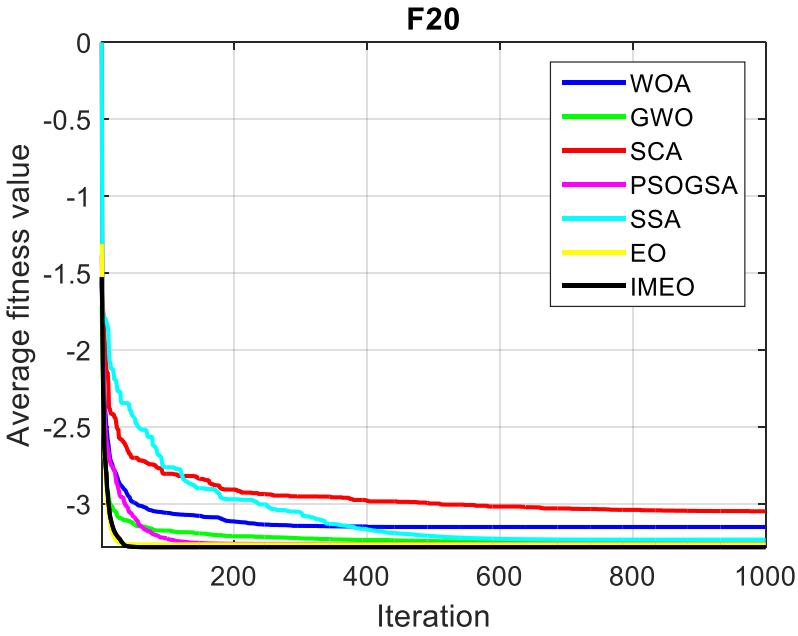

Fig 20: Average fitness value for F20

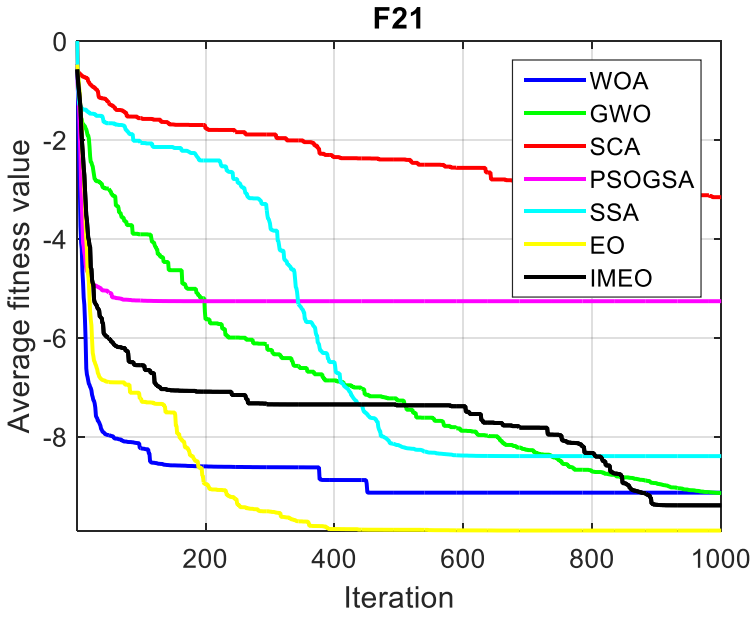

Fig 21: Average fitness value for F21

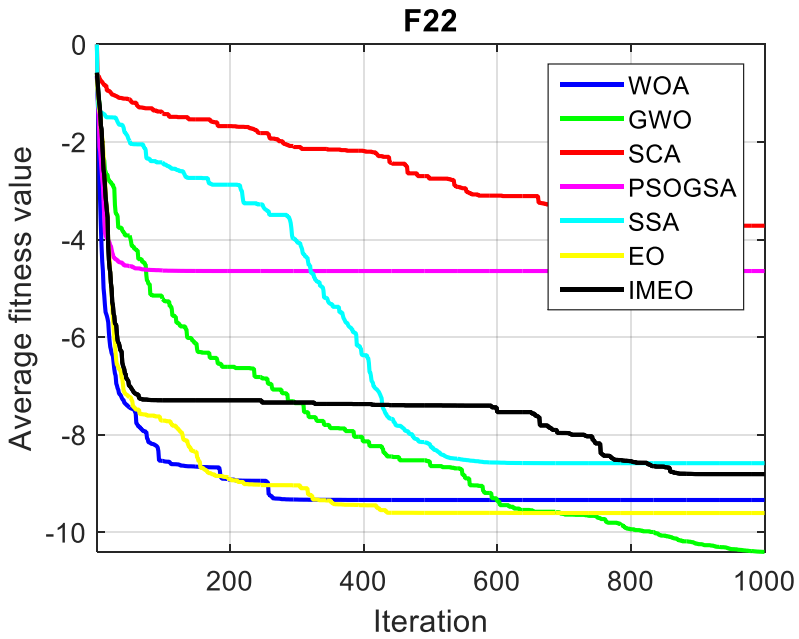

Fig 22: Average fitness value for F22

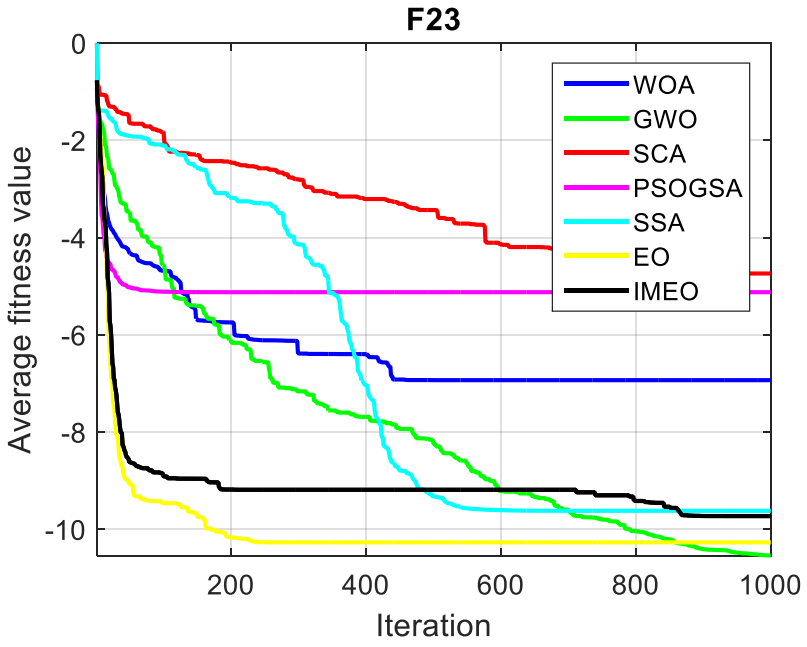

Fig 23: Average fitness value for F23

Table 8 all over statistical analysis

Fun		GWO	WOA	PSO -GSA	SSA	SCA	EO	IMEO
F1	Best	1.8238e-61	1.8644e-166	1.5772e-19	7.5194e-09	1.5083e-05	3.7503e-90	**0**
	Worst	2.4203e-58	7.3955e-148	1.0000e+04	1.6635e-08	0.1136	3.9146e-84	**0**
	AVG	1.7975e-59	3.7101e-149	2.0000e+03	1.2207e-08	0.0212	2.3919e-85	**0**
	STD	5.3556e-59	1.6534e-148	4.1039e+03	2.4280e-09	0.0338	8.6915e-85	**0**
F2	Best	1.5363e-34	3.6967e-113	1.8251e-09	4.5650e+02	4.3681e-08	5.2772e-49	**6.5896e-242**
	Worst	7.9190e-33	5.5294e-102	127.5683	1.0648e+18	0.0014	5.2014e-47	**5.4099e-236**
	AVG	1.3868e-33	5.0495e-103	12.2510	1.4075e+17	1.2999e-04	8.3220e-48	**3.8455e-237**
	STD	1.7415e-33	1.3364e-102	30.7085	3.3870e+17	3.1692e-04	1.1846e-47	**0**
F3	Best	4.1011e-20	1.6477e+03	712.4705	59.3647	106.0618	5.7850e-28	**0**
	Worst	6.5511e-12	3.9407e+04	4.0081e+04	909.9345	1.0388e+04	5.9848e-22	**0**
	AVG	3.3493e-13	2.2387e+04	1.0205e+04	361.3558	4.2940e+03	8.3382e-23	**0**
	STD	1.4632e-12	1.0928e+04	8.7801e+03	220.3436	3.2530e+03	1.7973e-22	**0**
F4	Best	1.0427e-15	2.9959	25.3828	2.3897	2.8108	2.1314e-24	**1.0847e-223**
	Worst	1.6650e-13	85.8951	89.9957	14.4690	50.8192	7.9271e-21	**9.9226e-216**
	AVG	2.0119e-14	47.7184	62.0984	8.4667	21.2251	1.8151e-21	**5.6702e-217**
	STD	4.2154e-14	32.5033	25.4474	3.6454	13.1247	2.7578e-21	**0**
F5	Best	25.6359	26.3410	15.7772	22.2042	27.9390	24.4147	**24.0796**
	Worst	28.5193	28.7400	8.0032e+07	1.6582e+03	1.1155e+03	25.2220	**24.7565**
	AVG	26.7128	27.3463	1.2000e+07	242.9221	151.8250	24.8063	**24.5063**
	STD	0.7063	0.6574	2.9296e+07	446.1293	253.1963	0.1775	**0.1743**
F6	Best	1.9717e-05	0.5873	1.4035e-19	8.5409e-09	3.9938	**3.2366e-10**	5.0956e-09
	Worst	1.2549	0.3944	1.0100e+04	1.7352e-08	5.5375	**5.2880e-08**	7.7155e-07
	AVG	0.5336	0.0776	2.0000e+03	1.2460e-08	4.5791	**6.4819e-09**	1.4283e-07
	STD	0.3880	0.1309	4.1042e+03	2.5780e-09	0.8959	**1.2385e-08**	2.4026e-07
F7	Best	2.3153e-04	2.0250e-05	0.0382	0.0415	0.0035	3.7536e-04	**4.6102e-06**

	Worst	0.0013	0.0049	0.1487	0.1541	0.1306	0.0016	**3.0062e-04**
	AVG	7.3252e-04	0.0012	0.0798	0.1004	0.0313	7.7244 e-04	**1.1991e-04**
	STD	3.4343e-04	0.0013	0.0295	0.0370	0.0291	3.3244e-04	**8.0394e-05**
F8	Best	-7.6072e+03	-1.2569e+04	-8.945e+03	-8.2841e+03	-4.5240e+03	-1.0211e+04	**-7.1349e+241**
	Worst	-4.6065e+03	-8.7377e+03	-5.5040e+03	-6.0416e+03	-3.4682e+03	-7.1334e+03	**-2.6003e+207**
	AVG	-5.8652e+03	-1.2091e+04	-7.2012e+03	-7.5161e+03	-3.9062e+03	-8.9686e+03	**-3.5812e+240**
	STD	7.6126e+02	1.1275e+03	8.8465e+02	5.4920e+02	2.9219e+02	7.8889e+02	**Inf**
F9	Best	0	0	107.4551	23.8790	8.8590e-05	0	**0**
	Worst	5.6843e-14	0	195.0107	107.4552	121.7558	0	**0**
	AVG	8.5265e-15	0	153.0878	62.8315	26.6900	0	**0**
	STD	2.0824e-14	0	25.3416	22.6247	32.4557	0	**0**
F10	Best	1.1546e-15	8.8818e-16	3.6286	1.8551e-05	3.2430e-04	4.4409e-15	**8.8817e-16**
	Worst	1.5099e-14	7.9936e-15	18.9748	2.8871	20.2990	7.9936e-15	**8.8817e-16**
	AVG	1.4744e-14	4.4409e-15	14.0770	1.8222	15.2499	4.7962e-15	**8.8817e-16**
	STD	1.0935e-15	2.5774e-15	5.0115	0.6259	8.3369	1.0935e-15	**0**
F11	Best	0	0	7.8556e-07	4.9129e-08	2.1168e-04	0	**0**
	Worst	0.0293	0	180.1727	0.0489	1.0066	0	**0**
	AVG	0.0025	0	18.0668	0.0121	0.2323	0	**0**
	STD	0.0064	0	47.1423	0.0131	0.2897	0	**0**
F12	Best	0.0146	9.0633e-04	0.4232	0.9935	0.4919	**1.3462e-12**	3.7442e-10
	Worst	0.0865	0.0204	2.5600e+08	8.2201	5.6686	**2.9976e-10**	3.0196e-08
	AVG	0.0446	0.0062	1.2800e+07	5.2150	1.2479	**4.8316e-11**	4.4574e-09
	STD	0.0191	0.0057	5.7243e+07	1.8548	1.2855	**7.0460e-11**	6.9205e-09
F13	Best	0.0529	0.0622	11.5332	1.3505e-09	2.1978	**2.2486e-10**	1.2033e-08
	Worst	0.5489	0.4270	56.4995	15.7395	211.0958	**0.1084**	0.1537
	AVG	0.1948	0.2210	27.7197	0.8971	18.9215	**0.0276**	0.0370
	STD	0.1399	0.1048	11.7789	3.5174	46.8986	**0.0429**	0.0494
F14	Best	0.9880	0.9880	0.9980	0.9880	0.9880	0.9880	0.9880
	Worst	10.7631	10.7632	18.3034	0.9880	2.9821	0.9880	0.9880
	AVG	2.5204	3.5463	4.8475	0.9880	1.4955	0.9880	0.9880
	STD	2.9296	3.8057	5.2710	1.9060e-16	0.8805	**1.4408e-16**	1.6109e-16
F15	Best	3.0748e-04	3.0916e-04	3.5828e-04	5.5385e-04	4.2929e-04	3.0749e-04	**3.0748e-04**
	Worst	0.0204	0.0016	0.0204	0.0012	0.0015	0.0204	**3.1337e-04**
	AVG	0.0024	7.1081e-04	0.0037	8.8492e-04	0.0010	0.0015	**3.0875e-04**
	STD	0.0061	4.2348e-04	0.0072	2.6698e-04	3.8816e-04	0.0045	**1.4486e-06**
F16	Best	-1.0316	-1.0316	-1.0316	-1.0316	-1.0316	-1.0316	**-1.0316**
	Worst	-1.0316	-1.0316	-1.0316	-1.0316	-1.0316	-1.0316	**-1.0316**
	AVG	-1.0316	-1.0316	-1.0316	-1.0316	-1.0316	-1.0316	**-1.0316**
	STD	5.6753e-09	2.5063e-11	1.8366e-16	1.5459e-14	2.3164e-05	2.1612e-16	**2.1004e-16**
F17	Best	0.3979	0.3979	0.3979	0.3979	0.3979	0.3979	**0.3979**
	Worst	0.3982	0.3979	0.3979	0.3979	0.4018	0.3979	**0.3979**
	AVG	0.3979	0.3979	0.3979	0.3979	0.3990	0.3979	**0.3979**
	STD	7.7697e-05	2.8389e-06	0	2.1194e-14	0.0010	0	**0**
F18	Best	3.0000	3.0000	3.0000	3.0000	3.0000	3.0000	3.0000
	Worst	3.0001	3.0000	3.0000	3.0000	3.0000	3.0000	3.0000
	AVG	3.0000	3.0000	3.0000	3.0000	3.0000	3.0000	3.0000
	STD	5.0249e-05	3.5121e-05	1.3592e-15	7.8572e-14	1.7232e-05	**5.4865e-16**	8.8817e-16
F19	Best	-3.8628	-3.8626	-3.8628	-3.8628	-3.8627	-3.8628	**-3.8628**
	Worst	-3.8582	-3.8493	-3.8628	-3.8628	-3.8532	-3.8628	**-3.8628**
	AVG	-3.8617	-3.8595	-3.8628	-3.8628	-3.8560	-3.8628	**-3.8628**
	STD	0.0018	0.0037	2.1729e-15	2.8341e-14	0.0032	2.2157e-15	**2.1297e-15**
F20	Best	-3.3219	-3.3218	-3.3220	-3.3220	-3.2204	-3.3220	**-3.3220**
	Worst	-2.4317	-3.0152	-3.1345	-3.1951	-2.9950	-3.2	**-3.2031**
	AVG	-3.1493	-3.2582	-3.2591	-3.2313	-3.0472	-3.2736	**-3.2804**
	STD	0.2770	0.0944	0.0662	0.0583	0.0670	0.0700	**0.0582**
F21	Best	-10.1530	-10.1530	-10.1532	-10.1532	-6.6050	**-10.1532**	-10.1532
	Worst	-5.0552	-2.6303	-2.6305	-2.6305	-0.4973	**-5.0552**	-5.0552
	AVG	-9.1400	-8.7344	-5.2608	-8.3931	-3.1572	**-9.8983**	-9.3885
	STD	2.0782	2.5180	3.0845	2.8335	2.1278	**1.1399**	1.8676
F22	Best	**-10.4028**	-10.4027	-10.4029	-10.4029	-5.4310	-10.4029	-10.4029
	Worst	**-10.4019**	-5.0876	-1.8376	-2.7659	-0.9046	-5.0877	-5.0877
	AVG	**-10.4024**	-8.3388	-4.6460	-8.5844	-3.7175	-9.6057	-8.8084

	STD	**2.7248e-04**	2.1808	3.1145	2.9179	1.7681	1.9472	2.4990
F23	Best	**-10.5363**	-10.5364	-10.5364	-10.5364	-8.6596	-10.5364	-10.5364
	Worst	**-10.5357**	-1.6765	-1.8595	-2.8711	-0.9460	-5.1285	-5.1285
	AVG	**-10.5361**	-6.9300	-5.1203	-9.6171	-4.731	-10.2660	-9.7252
	STD	**1.8190e-04**	3.4803	3.6735	2.2864	1.5577	1.2092	1.9812

5. Conclusions

To avoid the disadvantages of the EO algorithm, its methodology has been improved by adding an elite opposition based learning strategy, resulting in the addition of the EO algorithm, which in turn has been applied to many optimization problems. Elite opposition strategy significantly improved the EO algorithm's exploration capability to explore most of the search area. On the other hand, when looking at the new improvement has been added an ability to IMEO to jump away from the local solution problem, which has allowed them to achieve the best solutions. From the results shown above, it appears that the improved algorithm has demonstrated superiority compared to the algorithms mentioned in most of problems. Looking at the results and the analysis, it is clear that the improved algorithm is converging faster and more stable. In future research work, the application of the improved algorithm will be extended to several applications, such as the travel salesman problem and the knapsack problem.

References

1. Abdel-Basset, M., et al., *A modified flower pollination algorithm for the multidimensional knapsack problem: human-centric decision making.* Soft Computing, 2018. **22**(13): p. 4221-4239.
2. Törn, A. and A. Zilinskas, *Global optimization.* 1989.
3. Mirjalili, S., S.M. Mirjalili, and A. Lewis, *Grey wolf optimizer.* Advances in engineering software, 2014. **69**: p. 46-61.
4. Yan, F., X. Xu, and J. Xu, *Grey wolf optimizer with a novel weighted distance for global optimization.* IEEE Access, 2020. **8**: p. 120173-120197.
5. Luo, J. and Z. Liu, *Novel grey wolf optimization based on modified differential evolution for numerical function optimization.* Applied Intelligence, 2020. **50**(2): p. 468-486.
6. Mirjalili, S., et al., *Salp Swarm Algorithm: A bio-inspired optimizer for engineering design problems.* Advances in Engineering Software, 2017. **114**: p. 163-191.
7. Nautiyal, B., et al., *Improved Salp Swarm Algorithm with mutation schemes for solving global optimization and engineering problems.* Engineering with Computers, 2021: p. 1-23.
8. Salgotra, R., et al., *Self-adaptive salp swarm algorithm for engineering optimization problems.* Applied Mathematical Modelling, 2021. **89**: p. 188-207.
9. Mirjalili, S. and A. Lewis, *The whale optimization algorithm.* Advances in engineering software, 2016. **95**: p. 51-67.
10. Chakraborty, S., et al., *A novel enhanced whale optimization algorithm for global optimization.* Computers & Industrial Engineering, 2021. **153**: p. 107086.
11. Fan, Q., et al., *ESSAWOA: Enhanced Whale Optimization Algorithm integrated with Salp Swarm Algorithm for global optimization.* Engineering with Computers, 2020: p. 1-18.
12. Mirjalili, S., *SCA: a sine cosine algorithm for solving optimization problems.* Knowledge-based systems, 2016. **96**: p. 120-133.
13. Chen, H., M. Wang, and X. Zhao, *A multi-strategy enhanced sine cosine algorithm for global optimization and constrained practical engineering problems.* Applied Mathematics and Computation, 2020. **369**: p. 124872.
14. Gupta, S., K. Deep, and A.P. Engelbrecht, *A memory guided sine cosine algorithm for global optimization.* Engineering Applications of Artificial Intelligence, 2020. **93**: p. 103718.

15. Gupta, S., et al., *A modified sine cosine algorithm with novel transition parameter and mutation operator for global optimization.* Expert Systems with Applications, 2020. **154**: p. 113395.
16. Mirjalili, S. and S.Z.M. Hashim. *A new hybrid PSOGSA algorithm for function optimization*. in *2010 international conference on computer and information application*. 2010. IEEE.
17. Dhiman, G., et al., *EMoSOA: a new evolutionary multi-objective seagull optimization algorithm for global optimization.* International Journal of Machine Learning and Cybernetics, 2021. **12**(2): p. 571-596.
18. Dhiman, G., et al., *A novel algorithm for global optimization: Rat swarm optimizer.* Journal of Ambient Intelligence and Humanized Computing, 2020: p. 1-26.
19. Çelik, E., *A powerful variant of symbiotic organisms search algorithm for global optimization.* Engineering Applications of Artificial Intelligence, 2020. **87**: p. 103294.
20. Hu, Z., et al., *Grey prediction evolution algorithm for global optimization.* Applied Mathematical Modelling, 2020. **79**: p. 145-160.
21. Rodrigues, D., et al., *Adaptive improved flower pollination algorithm for global optimization*, in *Nature-Inspired Computation in Data Mining and Machine Learning*. 2020, Springer. p. 1-21.
22. Hu, K., et al., *A modified butterfly optimization algorithm: An adaptive algorithm for global optimization and the support vector machine.* Expert Systems, 2020: p. e12642.
23. Kaur, S., et al., *Tunicate swarm algorithm: a new bio-inspired based metaheuristic paradigm for global optimization.* Engineering Applications of Artificial Intelligence, 2020. **90**: p. 103541.
24. Faramarzi, A., et al., *Equilibrium optimizer: A novel optimization algorithm.* Knowledge-Based Systems, 2020. **191**: p. 105190.
25. Gao, Y., Y. Zhou, and Q. Luo, *An efficient binary equilibrium optimizer algorithm for feature selection.* IEEE Access, 2020. **8**: p. 140936-140963.
26. Menesy, A.S., H.M. Sultan, and S. Kamel. *Extracting model parameters of proton exchange membrane fuel cell using equilibrium optimizer algorithm*. in *2020 International Youth Conference on Radio Electronics, Electrical and Power Engineering (REEPE)*. 2020. IEEE.
27. Abdel-Basset, M., et al., *Solar photovoltaic parameter estimation using an improved equilibrium optimizer.* Solar Energy, 2020. **209**: p. 694-708.
28. Rabehi, A., et al., *Optimal estimation of Schottky diode parameters using a novel optimization algorithm: Equilibrium optimizer.* Superlattices and Microstructures, 2020. **146**: p. 106665.
29. Agnihotri, S., A. Atre, and H. Verma. *Equilibrium optimizer for solving economic dispatch problem*. in *2020 IEEE 9th Power India International Conference (PIICON)*. 2020. IEEE.
30. Micev, M., M. Ćalasan, and D. Oliva, *Design and robustness analysis of an Automatic Voltage Regulator system controller by using Equilibrium Optimizer algorithm.* Computers & Electrical Engineering, 2021. **89**: p. 106930.
31. Wunnava, A., et al., *A novel interdependence based multilevel thresholding technique using adaptive equilibrium optimizer.* Engineering Applications of Artificial Intelligence, 2020. **94**: p. 103836.
32. Zhou, Y., R. Wang, and Q. Luo, *Elite opposition-based flower pollination algorithm.* Neurocomputing, 2016. **188**: p. 294-310.
33. Mohamed, A.-A.A., et al., *Optimal power flow using moth swarm algorithm.* Electric Power Systems Research, 2017. **142**: p. 190-206.