Metal ratio optimization method for training of feed-forward neural networks

Harshit Batra[1,2], Vijay Kumar Bohat[2]

[1] *Centre of Excellence in AI, Netaji Subhas University of Technology, Dwarka, New Delhi-110078, India*
[2] *Department of Computer Science & Engineering, Netaji Subhas University of Technology, Dwarka, New Delhi-110078, India*

Abstract

Training a feed-forward neural network (FNN) presents a significant challenge. Traditional methods often rely on gradient-based techniques, which are prone to getting trapped in local optima. This study proposes a novel approach for FNN training by enhancing the Golden Ratio Optimization Method (GROM)

GROM is a recently introduced nature-inspired optimization method based on the golden ratio, which has shown good performance in solving real-world problems. One of the key strengths of GROM is its simplicity and unlike many other optimization methods, GROM does not require any tuning parameters to be set by the user. This makes it an attractive choice for researchers who want a straightforward optimization method that can be easily applied to a wide range of problems. However, GROM faces issues with slow convergence and a tendency to get stuck in local optima. To overcome these problems, a novel optimization framework named the Metal Ratio Optimization Method (MROM) has been proposed in this study. The search mechanism of GROM has been improved, and various metal ratios have been explored in the proposed framework. The MROM-based algorithms were tested on two groups of well-known benchmark test functions, and they showed improved performance compared to GROM and other gradient-based optimisation techniques. Furthermore, the efficiency of MROM in training feedforward neural networks has been evaluated and compared with several state-of-the-art optimization methods on 30 classification datasets. The results demonstrate the outstanding performance of MROM compared to other competitive optimization methods on various classification problems.

Keywords:
Feed-Forward Neural Network, Golden Ratio Optimization Method, Metal Ratio, Meta-heuristic, Optimization algorithm, Nature inspired Algorithms.

1. Introduction

Artificial Neural Networks (ANNs) (Grosan & Abraham (2011)) are highly effective machine learning tools inspired by the human brain (Fausett (2006)). Over the years, they have become a vital tool for modeling and simulating complicated nonlinear systems (Abiodun et al. (2018)). ANNs have been widely used in a range of fields including pattern recognition (Bridle (1990)), function approximation (Smith & Boning (1997)), time series prediction (Hamzaçebi (2008); Tealab et al. (2017)), classification (Chen et al. (2014); Mahmon & Ya'acob (2014)), signal processing (Azad et al. (2016)), system identification and control (Gautam (2016); Li et al. (2016)), and many others (Marugán et al. (2018); Cachim & Bezuijen (2019); Marinković et al. (2020); Lin et al. (2020)).

In general, ANNs refer to a diverse set of mathematical models that are characterized by the presence of interconnected units known as artificial neurons (Yegnanarayana (2009)). Over the past three to four decades, a plethora of ANN models have been proposed, including feed-forward neural networks (FNNs) (Bebis & Georgiopoulos (1994)), recurrent neural networks (RNNs) Sherstinsky (2020), radial basis functions (RBFs) (Buhmann (2000)), Kohonen self-organizing networks (KSONs) (Buhmann (2003)), and spiking neural networks (SNNs) (Ghosh-Dastidar & Adeli (2009)). Among these, Feedforward Neural Networks(FNNs) have garnered significant attention from machine learning researchers, due to their distinct non-cyclic structure (Svozil et al. (1997)).

FNNs have three main components, namely network architecture, optimizer/training algorithm, and activation function (Yegnanarayana (2009)). The architecture of the network determines the way neurons are connected, the optimizer/training algorithm calculates the weights and biases of the neural network, and the activation function determines the state of each neuron based on the inputs it receives.

The performance of FNNs is closely tied to both the network structure (a function) and the weights (the parameters of the function). To determine appropriate weights, it is necessary to employ the proper optimization techniques or training algorithms. Throughout the past few

Email addresses: `harshit.batra.ug20@nsut.ac.in` (Harshit Batra[1,2]), `vijay.bohat@gmail.com` (Vijay Kumar Bohat[2])

decades, researchers have employed a range of optimization techniques to refine FNN models, with gradient-based optimization being a commonly used method among early researchers in the field (Jameson (1995); Haji & Abdulazeez (2021); Ruder (2016)). However, due to limitations in gradient-based techniques (Shalev-Shwartz et al. (2017)), researchers have shifted their focus towards alternative optimization methods, such as meta-heuristic optimization (Glover & Kochenberger (2006); Chopard & Tomassini (2018)). Nature-inspired optimization algorithms (NIAs) are noted for their ability to handle complex, nonlinear, and non-differentiable problems, which traditional methods are unable to address as they require a continuous and differentiable objective function.

Over the years, a variety of Nature-inspired optimization techniques have been used to train FNNs (Ojha et al. (2017)). The first implementation was by Montana & Davis (1989) using the well-known Genetic Algorithm (GA), which was inspired by Darwin's natural selection and evolution theories. Ku et al. (1995) used a cellular genetic algorithm for training recurrent neural networks. Shaw & Kinsner (1996) introduced chaotic simulated annealing, which is effective at escaping local optima in multi-layer feed-forward neural network training. Zhang et al. (2007) proposed a hybrid particle swarm optimization (PSO)-back propagation algorithm. Kuo et al. (2010) integrated particle swarm optimization-based fuzzy neural network and artificial neural network for supplier selection. Wang et al. (2015) hybridized PSO and ABC to train FNN for abnormal brain identification. Zhang et al. (2016) developed a novel fruit classification framework using a biogeography-based optimization technique. Huang & Chou (2019) combined particle swarm optimization, gravitational search algorithm, and fuzzy rules to improve feed-forward neural network classification performance. Luo et al. (2020) proposed a deep feed-forward neural network with a genetic algorithm for estimating power use in buildings. Other optimization algorithms used for FNN training include Differential evolution (DE) (Price (2013); Bilal et al. (2020)), evolution strategy (ES) (Bilal et al. (2020)), and grey wolf optimizer(GWO) (Mohamed et al. (2015); Amirsadri et al. (2018)).

The use of NIAs for solving optimization problems is challenged by the presence of multiple tuning parameters, making it difficult to find the optimal values. These parameters play a crucial role in the success of these algorithms and therefore, reducing the number of tuning parameters has become a focus for researchers. To overcome this issue, efforts are being made to develop new optimization algorithms with fewer tuning parameters. Golden Ratio Optimization Method (GROM) is a recently introduced optimization algorithm developed by Nematollahi et al. (2020) The GROM algorithm is inspired by the golden ratio, which is a mathematical ratio that is found in many natural phenomena, including the arrangement of leaves on a stem and the spiral patterns in seashells.THe method has shown promising results in solving various optimization problems and is particularly advantageous due to its lack of tuning parameters. This eliminates the need for calibrating parameters, making it a desirable option compared to other algorithms. Researchers have applied GROM to real-world problems such as COVID-19 virus detection through improved clustering techniques (Chattopadhyay et al. (2021)), optimal power flow management in a power network with stochastic renewable energy resources (Nusair & Alasali (2020)), and speech emotion recognition through a hybrid meta-heuristic feature selection method (Dey et al. (2020)).

However, GROM suffers form various limitations including limited exploration potential, slow convergence, and a tendency to get stuck in local optima. Nematollahi et al. (2020) only considered the Golden ratio in the formulation of GROM leaving other metal ratios unexplored. another issue with GORM is the utilization of only the best solution for the updation of GROM's search agent which results in a potential drawback of being prone to local optima, a phenomenon where the agent may converge towards suboptimal solutions in the vicinity of the current position, impeding its ability to explore the entire solution space. Hence to overcome this a new search mechanism is introduced to GROM. This study aims to address these limitations by proposing a modification to GROM and evaluating various metal ratios. The results of the performance evaluation show that the proposed modification offers a better balance between exploration and exploitation, especially for the training of FNNs. The primary objective of this research paper is twofold.

- Firstly, a novel MROM framework is introduced and its performance is evaluated against the GROM approach using standard benchmark function problems.
- Secondly, the effectiveness of the MROM framework is analyzed through a comparative study with other state-of-the-art metaheuristics for the training of feed-forward neural networks (FNN) problem.

The rest of the paper is organized as follows. The next section describes the problem of training FNNs. Section 3 introduces the Golden Ratio optimization method, while Section 4 presents the proposed Metal Ratio optimization method (MROM). Section 5 explains how to train a Feed Forward Neural Network using MROM. In Section 6, the experimental results on 23 well-known benchmark functions (Ling et al. (2017); Nabil (2016)) and CEC 2014 test suite (Liang et al. (2013)) are presented. Also, a comparative performance analysis of our proposed framework with its peers for the problem of training FNNs has been provided. Finally, the discussion is concluded in the last section.

2. Feedforward Neural Networks

Feedforward neural network (FNN) has a special place among neural network designs because of their straightforward architecture, which is composed of multiple neurons organized in layers where each layer is fully connected to the preceding one. Input is fed into the first layer (input layer) and the output is produced from the last layer (output layer). Hidden layers are situated between the input and output layers. The neurons in FNN are connected in

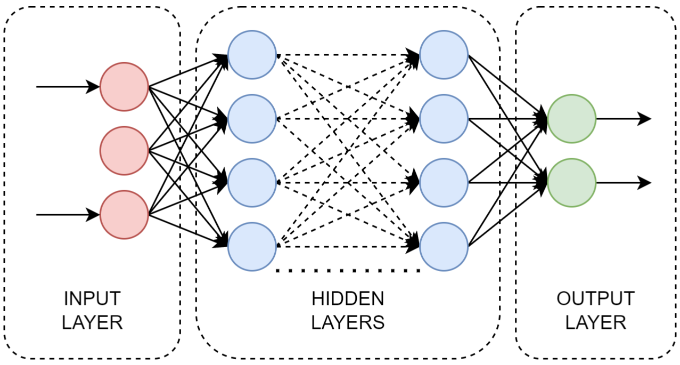


Fig. 1: A representative FNN Architecture

a unidirectional manner. Connections are represented by real numbers called weights, which lie between -1 and 1. Fig. 1 shows an example of a simple FNN architecture. It has n nodes in the input layer, m nodes in the hidden layer, and k nodes in the output layer. With the exception of the input layer, the output of each node in every layer is calculated in two steps. Firstly, the output is calculated using the weights and biases of the network, followed by an activation function that activates the network based on the weighted sum of the node's inputs. There are various activation functions (Bhoi et al. (2021); Nwankpa et al. (2018)) available in the literature, but one of the most widely used and popular activation functions is the sigmoid function. The following mathematical steps describe the FNN model: Inputs to the hidden layer node are calculated using Eq. 1.

$$S_j = \sum_{i=1}^{n}(W_{ij}.X_i) - b_j, \qquad j = 1,2,3.....,m \tag{1}$$

Here, W_{ij} denotes the weight of the connection from the i^{th} node in the input layer to the j^{th} node in the hidden layer and n is the number of input nodes. The bias of the j^{th} hidden node is represented by b_j, and the input value is represented by X_i. To calculate the output of each hidden layer node, the following calculation is performed using the sigmoid function:

$$S_j = sigmoid(s_j) = \frac{1}{(1+exp(-s_j))}, \qquad j = 1,2,3.....,m \tag{2}$$

The weighted sum of inputs to the output layer nodes is determined using the following equation:

$$o_h = \sum_{j=1}^{m}(W_{jh}.S_j) - b_j^{'}, \qquad h = 1,2,3.....,k \tag{3}$$

Here, weight of the connection between the j^{th} hidden layer node and the h^{th} output layer node is represented by W_{jh}, and the bias of the h^{th} output node is indicated by the variable $b_j^{'}$.

The final output of the FNN may be calculated using Eq. 4.

$$O_h = sigmoid(o_h) = \frac{1}{(1+exp(-o_h))}, \qquad h = 1,2,3.....,k \tag{4}$$

Since, achieving the lowest possible performance error for both training and testing data is the fundamental goal of neural network training. There are many metrics to evaluate the performance of neural networks. Mean Square Error (MSE) is a popular measure for assessing FNN. It calculates the difference between the output produced by the FNN and the actual desired output. MSE calculation is done using the following formula:

$$MSE = \sum_{i=1}^{k}(o_i^c - d_i^c)^2 \tag{5}$$

The desired output from the i^{th} neuron(output layer) of FNN when the c^{th} training sample is utilised is d_i^c, and the observed output of the i^{th} neuron(output layer) of FNN is o_i^c. Every FNN must adjust the values of its weights and biases on all of the training data-set supplied in order to realize the lowest possible error rate; hence, the effectiveness of an FNN is determined by computing its average Mean Square Error (MSE) across all of the training data as follows:

$$\overline{MSE} = \sum_{c=1}^{s}\frac{\sum_{i=1}^{k}(o_i^c - d_i^c)^2}{s} \tag{6}$$

where the quantity of training samples is represented by s. With weights, biases, and average MSE as variables, the objective function for FNN training can be written as follows:

$$Minimize \;\; F(X) = \overline{MSE} \tag{7}$$

Here, the set of vectors X stands for the weights and biases of the FNN.

3. Golden Ratio Optimization Method

The Golden Ratio Optimization Method (GROM) is a recent nature-inspired algorithm inspired by the Fibonacci series and the Golden ratio. The Fibonacci numbers, denoted as F_n, form a sequence where each number is the sum of the two preceding numbers, typically starting with 0 and 1. The golden ratio is a number frequently observed in nature. The concept of the golden ratio can be represented by Eq. 8.

$$X_n = \frac{\phi^n - (1-\phi)^n}{\sqrt{5}} \tag{8}$$

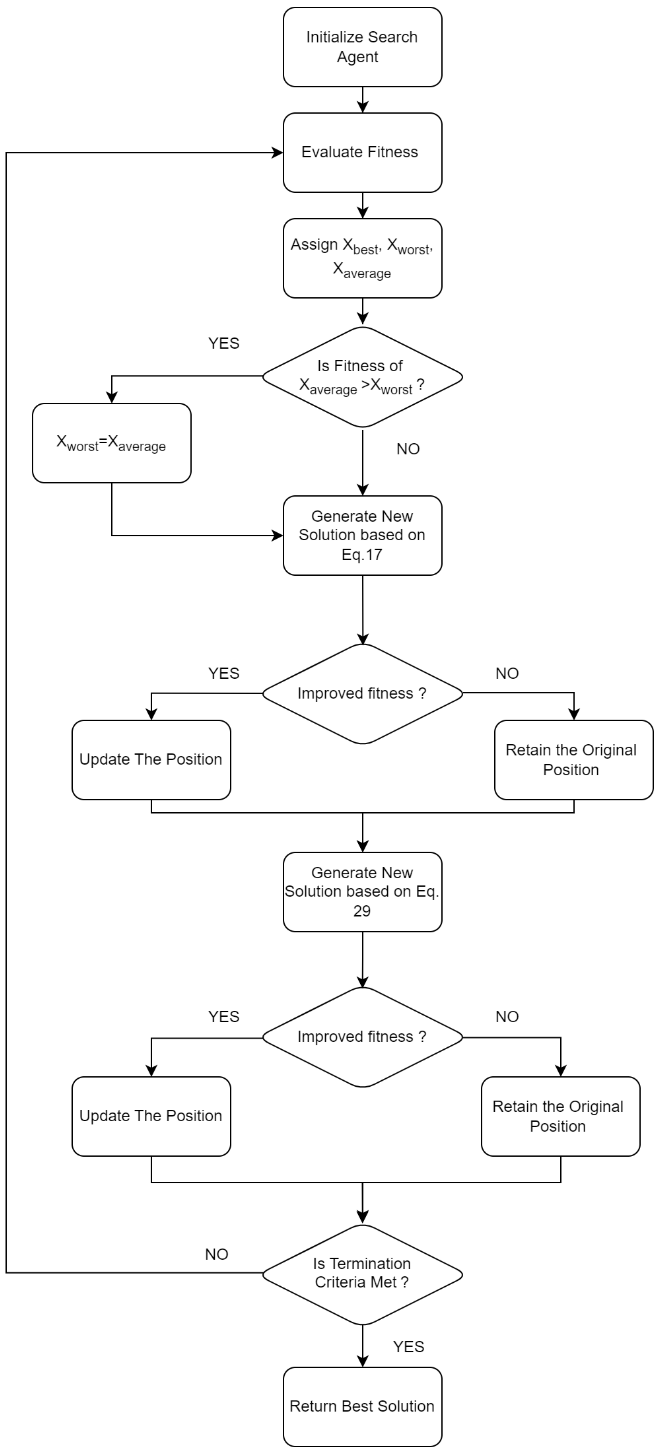

Initialize Search Agent
Evaluate Fitness
Assign X_best, X_worst, X_average
YES
Is Fitness of X_average >X_worst ?
X_worst=X_average
NO
Generate New Solution based on Eq.17
YES
Improved fitness ?
NO
Update The Position
Retain the Original Position
Generate New Solution based on Eq. 29
YES
Improved fitness ?
NO
Update The Position
Retain the Original Position
NO
Is Termination Criteria Met ?
YES
Return Best Solution

Fig. 2: Flow Chart Representing GROM

Here, X_n represents the n^{th} term of the Fibonacci series and ϕ represents the Golden ratio, which is approximately equal to 1.618034.

The GROM operates in three phases, which are described in detail in the following subsections.

3.1. First Phase

The GROM method starts with initializing search agents, with each agent's location being represented as follows:

$$X_i = x_i^1, ..., x_i^n, ..., x_i^{dim} \qquad i = 1, 2, ..., N \tag{9}$$

In the above representation, N represents the number of agents in the search space, dim is the pre-specified dimension of the given objective function, and x_i^n denotes the location of the i^{th} agent in the n^{th} dimension. To make sure that the agents are within the pre-determined search space, a boundary condition check is performed. Any agents located outside the search space are clipped back into it.

Next, the fitness of each search agent is evaluated based on the objective function. The best agent is labeled as X_{Best}, the worst agent as X_{worst}, and $X_{average}$ represents the average of the positions of all search agents. If the fitness of $X_{average}$ is found to be higher than the fitness of X_{worst}, then X_{worst} is replaced by $X_{average}$.

3.2. Second Phase

The second phase of GROM involves iterating through the population set. For each search agent X_i, a different search agent X_j is randomly selected, with $i \neq j$. Then, a comparison is made between X_i, X_j, and $X_{average}$, and the best is labeled as X_b, the worst as X_w, and the medium as X_m. The fitness relationship between X_b, X_w, and X_m is depicted in Eq. 10.

$$F_b < F_m < F_w \tag{10}$$

$$\overrightarrow{X_t} = \overrightarrow{X_m} + \overrightarrow{X_w} \tag{11}$$

To find out the extent of movement in the direction of the vector X_b obtained, the golden ratio and the Fibonacci formula are employed and presented in Eq. 12.

$$F_t = GF * \frac{\phi^T - (1-\phi)^T}{\sqrt{5}} \qquad GF = 1.1618 \tag{12}$$

$$GF = 1.1618 \tag{13}$$

$$T = t/t_{max} \tag{14}$$

The step size F_t used to update the locations of the search agents is defined by the current iteration t, the maximum number of iterations t_{max}, and the golden ratio factor GF. The objective of this position update is to enhance the best solution in the population, and a random component is added to the new solution for effective exploration of the search space. The solutions are updated stochastically using the Eq. 15. The value of F_t is calculated using Eq. 12 and the value of X_t is obtained from Eq. 11.

$$X_{new} = (1 - F_t)X_b + rand * F_t * X_t \tag{15}$$

The updated position of the search agent is only accepted if it results in an improvement in fitness as per Eq. 16. If the new position falls outside the search space, the search agent is brought back within the search space.

$$\begin{cases} X^i = X^i_{new} & If\ the\ fitness\ improves \\ X^i = X^i_{old} & otherwise \end{cases} \tag{16}$$

3.3. Third Phase

In the third phase of the algorithm, the focus shifts from avoiding the worst agent to approaching the optimum solution. The method employs the golden ratio in its approach as presented in Eq. 17.

$$X_{new} = X_{old} + rand * \frac{1}{GF} * X_{best} \tag{17}$$

The GROM method verifies the upper and lower limits of the variables once more. If a search agent is located outside the search space, it is brought back. If the new fitness is better, the previous solution is replaced by the new one as indicated in Eq. 18. The GROM method is summarized in Fig. 2 and its pseudocode is given in Algorithm 1.

$$\begin{cases} X^i = X^i_{new} & If\ the\ fitness\ improves \\ X^i = X^i_{old} & otherwise \end{cases} \tag{18}$$

Even though GROM outperforms many of the state-of-the-art algorithms. However, it has the tendency to prematurely converge on a local optimal solution. Also, the GROM framework has only explored the golden ratio. Hence, to address these concerns the next section presents an enhanced framework based on GROM exploring various metal ratio's.

4. Metal Ratio Optimization Method

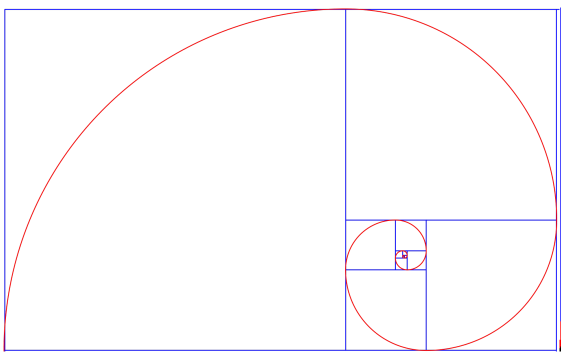
Fig. 3a displays a logarithmic spiral that has the Golden ratio as its growth factor. According to the standard GROM search mechanism, the ability of the search agent to avoid local optima is determined by the value of the Golden ratio. Fig. 3b provides a logarithmic spiral that has the Silver ratio as its growth factor. From Fig. 3a and 3b, it can be seen that different metal ratios result in different balances between exploration and exploitation. Hence, it is worthwhile to study various metal ratios within the context of GROM. Fig. 4 illustrates different logarithmic spirals that are based on the growth rate of various metal ratios.

The MROM framework proposed in this section offers two significant advancements over the GROM method:

Instead of relying solely on the Golden ratio, the MROM framework explores a range of metal ratios to weigh the

Algorithm 1 Golden Ratio Optimization Method (GROM)

```
1: Initialize population, set maximum number of iterations as $t_{total}$
2: Evaluate the fitness $f(X_i)$ for each search agent $X_i$
3: Set the best solution to be $X_{best}$, worst solution to be $X_{worst}$, and arithmetic mean of the population to be $X_{average}$
4: if Fitness of $X_{average} < X_{worst}$ then
5:     $X_{worst} = X_{best}$
6: end if
7: Check for Boundary condition
8: $t = 0$
9: while $t < t_{total}$ do
10:     for $i = 1$ to $N$ (where $N$ is population size) do
11:         randomly select $X_j$ where $i \neq j$
12:         Among $X_i$, $X_j$, and $X_{average}$, set the best solution as $X_b$, worst solution as $X_w$, and middle solution designated as $X_m$, respectively.
13:         Update the search agent's location based on the Eqs. 11-15
14:     end for
15:     for $i = 1$ to $N$ (where $N$ is population size) do
16:         Update the search agent's location based on the Eqs. 17-18
17:     end for
18:     Check for Boundary condition
19:     $t = t + 1$
20: end while
21: Output the best solution
```


(a) Golden Ratio Spiral

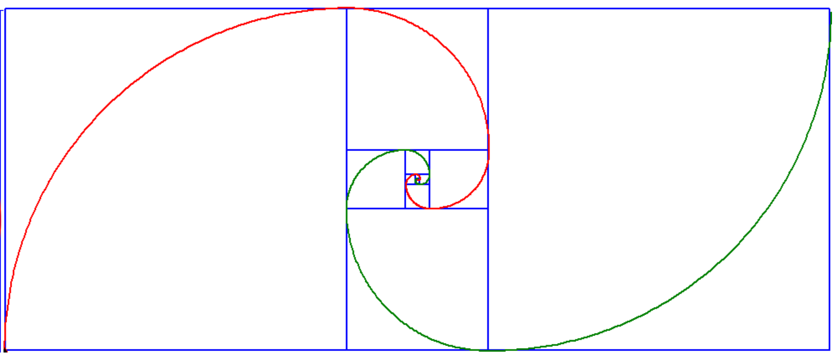

(b) Silver Ratio Spiral

Fig. 3: Spiral for Golden and silver ratio

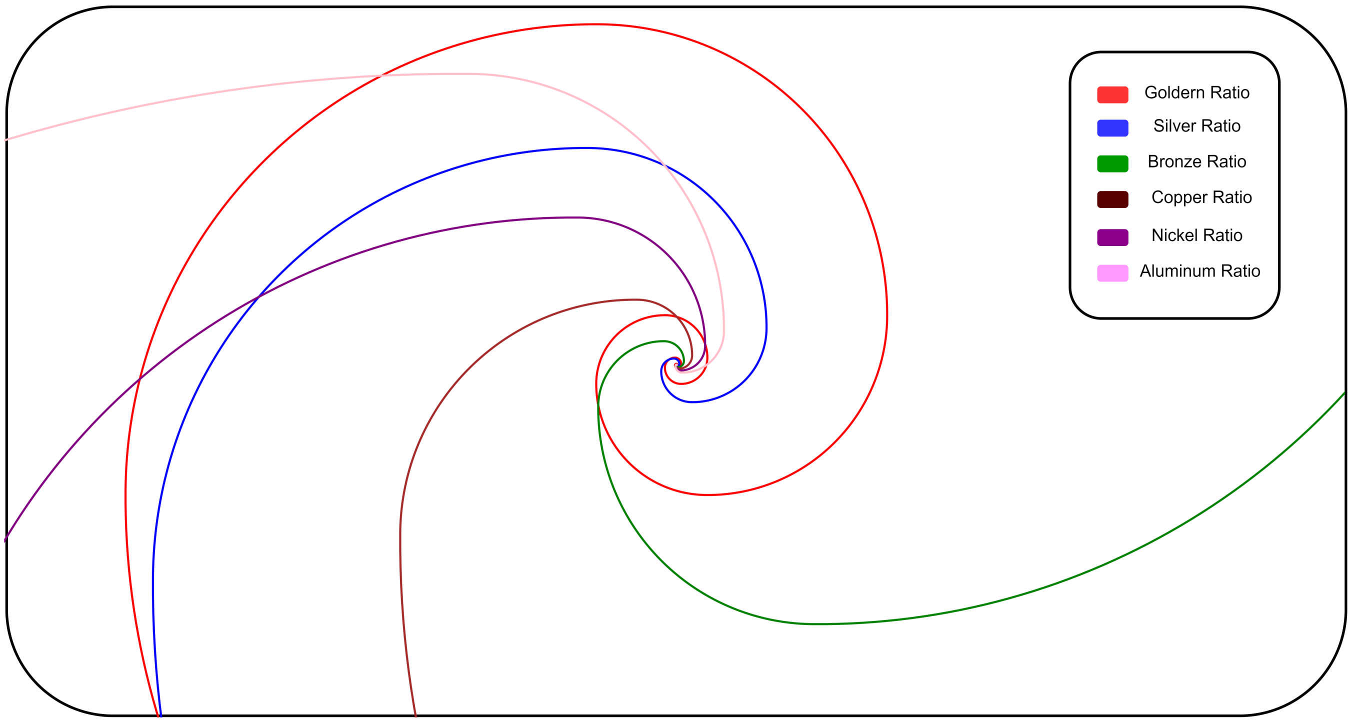


Fig. 4: Spiral for different metal ratios, Red for Golden ratio, Blue for Silver ratio, Green for Bronze Ratio, Brown for Copper Ratio, Purple for Nickel Ratio, and Pink for Aluminum Ratio

balance between exploration and exploitation. These metal ratios are displayed in Table 4.

MROM framework also introduces an enhanced search mechanism to address the issue of premature convergence present in GROM. This improved approach allows MROM to thoroughly search the problem search space and effectively find the optimal solution. A phase-wise description of the algorithm is provided in the following subsections.

4.1. Phase 1

Phase one of the proposed MROM framework resembles the GROM method. MROM begins with the initialization of search agents, each agent's location in the search space is indicated as follows:

$$X_i = x_i^1, ..., x_i^n, ..., x_i^{dim} \qquad i = 1, 2, ..., N \quad (19)$$

In Eq. 19 N depicts the number of agents in the search space, dim is a pre-specified dimension of the given objective function, and x_i^n is the location of the i^{th} agent in the n^{th} dimension.

The X_{Best}, X_{worst}, and $X_{average}$ are determined in similar way to GROM.

4.2. Phase 2

The second stage of MROM Framework, similar to GROM, starts by iterating on each search agent. For each search agent X_i, another agent X_j is chosen randomly, ensuring that $i \neq j$. The three agents, X_i, X_j, and $X_{average}$, are then compared to determine the best (X_b), worst (X_w), and medium (X_m) agents. The fitness relationship between X_b, X_w, and X_m is expressed in Eq. 20. Eq. 21 and 22 provide the movement vectors $\overrightarrow{X_t}$ and $\overrightarrow{X_{t1}}$ based on the values of X_b, X_w, and X_m. These vectors will be utilized to update the position of each search agent.

$$F_b < F_m < F_w \quad (20)$$

$$\overrightarrow{X_t} = \overrightarrow{X_m} + \overrightarrow{X_w} \quad (21)$$

$$\overrightarrow{X_{t1}} = \overrightarrow{X_b} - \overrightarrow{X_m} \quad (22)$$

Unlike the GROM, a generalized Fibonacci's formula is used in MROM framework presented through Eq. 23-25.

$$F_t = MR * \frac{MR^T - (m - MR)^T}{\sqrt{m^2 + 4}} \quad (23)$$

$$MR = (m + \sqrt{(m^2 + 4)}))/2 \quad (24)$$

$$T = t/t_{max} \quad (25)$$

Here, MR is the metal ratio, m is the metal ratio number, t_{max} is the maximum number of iterations, and t is the current iteration number. Fig. 7 highlights the difference in F_t value for various metal ratio.

In GROM, the new position of the search agent mainly depends on X_b can be seen through Eq. 15. However, this does not promote exploration of the search space. To balance between exploration and exploitation, the suggested method utilizes both X_b and X_{t1} with equal probability. In MROM, if the rand() is < 0.5 then use Eq. 26 else use Eq. 27.

$$X_{new} = (1 - F_t)X_b + rand * F_t * X_t \quad (26)$$

$$X_{new} = (1 - F_t)X_{t1} + rand * F_t * X_t \quad (27)$$

Fig. 5 visually highlights the difference in the updation rule for GROM and MROM. If the equation improves the fitness of the current search agent update the search agent else retain the old position as shown in Eq. 28. If the new position of the agent is found to be outside of the search space bring back the search agent to the search space.

$$\begin{cases} X^i = X^i_{new} & If\ the\ fitness\ improves \\ X^i = X^i_{old} & otherwise \end{cases} \quad (28)$$

Where X_{old} is the old position of the search agent.

4.3. Phase 3

In contrast to GROM, the third phase of MROM Framework focuses on thoroughly exploring the search space in addition to avoiding the worst agent and making a concerted effort to approach the optimal agent. Consequently, the method uses two equations i.e., Eq. 29 and 30 that are applied with equal probability to achieve the same as follows :

$$X_{new} = X_{old} + rand * \frac{1}{MF} * X_{best} \quad (29)$$

$$X_{new} = (X_{old} - X_{best}) + rand * \frac{1}{MF} * (X_{mean} - X_{worst}) \quad (30)$$

If the agent is found to be outside the search space it is brought back. If the fitness improves, the new solution would be replaced by the previous one, as seen below in Eq. 31:

$$\begin{cases} X^i = X^i_{new} & If\ the\ fitness\ improves \\ X^i = X^i_{old} & otherwise \end{cases} \quad (31)$$

Fig. 6 summarizes the GROM Method. Algorithm 2 provides the pseudo-code for the MROM method

4.4. Complexity Analysis of MROM and GROM

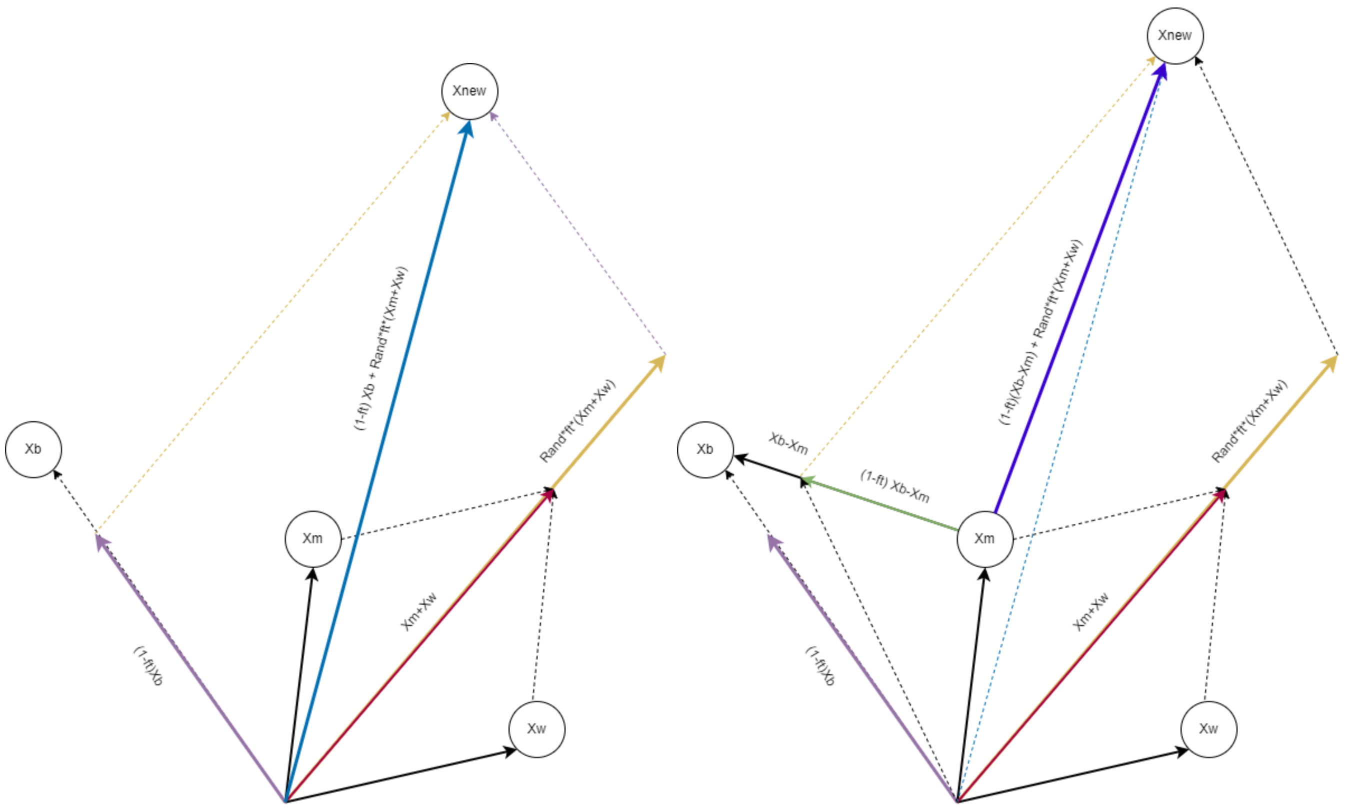
This Section explores and contrasts the computational complexity of MROM and its counterpart GROM.

4.4.1. Phase 1

Initialization of the population forms the first step of phase 1. Initialization can be achieved in $O(n \times d)$, here, n is the population size and d is the dimension of the problem. Examination of each agent's fitness forms the second step in phase 1. This is realizable in linear time for each iteration since this step is repeated for all iterations assuming that the objective function has a computational cost

Table 1: Metal ratios and their values

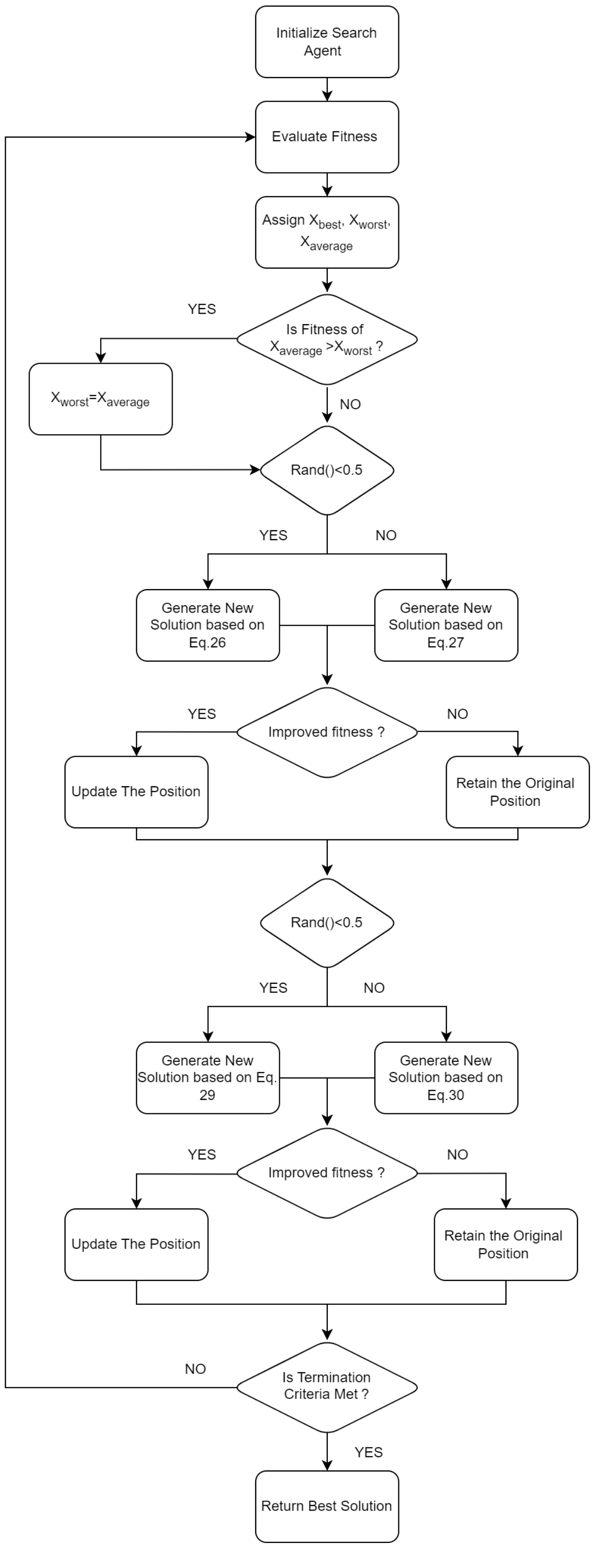
Name of Metal Ratio	**m**	**Ratio**	**Value(till 4 decimals)**
Platinum	0	$(0 + \sqrt{4})\ /\ 2$	1
Golden	1	$(1 + \sqrt{5})\ /\ 2$	1.6180
Silver	2	$(2 + \sqrt{8})\ /\ 2$	2.4142
Bronze	3	$(3 + \sqrt{13})\ /\ 2$	3.3027
Copper	4	$(4 + \sqrt{20})\ /\ 2$	4.2360
Nickel	5	$(5 + \sqrt{29})\ /\ 2$	5.1925
Aluminum	6	$(6 + \sqrt{40})\ /\ 2$	6.1622
Iron	7	$(7 + \sqrt{53})\ /\ 2$	7.1400
Tin	8	$(8 + \sqrt{68})\ /\ 2$	8.1231
Lead	9	$(9 + \sqrt{85})\ /\ 2$	9.1097



(a) Updating the search agent in GROM during Phase 2

(b) Updating the search agent in MROM during Phase 2

Fig. 5: Difference between updation rule in GROM and MROM



Initialize Search Agent
Evaluate Fitness
Assign X_{best}, X_{worst}, $X_{average}$
Is Fitness of $X_{average}$ >X_{worst} ?
YES
NO
X_{worst}=$X_{average}$
Rand()<0.5
YES
NO
Generate New Solution based on Eq.26
Generate New Solution based on Eq.27
Improved fitness ?
YES
NO
Update The Position
Retain the Original Position
Rand()<0.5
YES
NO
Generate New Solution based on Eq. 29
Generate New Solution based on Eq.30
Improved fitness ?
YES
NO
Update The Position
Retain the Original Position
Is Termination Criteria Met ?
NO
YES
Return Best Solution

Fig. 6: Flow Chart Representing MROM Algorithm

Algorithm 2 Metal Ratio Optimization Method (MROM)

```
1: Initialize population and set maximum iteration to $t_{total}$ and t = 0
2: while $t < t_{total}$ do
3:   Evaluate the fitness $f(X_i)$ for each search agent $X_i$
4:   Set the best solution as $X_{best}$, worst solution as $X_{worst}$, and arithmetic mean of the population to be $X_{average}$
5:   if fitness of $X_{average} < X_{worst}$ then
6:     $X_{worst} = X_{average}$
7:   end if
8:   Check for Boundary condition
9:   for $i$ = 1 to N (where n is population size) do
10:    randomly select $X_j$ where i!=j
11:    Set the best solution among $X_i$, $X_j$, $X_{average}$ as $X_b$, second-best solution as $X_m$, and assign third best to $X_w$, respectively
12:    if rand()<0.5 then
13:      Generate the search agent's location depending on the Eq. 26
14:    else
15:      Generate the search agent's location depending on the Eq. 27
16:    end if
17:    Verify constraints and update position based on Eq. 28
18:  end for
19:  for $i$ = 1 to N (where n is population size) do
20:    if rand()<0.5 then
21:      Generate the search agent's location depending on the Eq. 29
22:    else
23:      Generate the search agent's location depending on the Eq. 30
24:    end if
25:    Verify constraints and update position based on Eq. 31
26:  end for
27:  t = t+1
28: end while
29: Output the best solution $X_{best}$
```

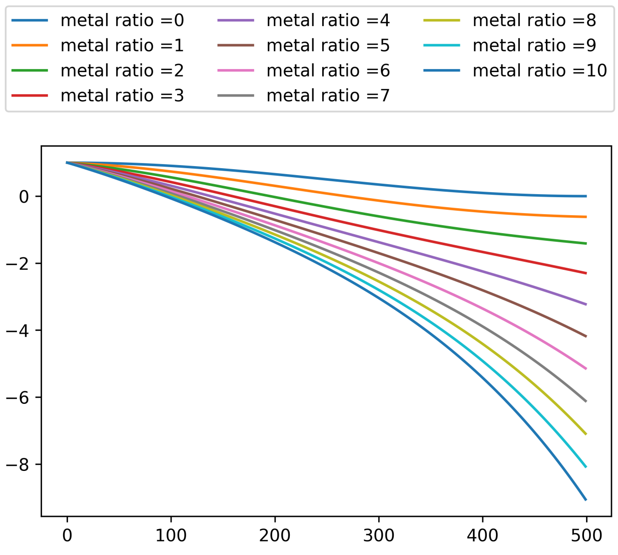


Fig. 7: Variation of $1 - F_t$ for various metal ratios

c. Therefore, the computational cost is $O(n \times c)$ for each iteration making the overall computational cost for all iterations $O(M \times (c \times n))$, where M is the maximum number of iterations. Determining the best and worst agents may be done concurrently utilizing linear time. While finding the average position of all the agents can be done in constant time. Therefore, overall the time complexity of phase 1 of both algorithms is presented in Eq. 32 and 33

$$\begin{aligned} T_{Phase\ 1\ of\ GROM} &= T_{initialization} + T_{fitness-evaluation} \\ &= O(n \times d) + O(M \times (n \times c)) \end{aligned} \tag{32}$$

$$\begin{aligned} T_{Phase\ 1\ of\ MROM} &= T_{initialization} + T_{fitness-evaluation} \\ &= O(n \times d) + O(M \times (n \times c)) \end{aligned} \tag{33}$$

4.4.2. Phase 2

Phase 2 of both GROM and MROM methods involves computation of the term F_t in Eq. 12 that determines the step size taken by the search agent. Since the value of the F_t needs to be evaluated at each iteration and evaluation is possible in constant time, hence, this operation is realizable in $O(M)$. This is followed by updating the position of the search agent. GROM algorithm produces a new position based on Eq. 15, while the MROM algorithm with equal probability employs either Eq. 26 or 27. Although, Eq. 15 and Eq. 27 are similar and hence have the same computational cost i.e., $O(n)$ whereas Eq. 26 is different but incur the same computational cost i.e., $O(n)$. Therefore this production operation has a total of $O(n)$ cost per iteration making the total cost as $O(M \times n)$. This is followed by updating the position of the search agent based on Eq. 16 or 28 for GROM and MROM, respectively. Since the equations are similar, the updation operation has a $O(c \times n)$ computational cost per iteration as it requires fitness comparison between the new and old positions. Hence, making the total cost of the operation as $O(M \times c \times n)$. The complexity of phase 2 for GROM and MROM is shown in Eq. 34 and 35

$$\begin{aligned} T_{Phase\ 2\ of\ GROM} &= T_{new\ solution} + T_{updation\ of\ solution} \\ &= O(M \times n) + O(M \times (n \times c)) \end{aligned} \tag{34}$$

$$\begin{aligned} T_{Phase\ 2\ of\ MROM} &= T_{new\ solution} + T_{updation\ of\ solution} \\ &= O(M \times n) + O(M \times (n \times c)) \end{aligned} \tag{35}$$

4.4.3. Phase 3

GROM algorithm in Phase 3 produces a new solution using the Eq. 17 which has a total computational cost of $O(n)$ per iteration, hence, the total computational cost is $O(M \times n)$. Whereas, MROM employs Eq. 29 and Eq. 30, both of which are employed with equal probability and have a $O(n)$ computational cost per iteration, hence, the total computational cost is $O(M \times n)$. The updation of the position of the search agent in GROM is done based on Eq. 31 which can be done in $O(c \times n)$ time for each iteration and $O(M \times c \times n)$ for all iterations. MROM has a similar updation rule that cause computational time taken is $O(M \times c \times n)$.

$$\begin{aligned} T_{Phase\ 3\ of\ GROM} &= T_{new\ solution} + T_{updation\ of\ solution} \\ &= O(M \times n) + O(M \times (n \times c)) \end{aligned} \tag{36}$$

$$\begin{aligned} T_{Phase\ 3\ of\ MROM} &= T_{new\ solution} + T_{updation\ of\ solution} \\ &= O(M \times n) + O(M \times (n \times c)) \end{aligned} \tag{37}$$

The final time complexity each is presented in Eq. 38 and Eq. 39 for GROM and MROM respectively

$$\begin{aligned} T_{GROM} &= T_{Phase\ 1} + T_{Phase\ 2} + T_{Phase\ 3} \\ &= O(n \times d) + O(M \times n \times c) \\ &+ O(M \times n) + O(M \times n \times c) \\ &+ O(M \times n) + O(M \times n \times c) \\ &= O(n \times d + M \times n \times c) \end{aligned} \tag{38}$$

$$\begin{aligned} T_{MROM} &= T_{Phase\ 1} + T_{Phase\ 2} + T_{Phase\ 3} \\ &= O(n \times d) + O(M \times n \times c) \\ &+ O(M \times n) + O(M \times n \times c) \\ &+ O(M \times n) + O(M \times n \times c) \\ &= O(n \times d + M \times n \times c) \end{aligned} \tag{39}$$

Table 2 presents the summary of the above complexity analysis. Its clear from the computational complexity analysis of GROM and MROM that the computational cost of both algorithms is similar.

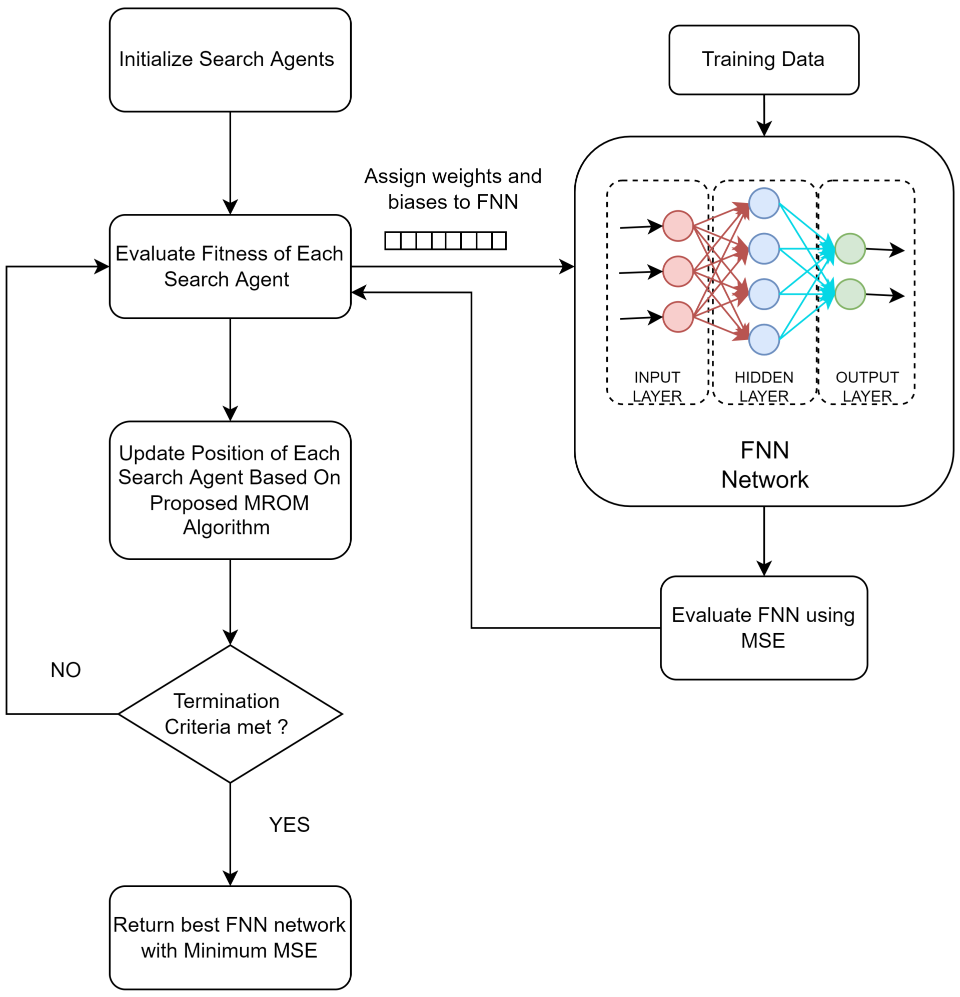

Initialize Search Agents
Training Data
Assign weights and biases to FNN
Evaluate Fitness of Each Search Agent
INPUT LAYER
HIDDEN LAYER
OUTPUT LAYER
FNN Network
Update Position of Each Search Agent Based On Proposed MROM Algorithm
Evaluate FNN using MSE
NO
Termination Criteria met ?
YES
Return best FNN network with Minimum MSE

Fig. 8: FNN Training Using MROM Algorithm

Table 2: Computational Complexity of GROM and MROM

Steps of GROM	Complexity	Steps of MROM	Complexity
Phase 1	$N \times D$	Phase 1	$N \times D$
Phase 2	$M \times N + M \times N \times C$	Phase 2	$M \times N + M \times N \times C$
Phase 3	$M \times N + M \times N \times C$	Phase 3	$M \times N + M \times N \times C$

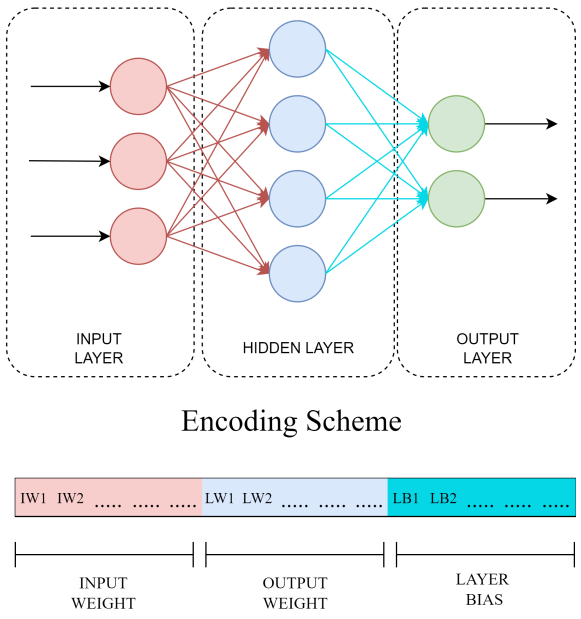


Fig. 9: Search agent formation by weights and biases

5. Training of Feed-forward Neural Networks Using Metal Ratio Optimization Method

The process of training feed-forward neural networks (FNN) involves adjusting the weights and biases to minimize the error between the predicted output and the actual output. Meta-heuristic or evolutionary methods, such as the proposed MROM, have been employed to optimize FNN training by finding the optimal weights and biases that can produce accurate predictions. However, the encoding scheme of possible solutions and the fitness function used in such methods play crucial roles in the success of FNN training.

In the proposed MROM Framework, the search agents are randomly generated within the interval $[-1, 1]$, and encoded as one-dimensional arrays, as shown in Fig. 9. The encoding vector consists of three components: weights connecting the input layer to the hidden layer, weights linking the hidden layer to the output layer, and bias terms. Each search agent is mapped to an FNN, and its components are assigned to the corresponding weights and biases of the network. While there is no consensus on the number of hidden layers in FNN. For the purpose of this study, a feed-forward neural network with one hidden layer is optimized Luo et al. (2021).

The fitness function used in the MROM Framework is the commonly used Mean Squared Error (MSE). In each learning cycle, the fitness of each search agent is evaluated by encoding it as a set of weights for the FNN, and using the training data to calculate the MSE. The objective is to minimize the value of MSE until the maximum number of iterations is reached. The MSE can be computed using Eq. 40, where y is the real class label, $\hat{y}$ is the predicted label, and n is the total number of occurrences.

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2 \tag{40}$$

The following steps present the MROM procedure for training FNN:

1. **Initialization:** The MROM procedure starts by creating a predetermined number of random search agents, which represents the initial population size for the optimization process. These search agents are generated randomly.

2. **Mapping:** Each search agent is allocated to an FNN by mapping its components to the weights and biases of a feed-forward neural network (FNN). The mapping process involves assigning each component of a search agent to a corresponding weight or bias parameter in the FNN.

3. **Evaluation:** The FNN produced in the previous step is evaluated by applying the mean squared error (MSE) criterion to all occurrences in the training dataset. The MSE measures the difference between the predicted outputs of the FNN and the actual outputs of the training data. The goal is to obtain an FNN with a minimum MSE value, indicating that it provides a more accurate fit to the training data.

4. **Update:** Based on the evaluation process in the previous step, all the search agents are updated according to the MROM algorithm. This process aims to improve the performance of the FNN by exploring different regions of the search space and exploiting promising areas for further improvement.

These steps are repeated till the termination criteria are met. Fig. 8 summarizes the FNN training process using the MROM method

6. Experimental results and discussion

The proposed optimization framework, MROM, has undergone extensive testing on two sets of benchmark functions. The first set of benchmark functions, shown in Table

A.9, is widely used by researchers in the field and has been previously studied by Singh & Bansal (2022) and Bohat & Arya (2018). In this set of benchmarks functions, MROM based algorithms was compared to GROM in terms of its efficiency in converging to an optimal solution, computational complexity, and solution quality.

To further showcase the capabilities of MROM framework based algorithms was also tested on a set of thirty challenging optimization problems from the CEC 2014 competition on Single Objective Real Parameter Optimization. This comprehensive evaluation demonstrates the overall performance of MROM.

6.1. Comparison on 23 Benchmark Functions

The set of benchmark problems includes 23 minimization functions that are grouped into three categories: unimodal functions, multi-modal functions with high dimensions, and multi-modal functions with low dimensions. The details of these benchmark functions can be found in Table 3.

To evaluate the performance of the algorithms, the results from 30 runs of each benchmark function were averaged, and the mean fitness, standard deviation, and best fitness are presented. The population size for both MROM based algorithms and GROM was set to 20 and the algorithms were run for a maximum of 500 iterations, which serves as the stopping criterion. The experimental results for both MROM and GROM on the benchmark functions are shown in Table 3.

6.1.1. Results Discussion

Category I: Unimodal functions

The results shown in Table 3 clearly demonstrate the superiority of the MROM framework algorithms over the GROM when it comes to optimizing unimodal functions from the 23-function benchmark set. This is significant, as the optimization of unimodal functions is often seen as a gauge for the exploitative abilities of optimization algorithms. In the case of unimodal functions, there are no local optima, making it crucial for optimization algorithms to efficiently locate and converge on the global optimum.

The MROM algorithm's improved exploitation capabilities are apparent in its outstanding performance in functions f_1, f_2, f_3, and f_4. In these functions, the proposed algorithms successfully identifies and converges on the global optimum solution, showcasing its ability to effectively explore the search space.

Additionally, MROM framework based algorithms outperforms GROM in functions f_6, and f_7, highlighting its adaptability to the unique characteristics of each function. The combination of MROM's effective navigation of the search space and its superior exploitation capabilities makes it a better choice over GROM.

Category II: Multi-modal Functions with High Dimensions

The functions from f_8 to f_{13} in the 23-function benchmark suite are multi-modal and large-dimensional in nature. these problems are characterized by the abundance of local optima. These functions offer considerable difficulty for any optimization technique as they require an algorithm to not only find the global optimum but also to avoid getting trapped in local optima. The results in Table 3 show that the MROM based variants outperforms the GROM in all multi-modal high-dimension functions, and has better-escaping capabilities to avoid local optimum.

One of the key strengths of the MROM framework in these functions is its ability to balance exploration and exploitation during the optimization process. The new search mechanism along with the metal ratio component provides a new trade-off between exploration and exploitation as compare to GROM, ensuring that the algorithm is able to effectively explore the search space while also exploiting the information gained from the previous iteration. This balance is crucial for global optimization in multi-modal high-dimension functions, as a purely exploitative algorithm may become trapped in local optima, while a purely exploratory algorithm may not make use of the information gained during previous iterations.

The results in Table 3 also demonstrate the stability and robustness of the MROM algorithm's in multi-modal high-dimension functions. A lower standard deviation combined with excellent mean fitness value shows the proposed framework ability to find near-optimal solutions as well as consistency in repeating superior results over multiple iterations.

Category III: Multi-modal Functions with Low Dimensions

The range of low-dimensional multi-modal functions is from f_{14} to f_{23}. These functions have fewer local optima as compared to category II functions, but they can still be employed to evaluate the robustness of optimization techniques. The algorithms based on the proposed MROM framework exhibit better performance than their GROM counterparts across all functions. This is evident in the superior outcomes produced by the suggested framework, which demonstrates the greater robustness of the MROM framework over GROM.

6.1.2. Convergence analysis

Through convergence analysis, researchers can better understand the algorithm's behavior and fine-tune its parameters to achieve improved performance. Furthermore, the convergence analysis can facilitate a comparative analysis of the performance of different optimization algorithms for a given optimization problem. The convergence curves of selected benchmark functions are shown in Fig. 10 illustrating how rapidly and conclusively MROM based algorithms converges in comparison to GROM.

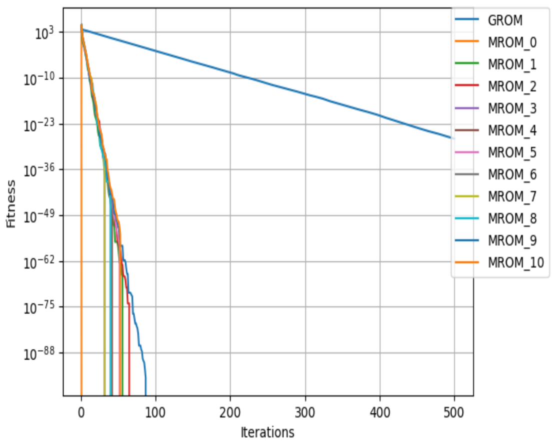
For the functions f_1 and f_2, the logarithmic curve showed that MROM variants had a sharp convergence to the global optima. This was demonstrated by the fact that proposed framework reached the global optima within the first

Table 3: Results on 23 Benchmark functions

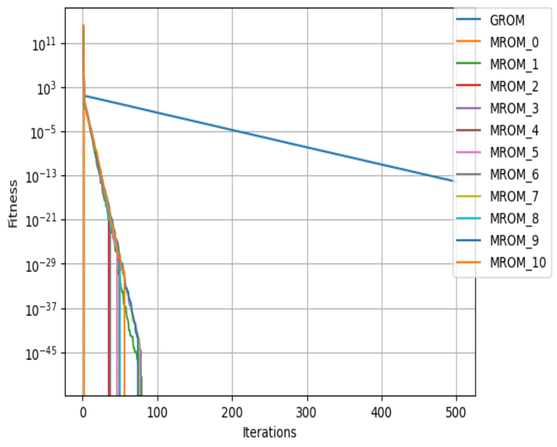
Optimizer Function	Metric	GROM	MROM_0	MROM_1	MROM_2	MROM_3	MROM_4	MROM_5	MROM_6	MROM_7	MROM_8	MROM_9	MROM_10
F1	MEAN	3.80E-03	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**
	BEST	4.26E-04	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**
	STD	3.48E-03	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	0.00E+00	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**
F2	MEAN	2.14E-02	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**
	BEST	6.19E-03	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**
	STD	9.87E-03	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**
F3	MEAN	4.26E-02	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**
	BEST	2.38E-03	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**
	STD	6.46E-02	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**
F4	MEAN	2.17E-02	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**
	BEST	3.58E-03	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	0.00E+00	**0.00E+00**	0.00E+00	**0.00E+00**
	STD	9.47E-03	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**
F5	MEAN	2.91E+01	**2.90E+01**	**2.90E+01**	**2.90E+01**	**2.90E+01**	**2.90E+01**	**2.90E+01**	**2.90E+01**	**2.90E+01**	**2.90E+01**	**2.90E+01**	**2.90E+01**
	BEST	2.90E+01	**2.90E+01**	**2.89E+01**	**2.89E+01**	**2.89E+01**	**2.89E+01**	**2.89E+01**	**2.89E+01**	**2.89E+01**	**2.89E+01**	**2.90E+01**	**2.89E+01**
	STD	7.22E-02	8.58E-03	2.22E-02	1.92E-02	1.57E-02	1.14E-02	2.17E-02	2.22E-02	1.87E-02	1.36E-02	**7.79E-03**	2.05E-02
F6	MEAN	7.35E+00	7.13E+00	6.80E+00	6.82E+00	6.75E+00	6.55E+00	**6.64E+00**	6.84E+00	6.82E+00	6.84E+00	6.89E+00	6.74E+00
	BEST	6.85E+00	5.17E+00	6.16E+00	4.23E+00	5.64E+00	5.06E+00	**4.85E+00**	6.10E+00	6.19E+00	5.53E+00	6.29E+00	5.71E+00
	STD	**2.36E-01**	4.86E-01	4.00E-01	7.77E-01	4.94E-01	7.22E-01	4.78E-01	4.01E-01	3.93E-01	4.65E-01	3.48E-01	4.45E-01
F7	MEAN	2.60E-02	7.25E-04	1.04E-03	5.00E-04	**4.91E-04**	6.94E-04	8.26E-04	8.40E-04	7.16E-04	8.59E-04	5.18E-04	8.15E-04
	BEST	1.19E-02	3.13E-05	4.61E-05	1.33E-04	7.91E-05	**4.16E-05**	4.13E-05	**4.16E-05**	6.92E-06	1.14E-04	8.97E-05	1.31E-04
	STD	6.53E-03	7.07E-04	6.18E-04	**2.15E-04**	4.50E-04	5.31E-04	9.43E-04	4.90E-04	8.24E-04	1.07E-03	2.71E-04	5.80E-04
F8	MEAN	-2.84E+03	-3.04E+03	-3.13E+03	-3.27E+03	-3.30E+03	-3.23E+03	**-3.49E+03**	-3.42E+03	-3.26E+03	-3.46E+03	-3.78E+03	-3.09E+03
	BEST	-3.55E+03	**-6.28E+03**	-4.41E+03	-4.15E+03	-4.17E+03	-4.25E+03	-4.54E+03	-4.29E+03	-3.89E+03	-5.17E+03	-5.87E+03	-4.00E+03
	STD	3.60E+02	8.07E+02	5.98E+02	4.13E+02	4.31E+02	4.83E+02	4.11E+02	4.93E+02	**3.56E+02**	5.87E+02	9.51E+02	4.11E+02
F9	MEAN	2.19E+02	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**
	BEST	8.17E-04	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**
	STD	7.74E+01	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**
F10	MEAN	4.73E-01	**4.44E-16**	**4.44E-16**	**4.44E-16**	**4.44E-16**	**4.44E-16**	**4.44E-16**	**4.44E-16**	**4.44E-16**	**4.44E-16**	**4.44E-16**	**4.44E-16**
	BEST	8.41E-03	**4.44E-16**	**4.44E-16**	**4.44E-16**	**4.44E-16**	**4.44E-16**	**4.44E-16**	**4.44E-16**	**4.44E-16**	**4.44E-16**	**4.44E-16**	**4.44E-16**
	STD	2.49E+00	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	0.00E+00	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**
F11	MEAN	4.51E-01	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**
	BEST	3.47E-03	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**
	STD	5.06E-01	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**
F12	MEAN	7.58E+00	1.33E+00	**1.09E+00**	1.15E+00	1.26E+00	1.18E+00	1.11E+00	1.10E+00	**1.09E+00**	1.17E+00	1.22E+00	1.22E+00
	BEST	1.34E+00	7.86E-01	6.99E-01	8.39E-01	8.46E-01	6.86E-01	6.74E-01	**5.94E-01**	8.78E-01	6.00E-01	6.33E-01	8.29E-01
	STD	7.08E+00	2.51E-01	2.12E-01	2.05E-01	2.24E-01	2.83E-01	2.99E-01	3.13E-01	**1.36E-01**	2.10E-01	2.62E-01	2.16E-01
F13	MEAN	1.86E+01	**3.00E+00**	**3.00E+00**	**3.00E+00**	**3.00E+00**	**3.00E+00**	**3.00E+00**	**3.00E+00**	**3.00E+00**	**3.00E+00**	**3.00E+00**	**3.00E+00**
	BEST	3.01E+00	3.00E+00	**2.99E+00**	**2.99E+00**	**2.99E+00**	**2.99E+00**	**2.99E+00**	**2.99E+00**	**2.99E+00**	**2.99E+00**	**2.99E+00**	**2.99E+00**
	STD	7.93E+00	1.36E-03	3.35E-03	2.11E-03	1.99E-03	**1.22E-03**	1.66E-03	2.72E-03	2.46E-03	1.78E-03	1.91E-03	2.03E-03
F14	MEAN	1.01E+01	7.13E+00	6.56E+00	**6.16E+00**	6.77E+00	5.48E+00	6.22E+00	7.24E+00	6.73E+00	4.59E+00	6.63E+00	4.75E+00
	BEST	1.98E+00	**9.98E-01**	**9.98E-01**	**9.98E-01**	**9.98E-01**	**9.98E-01**	**9.98E-01**	**9.98E-01**	**9.98E-01**	**9.98E-01**	**9.98E-01**	**9.98E-01**
	STD	5.43E+00	4.29E+00	3.50E+00	4.01E+00	4.38E+00	3.54E+00	**3.31E+00**	5.04E+00	4.37E+00	4.05E+00	4.91E+00	4.04E+00
F15	MEAN	4.12E-02	1.39E-02	6.77E-03	8.02E-03	9.48E-03	5.36E-03	**5.69E-03**	5.92E-03	7.36E-03	8.42E-03	8.48E-03	1.52E-02
	BEST	6.61E-04	3.56E-04	5.67E-04	**3.37E-04**	5.18E-04	3.22E-04	3.66E-04	4.00E-04	7.21E-04	4.68E-04	3.74E-04	3.73E-04
	STD	4.69E-02	2.06E-02	7.68E-03	1.25E-02	1.09E-02	6.88E-03	6.69E-03	1.36E-02	8.45E-03	1.04E-02	**1.03E-02**	1.77E-02
F16	MEAN	-6.86E-01	-1.00E+00	**-1.01E+00**	-9.28E-01	-9.99E-01	-1.02E+00	-1.02E+00	-1.00E+00	-9.75E-01	-9.94E-01	-1.00E+00	-9.84E-01
	BEST	**-1.03E+00**	**-1.03E+00**	**-1.03E+00**	**-1.03E+00**	**-1.03E+00**	**-1.03E+00**	**-1.03E+00**	**-1.03E+00**	**-1.03E+00**	**-1.03E+00**	**-1.03E+00**	**-1.03E+00**
	STD	4.02E-01	1.56E-01	7.23E-02	2.00E-01	5.70E-02	1.67E-02	**2.94E-02**	7.80E-02	1.00E-01	6.51E-02	7.23E-02	1.31E-01
F17	MEAN	9.93E-01	6.22E-01	4.20E-01	4.99E-01	5.09E-01	4.14E-01	4.29E-01	**4.13E-01**	7.89E-01	4.86E-01	5.05E-01	5.00E-01
	BEST	**3.98E-01**	**3.98E-01**	**3.98E-01**	**3.98E-01**	**3.98E-01**	**3.98E-01**	**3.98E-01**	**3.98E-01**	**3.98E-01**	**3.98E-01**	**3.98E-01**	**3.98E-01**
	STD	7.92E-01	2.89E-01	**1.17E-01**	2.40E-01	5.80E-01	2.45E-02	5.36E-02	3.13E-02	7.29E-01	2.41E-01	2.29E-01	3.26E-01
F18	MEAN	2.56E+01	**8.38E+00**	8.97E+00	8.68E+00	1.33E+01	1.65E+01	1.41E+01	1.14E+01	1.40E+01	8.30E+00	1.35E+01	1.55E+01
	BEST	**3.00E+00**	**3.00E+00**	**3.00E+00**	**3.00E+00**	**3.00E+00**	**3.00E+00**	**3.00E+00**	**3.00E+00**	**3.00E+00**	**3.00E+00**	3.36E+00	3.21E+00
	STD	3.16E+01	**1.00E+01**	1.15E+01	1.06E+01	1.17E+01	1.22E+01	2.38E+01	1.21E+01	1.18E+01	7.34E+00	1.11E+01	1.20E+01
F19	MEAN	-3.71E+00	-3.77E+00	**-3.79E+00**	-3.71E+00	-3.75E+00	-3.68E+00	-3.61E+00	-3.51E+00	-3.67E+00	-3.45E+00	-3.55E+00	-3.60E+00
	BEST	**-3.86E+00**	**-3.86E+00**	**-3.86E+00**	**-3.86E+00**	**-3.86E+00**	**-3.86E+00**	**-3.86E+00**	**-3.83E+00**	**-3.86E+00**	**-3.86E+00**	**-3.86E+00**	**-3.86E+00**
	STD	2.14E-01	1.56E-01	8.49E-02	1.39E-01	**8.34E-02**	1.16E-01	2.38E-01	3.22E-01	2.60E-01	4.33E-01	2.57E-01	3.22E-01
F20	MEAN	-2.16E+00	**-2.34E+00**	-2.27E+00	-2.26E+00	-2.30E+00	-2.28E+00	-2.14E+00	-2.23E+00	-2.28E+00	-2.17E+00	-2.07E+00	-1.87E+00
	BEST	-2.97E+00	-3.07E+00	-3.14E+00	-3.02E+00	**-3.27E+00**	-3.09E+00	-3.08E+00	-3.23E+00	-3.01E+00	-2.98E+00	-3.03E+00	-3.05E+00
	STD	**4.42E-01**	5.11E-01	4.90E-01	5.09E-01	4.68E-01	5.61E-01	5.14E-01	6.05E-01	4.72E-01	5.16E-01	4.87E-01	5.07E-01
F21	MEAN	-1.98E+00	-3.52E+00	-4.14E+00	**-5.34E+00**	-4.25E+00	-4.92E+00	-4.54E+00	-3.96E+00	-4.99E+00	-4.99E+00	-4.77E+00	-4.71E+00
	BEST	-9.79E+00	-8.27E+00	-9.23E+00	-1.01E+01	-8.60E+00	-9.05E+00	-7.05E+00	-7.50E+00	-9.73E+00	-9.13E+00	-9.00E+00	**-9.98E+00**
	STD	1.59E+00	1.58E+00	2.06E+00	2.35E+00	**1.01E+00**	1.76E+00	1.34E+00	1.35E+00	1.86E+00	1.78E+00	1.58E+00	1.64E+00
F22	MEAN	-2.51E+00	-2.93E+00	-4.93E+00	**-5.98E+00**	-5.07E+00	-4.36E+00	-4.20E+00	-4.05E+00	-4.38E+00	-4.87E+00	-4.68E+00	-5.03E+00
	BEST	-7.80E+00	-7.08E+00	-1.01E+01	-9.66E+00	-1.02E+01	-7.67E+00	-6.39E+00	-6.86E+00	**-1.03E+01**	**-1.03E+01**	-1.00E+01	-9.71E+00
	STD	1.69E+00	1.62E+00	2.33E+00	2.51E+00	1.95E+00	1.26E+00	**1.25E+00**	1.35E+00	1.90E+00	2.00E+00	1.52E+00	1.90E+00
F23	MEAN	-3.35E+00	-4.00E+00	-3.61E+00	**-5.30E+00**	-4.49E+00	-4.11E+00	-5.11E+00	-4.84E+00	-4.44E+00	-4.54E+00	-5.28E+00	-4.79E+00
	BEST	-8.61E+00	-8.72E+00	-8.32E+00	**-1.02E+01**	-7.70E+00	-9.67E+00	-8.24E+00	-8.67E+00	-7.03E+00	-1.01E+01	-9.49E+00	-8.31E+00
	STD	2.25E+00	2.31E+00	2.12E+00	1.99E+00	1.37E+00	1.69E+00	1.83E+00	1.99E+00	**9.17E-01**	1.69E+00	2.34E+00	1.74E+00

Table 4: Results of CEC 2014 benchmark functions

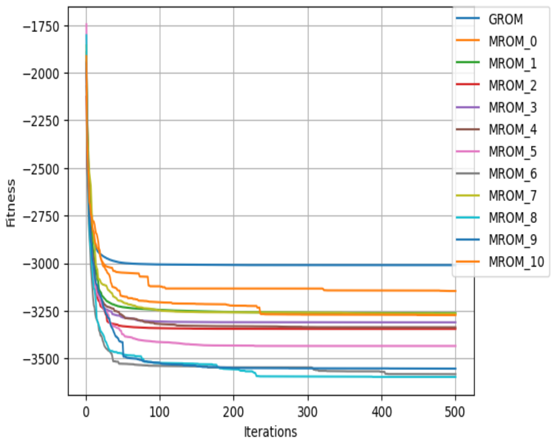
Optimizer Function	Metric	GROM	MROM_0	MROM_1	MROM_2	MROM_3	MROM_4	MROM_5	MROM_6	MROM_7	MROM_8	MROM_9	MROM_10
F101	MEAN	6.23E+10	5.74E+10	5.44E+10	5.63E+10	**5.21E+10**	6.04E+10	5.89E+10	5.87E+10	5.90E+10	6.05E+10	5.90E+10	5.97E+10
	BEST	5.04E+10	4.61E+10	4.10E+10	4.49E+10	**3.80E+10**	4.32E+10	4.45E+10	4.76E+10	4.42E+10	5.01E+10	5.01E+10	4.40E+10
	STD	6.16E+09	6.29E+09	7.10E+09	6.03E+09	5.71E+09	7.64E+09	4.73E+09	**3.90E+09**	6.55E+09	5.56E+09	5.12E+09	5.20E+09
F102	MEAN	1.54E+05	1.08E+05	**6.41E+04**	7.00E+04	7.88E+04	7.81E+04	9.89E+04	9.30E+04	1.11E+05	1.02E+05	1.02E+05	1.15E+05
	BEST	9.71E+04	6.63E+04	**3.75E+04**	4.35E+04	6.14E+04	5.35E+04	4.67E+04	4.72E+04	7.14E+04	5.80E+04	7.22E+04	6.23E+04
	STD	3.11E+04	3.62E+04	1.80E+04	**1.27E+04**	1.61E+04	1.70E+04	2.87E+04	2.35E+04	3.52E+04	3.42E+04	1.73E+04	3.07E+04
F103	MEAN	3.46E+02	3.45E+02	3.43E+02	**3.39E+02**	3.43E+02	3.44E+02	3.43E+02	3.43E+02	3.43E+02	3.44E+02	3.44E+02	3.43E+02
	BEST	3.41E+02	3.39E+02	3.37E+02	**3.34E+02**	3.40E+02	3.41E+02	3.38E+02	3.41E+02	3.37E+02	3.38E+02	3.41E+02	3.38E+02
	STD	2.18E+00	3.07E+00	2.73E+00	2.36E+00	1.83E+00	**1.74E+00**	1.85E+00	1.85E+00	2.36E+00	2.82E+00	1.91E+00	2.17E+00
F104	MEAN	9.07E+03	9.08E+03	8.29E+03	7.99E+03	**7.94E+03**	8.19E+03	8.04E+03	8.03E+03	8.17E+03	8.20E+03	8.04E+03	8.14E+03
	BEST	7.75E+03	8.39E+03	7.79E+03	7.32E+03	7.17E+03	7.27E+03	7.50E+03	7.16E+03	**6.89E+03**	7.48E+03	**6.98E+03**	7.09E+03
	STD	5.40E+02	5.11E+02	3.04E+02	**3.12E+02**	3.81E+02	3.50E+02	3.48E+02	3.66E+02	4.92E+02	4.21E+02	4.41E+02	4.07E+02
F105	MEAN	5.04E+02	5.05E+02	5.03E+02	**5.02E+02**	5.03E+02	5.03E+02	5.03E+02	5.03E+02	5.03E+02	5.03E+02	5.03E+02	5.03E+02
	BEST	5.03E+02	5.03E+02	**5.02E+02**	**5.02E+02**	**5.02E+02**	**5.02E+02**	**5.02E+02**	**5.02E+02**	**5.02E+02**	**5.02E+02**	**5.02E+02**	**5.02E+02**
	STD	8.07E-01	7.36E-01	5.12E-01	**3.29E-01**	5.35E-01	6.72E-01	6.18E-01	4.29E-01	5.17E-01	5.88E-01	8.75E-01	7.07E-01
F106	MEAN	6.06E+02	6.06E+02	6.05E+02	**6.05E+02**	**6.05E+02**	6.06E+02	6.06E+02	6.06E+02	6.06E+02	6.06E+02	6.06E+02	6.06E+02
	BEST	**6.05E+02**	**6.05E+02**	**6.05E+02**	**6.05E+02**	**6.05E+02**	**6.05E+02**	**6.05E+02**	**6.05E+02**	**6.05E+02**	**6.05E+02**	**6.05E+02**	**6.05E+02**
	STD	2.60E-01	2.17E-01	3.41E-01	3.68E-01	3.16E-01	2.94E-01	**2.06E-01**	3.36E-01	3.47E-01	2.17E-01	2.16E-01	2.78E-01
F107	MEAN	8.06E+02	8.11E+02	8.01E+02	**7.91E+02**	8.04E+02	8.08E+02	8.05E+02	8.07E+02	8.10E+02	8.09E+02	8.11E+02	8.11E+02
	BEST	7.81E+02	7.86E+02	7.74E+02	**7.70E+02**	7.82E+02	7.92E+02	7.86E+02	7.90E+02	8.03E+02	7.77E+02	7.85E+02	7.95E+02
	STD	1.11E+01	1.08E+01	1.12E+01	1.25E+01	1.11E+01	6.79E+00	1.10E+01	**1.03E+01**	6.12E+00	8.66E+00	9.34E+00	8.21E+00
F108	MEAN	1.43E+07	1.10E+07	1.05E+07	8.73E+06	**8.45E+06**	9.33E+06	9.00E+06	1.06E+07	1.11E+07	1.06E+07	1.22E+07	1.25E+07
	BEST	4.17E+06	3.03E+06	1.86E+06	1.11E+06	**9.25E+05**	2.93E+06	2.75E+06	5.01E+06	2.52E+06	3.34E+06	2.25E+06	2.16E+06
	STD	5.57E+06	9.79E+06	6.60E+06	**4.79E+06**	5.88E+06	5.12E+06	5.79E+06	5.03E+06	5.90E+06	5.38E+06	5.76E+06	7.26E+06
F109	MEAN	9.14E+02	9.14E+02	9.13E+02	**9.04E+02**	9.13E+02	9.13E+02	9.13E+02	9.13E+02	9.13E+02	9.14E+02	9.14E+02	9.13E+02
	BEST	9.14E+02	9.13E+02	9.13E+02	**9.04E+02**	9.13E+02	9.13E+02	9.13E+02	9.13E+02	9.13E+02	9.13E+02	9.13E+02	9.13E+02
	STD	1.29E-01	3.55E-01	1.82E-01	1.58E-01	2.24E-01	1.37E-01	1.78E-01	1.51E-01	2.41E-01	**1.27E-01**	1.55E-01	1.83E-01
F110	MEAN	2.06E+08	1.60E+08	**1.01E+08**	1.13E+08	1.12E+08	1.09E+08	1.20E+08	1.87E+08	1.31E+08	1.64E+08	1.55E+08	1.48E+08
	BEST	4.42E+07	5.11E+07	3.57E+07	**2.57E+07**	3.03E+07	3.23E+07	5.89E+07	3.70E+07	7.68E+07	3.11E+07	7.70E+07	5.34E+07
	STD	1.77E+08	8.67E+07	5.02E+07	4.38E+07	6.19E+07	5.58E+07	**4.31E+07**	8.35E+07	5.43E+07	8.07E+07	4.99E+07	6.38E+07
F111	MEAN	1.26E+09	2.01E+09	**7.89E+08**	8.36E+08	1.04E+09	1.72E+09	1.43E+09	1.41E+09	1.32E+09	1.71E+09	1.80E+09	1.56E+09
	BEST	5.33E+07	1.59E+08	5.54E+07	**4.81E+07**	6.25E+07	2.66E+08	1.90E+08	4.22E+08	9.78E+07	1.38E+08	1.68E+08	2.27E+08
	STD	7.66E+08	1.95E+09	8.15E+08	**6.53E+08**	8.30E+08	1.06E+09	1.04E+09	6.29E+08	9.77E+08	1.10E+09	9.51E+08	8.52E+08
F112	MEAN	5.59E+15	2.23E+15	**2.77E+14**	5.82E+14	1.74E+15	3.71E+15	1.09E+15	1.45E+15	1.07E+15	1.94E+15	2.06E+15	1.59E+15
	BEST	4.92E+12	3.34E+12	7.70E+11	2.00E+11	**1.13E+11**	1.39E+13	6.41E+12	3.38E+11	6.15E+12	3.50E+12	1.86E+12	2.06E+13
	STD	4.96E+15	5.83E+15	4.66E+14	**1.10E+15**	4.57E+15	4.13E+15	1.20E+15	2.27E+15	1.90E+15	2.74E+15	1.81E+15	2.22E+15
F113	MEAN	3.02E+03	2.35E+03	2.08E+03	**1.91E+03**	1.98E+03	1.93E+03	1.92E+03	1.89E+03	1.94E+03	1.90E+03	1.94E+03	1.95E+03
	BEST	2.14E+03	1.98E+03	1.80E+03	1.74E+03	1.69E+03	**1.68E+03**	1.78E+03	1.73E+03	1.76E+03	1.78E+03	1.73E+03	1.79E+03
	STD	6.11E+02	3.17E+02	1.69E+02	**9.66E+01**	1.62E+02	1.26E+02	8.68E+01	1.06E+02	1.12E+02	1.00E+02	1.05E+02	8.56E+01
F114	MEAN	**5.33E+03**	6.55E+03	6.93E+03	8.02E+03	9.50E+03	1.04E+04	1.04E+04	1.05E+04	9.86E+03	1.07E+04	1.06E+04	1.14E+04
	BEST	**3.18E+03**	2.40E+03	4.00E+03	4.61E+03	4.82E+03	5.56E+03	5.82E+03	6.50E+03	5.98E+03	6.36E+03	6.35E+03	5.99E+03
	STD	**1.21E+03**	2.59E+03	1.99E+03	2.67E+03	2.57E+03	2.20E+03	2.41E+03	2.64E+03	2.56E+03	1.89E+03	2.18E+03	2.49E+03
F115	MEAN	**3.00E+03**	3.21E+03	3.19E+03	3.18E+03	3.20E+03	3.31E+03	3.34E+03	3.24E+03	3.40E+03	3.44E+03	3.30E+03	3.49E+03
	BEST	2.95E+03	2.90E+03	3.00E+03	2.63E+03	2.86E+03	3.02E+03	2.97E+03	2.63E+03	3.00E+03	2.78E+03	2.66E+03	**2.59E+03**
	STD	**3.68E+01**	1.67E+02	2.17E+02	1.96E+02	1.86E+02	2.46E+02	2.11E+02	2.45E+02	2.75E+02	4.24E+02	2.79E+02	3.96E+02
F116	MEAN	1.61E+03	1.61E+03	1.61E+03	**1.60E+03**	1.61E+03	1.61E+03	1.61E+03	1.61E+03	1.61E+03	1.61E+03	1.61E+03	1.61E+03
	BEST	1.61E+03	1.61E+03	1.61E+03	**1.60E+03**	1.61E+03	1.61E+03	1.61E+03	1.61E+03	1.61E+03	1.61E+03	1.61E+03	1.61E+03
	STD	2.34E-01	2.31E-01	2.67E-01	**2.55E-01**	2.31E-01	2.03E-01	2.55E-01	2.25E-01	2.07E-01	2.29E-01	2.68E-01	3.32E-01
F117	MEAN	1.07E+08	8.00E+07	**5.05E+07**	6.29E+07	6.33E+07	6.62E+07	7.80E+07	8.38E+07	9.65E+07	9.86E+07	7.09E+07	1.06E+08
	BEST	1.76E+07	1.53E+07	1.79E+07	**7.29E+06**	9.80E+06	1.74E+07	4.49E+06	2.12E+07	2.82E+07	3.28E+07	2.81E+07	3.08E+07
	STD	1.15E+08	5.43E+07	**2.89E+07**	4.42E+07	3.60E+07	3.24E+07	6.47E+07	4.64E+07	5.47E+07	3.90E+07	3.22E+07	5.05E+07
F118	MEAN	8.42E+09	6.33E+09	4.87E+09	**4.80E+09**	5.18E+09	6.66E+09	6.38E+09	5.74E+09	5.89E+09	6.34E+09	5.85E+09	6.84E+09
	BEST	4.46E+09	**1.78E+09**	8.62E+08	2.36E+09	1.11E+09	1.70E+09	2.28E+09	2.55E+09	2.42E+09	2.80E+09	1.85E+09	2.41E+09
	STD	2.17E+09	2.74E+09	2.03E+09	**1.81E+09**	2.06E+09	2.62E+09	2.58E+09	1.99E+09	1.98E+09	1.75E+09	1.97E+09	2.51E+09
F119	MEAN	5.70E+08	1.57E+09	7.25E+08	**5.20E+08**	7.27E+08	7.01E+08	5.74E+08	1.53E+09	8.77E+08	1.41E+09	1.23E+09	1.63E+09
	BEST	2.01E+08	**1.54E+07**	2.87E+07	2.52E+07	3.44E+07	2.09E+08	9.13E+07	2.85E+08	3.77E+08	7.08E+07	5.08E+08	1.09E+08
	STD	2.57E+08	1.69E+09	5.36E+08	3.89E+08	5.26E+08	6.42E+08	**3.63E+08**	1.08E+09	5.23E+08	9.81E+08	6.24E+08	1.12E+09
F120	MEAN	1.34E+15	5.10E+14	3.71E+14	6.03E+14	**4.22E+14**	7.25E+14	5.90E+14	6.32E+14	8.05E+14	5.56E+14	7.33E+14	1.01E+15
	BEST	**1.17E+12**	6.11E+12	9.95E+12	2.29E+12	4.41E+12	6.43E+11	1.84E+12	1.74E+13	4.47E+13	1.80E+13	1.35E+13	3.17E+13
	STD	1.46E+15	6.42E+14	4.95E+14	8.88E+14	7.00E+14	8.21E+14	**4.90E+14**	5.91E+14	5.24E+14	5.57E+14	6.22E+14	8.58E+14
F121	MEAN	1.73E+09	1.20E+09	**8.38E+08**	9.42E+08	9.53E+08	1.28E+09	1.80E+09	2.20E+09	1.85E+09	2.43E+09	1.52E+09	2.03E+09
	BEST	5.15E+08	3.24E+08	**6.86E+07**	9.80E+07	2.63E+08	4.79E+08	4.32E+08	4.79E+08	1.05E+08	5.46E+08	8.03E+08	6.72E+08
	STD	1.04E+09	1.13E+09	7.83E+08	6.81E+08	**4.98E+08**	6.97E+08	1.47E+09	1.34E+09	9.34E+08	1.59E+09	9.58E+08	1.42E+09
F122	MEAN	1.78E+14	7.38E+13	**1.30E+13**	1.54E+13	3.57E+13	3.24E+13	2.71E+13	4.73E+13	6.23E+13	4.67E+13	3.05E+13	2.60E+13
	BEST	4.37E+11	7.12E+10	**7.48E+07**	2.93E+08	2.12E+10	8.06E+11	1.82E+11	4.72E+10	6.96E+09	5.35E+10	5.36E+11	1.23E+11
	STD	2.12E+14	1.82E+14	**1.67E+13**	3.38E+13	5.83E+13	3.72E+13	3.08E+13	4.61E+13	6.17E+13	4.50E+13	3.74E+13	3.30E+13
F123	MEAN	3.19E+03	**2.56E+03**	2.65E+03	2.67E+03	2.65E+03	2.66E+03	2.73E+03	2.59E+03	2.72E+03	2.61E+03	2.70E+03	2.67E+03
	BEST	**2.50E+03**	**2.50E+03**	**2.50E+03**	**2.50E+03**	**2.50E+03**	**2.50E+03**	**2.50E+03**	2.51E+03	**2.50E+03**	2.51E+03	2.51E+03	**2.50E+03**
	STD	7.10E+02	1.55E+02	**1.21E+02**	1.90E+02	1.75E+02	1.76E+02	2.29E+02	9.77E+01	2.05E+02	1.29E+02	2.04E+02	1.81E+02
F124	MEAN	**2.60E+03**	**2.60E+03**	**2.60E+03**	**2.60E+03**	**2.60E+03**	**2.60E+03**	**2.60E+03**	**2.60E+03**	**2.60E+03**	**2.60E+03**	**2.60E+03**	**2.60E+03**
	BEST	**2.60E+03**	**2.60E+03**	**2.60E+03**	**2.60E+03**	**2.60E+03**	**2.60E+03**	**2.60E+03**	**2.60E+03**	**2.60E+03**	**2.60E+03**	**2.60E+03**	**2.60E+03**
	STD	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**
F125	MEAN	**2.70E+03**	**2.70E+03**	**2.70E+03**	**2.70E+03**	**2.70E+03**	**2.70E+03**	**2.70E+03**	**2.70E+03**	**2.70E+03**	**2.70E+03**	**2.70E+03**	**2.70E+03**
	BEST	**2.70E+03**	**2.70E+03**	**2.70E+03**	**2.70E+03**	**2.70E+03**	**2.70E+03**	**2.70E+03**	**2.70E+03**	**2.70E+03**	**2.70E+03**	**2.70E+03**	**2.70E+03**
	STD	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**
F126	MEAN	**2.80E+03**	**2.80E+03**	**2.80E+03**	**2.80E+03**	**2.80E+03**	**2.80E+03**	**2.80E+03**	**2.80E+03**	**2.80E+03**	**2.80E+03**	**2.80E+03**	**2.80E+03**
	BEST	**2.80E+03**	**2.80E+03**	**2.80E+03**	**2.80E+03**	**2.80E+03**	**2.80E+03**	**2.80E+03**	**2.80E+03**	**2.80E+03**	**2.80E+03**	**2.80E+03**	**2.80E+03**
	STD	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**
F127	MEAN	**2.90E+03**	**2.90E+03**	**2.90E+03**	**2.90E+03**	**2.90E+03**	**2.90E+03**	**2.90E+03**	**2.90E+03**	**2.90E+03**	**2.90E+03**	**2.90E+03**	**2.90E+03**
	BEST	**2.90E+03**	**2.90E+03**	**2.90E+03**	**2.90E+03**	**2.90E+03**	**2.90E+03**	**2.90E+03**	**2.90E+03**	**2.90E+03**	**2.90E+03**	**2.90E+03**	**2.90E+03**
	STD	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**
F128	MEAN	**3.00E+03**	**3.00E+03**	**3.00E+03**	**3.00E+03**	**3.00E+03**	**3.00E+03**	**3.00E+03**	**3.00E+03**	**3.00E+03**	**3.00E+03**	**3.00E+03**	**3.00E+03**
	BEST	**3.00E+03**	**3.00E+03**	**3.00E+03**	**3.00E+03**	**3.00E+03**	**3.00E+03**	**3.00E+03**	**3.00E+03**	**3.00E+03**	**3.00E+03**	**3.00E+03**	**3.00E+03**
	STD	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**
F129	MEAN	1.07E+10	9.07E+09	**5.30E+09**	6.91E+09	7.39E+09	7.92E+09	6.23E+09	8.90E+09	7.44E+09	7.18E+09	1.02E+10	9.69E+09
	BEST	2.59E+09	1.68E+09	1.93E+09	1.75E+09	2.85E+09	2.92E+09	2.54E+09	3.97E+09	**1.30E+09**	3.39E+09	4.18E+09	3.56E+09
	STD	5.62E+09	4.29E+09	2.40E+09	4.18E+09	3.31E+09	3.41E+09	**3.27E+09**	3.48E+09	4.27E+09	2.84E+09	4.32E+09	3.74E+09
F130	MEAN	5.60E+12	3.26E+12	4.06E+11	**2.85E+11**	4.97E+11	7.92E+11	9.43E+11	1.10E+12	4.97E+11	8.02E+11	7.96E+11	8.89E+11
	BEST	1.49E+11	1.71E+10	1.71E+09	2.08E+09	3.52E+09	1.33E+10	8.22E+09	2.93E+09	5.41E+09	2.74E+09	**1.28E+09**	2.27E+10
	STD	7.90E+12	4.88E+12	**4.80E+11**	5.24E+11	6.85E+11	1.47E+12	1.63E+12	1.66E+12	4.82E+11	9.26E+11	1.01E+12	8.45E+11



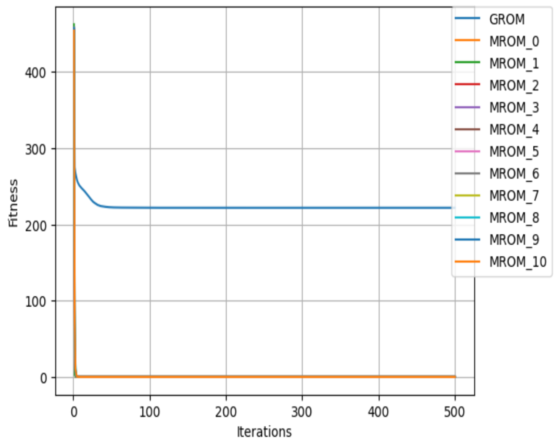
(a) F1



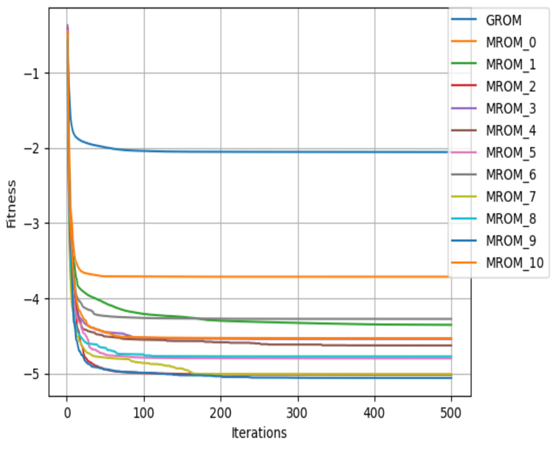
(b) F2



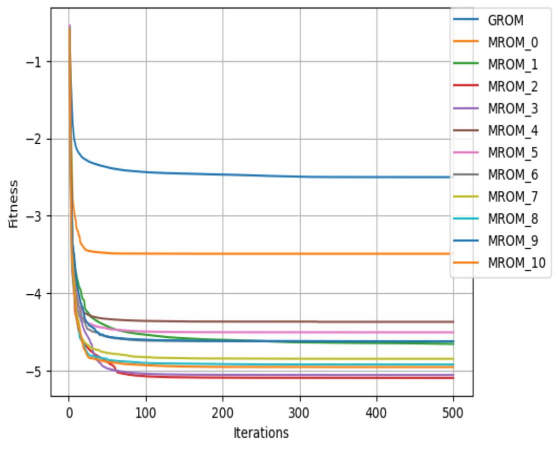
(c) F8



(d) F9



(e) F21



(f) F22

Fig. 10: Selective Convergence plot for F23 benchmark suit

100 iterations. The function f_8 was challenging for both MROM variants and GROM, as both algorithms found themselves trapped in the local optima. However, convergence curve shows superior performance of proposed framework algorithms. From the convergence curve of function f_9, f_{21}, and f_{22}, it is evident that there is significant amount of difference between convergence of MROM framework algorithms and GROM.

6.2. Comparison on CEC-2014 benchmark functions

The CEC-2014 problems include 30 minimization functions that are grouped into four categories: unimodal functions, multi-modal functions, Hybrid functions, and composite functions. The details of these benchmark functions can be found in Table A.10.

To evaluate the performance of the algorithms, the results from 30 runs of each benchmark function were averaged, and the mean fitness, standard deviation, and best fitness are presented. The population size for both MROM-based algorithms and GROM was set to 50 and the algorithms were run for a maximum of 300 iterations, which serves as the stopping criterion.

6.2.1. Results Discussion

Category A: Unimodal functions
The first three functions in CEC 2014 benchmark function set are unimodal functions. The results presented in Table 4 depict that the MROM variants outperform the GROM for all unimodal functions. As these functions have only one global optimum, making them ideal for testing the exploitative nature of optimization algorithms and results demonstrate the superiority of MROM algorithm's exploitation capability as compare to GROM.

Category B: Multi-modal Functions The CEC 2014 benchmark suite contains a large number of multi-modal benchmark functions ($f_{104} - f_{116}$), which are characterized by the presence of a large number of local optima. These functions are particularly challenging to optimize as they require an algorithm to not only find the global optimum but also to avoid becoming trapped in local optima. In addition, these functions are also rotated and shifted, further complicating the search for global optima. The results of the MROM algorithms and GROM on these multi-modal functions provide important insight into their performance in this type of search space.

In functions $f_{104-111}$, both the proposed framework algorithms and GROM perform competitively, with MROM variants having a slight advantage over GROM. This is likely due to the fact that these functions have a relatively moderate number of local optima and the search space is not overly complex. In this set of functions, the proposed framework variants outperforms GROM in all cases and MROM_2 (silver ratio) is the best performer among all competitors. This shows the suitability of silver ratio over golden ratio in solving optimization problems.

For the functions $f_{112-113}$, the results are quite different, with MROM-based algorithms significantly outperforming GROM. These functions have a larger number of local optima and a more complex search space, making them particularly challenging to optimize. In these functions, the novel search mechanism of MROM framework plays an important role as it helps in avoiding entrapment in local solution by moving away from worst solutions. In contrast, the GROM algorithm encounter difficulties in avoiding local optima and exploring the search space effectively in this group of functions. In the case of functions $f_{114-115}$, GROM outperforms the proposed MROM variants whereas, in f_{116}, MROM_2 outperforms GROM and other competitors.

Overall, the findings of the multi-modal functions make it clear that MROM framework performs better than GROM. The MROM algorithm's ability to strike a balance between exploration and exploitation with enhanced search mechanism helps in avoiding local optima, making it an effective tool for solving optimization problems that involve multimodal functions.

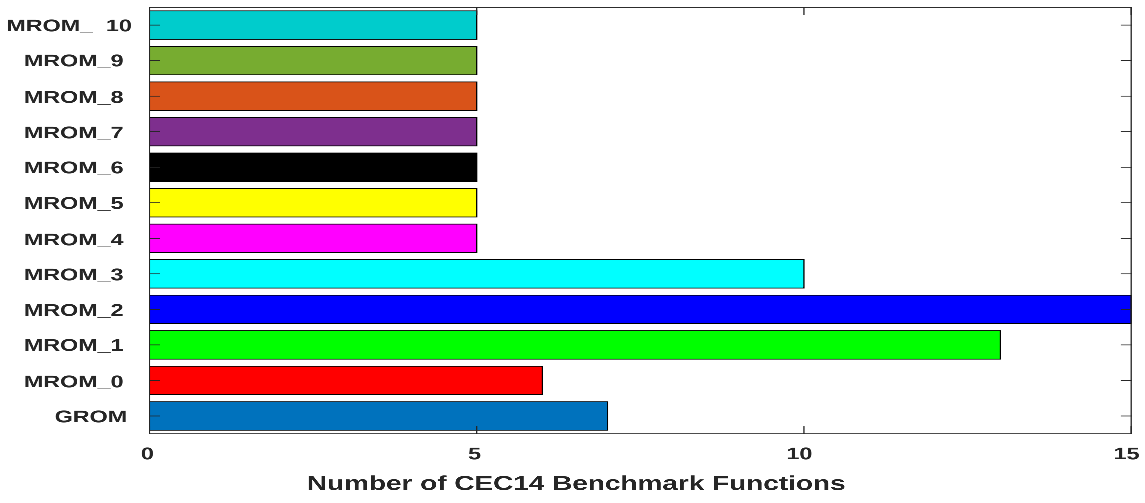
Category C: Hybrid Functions
The CEC 2014 benchmark suite also includes a set of hybrid functions, f_{117} to f_{128} that are characterized by the separation of variables into several sub-components (Liang et al. (2013)), with various basic functions then being employed for each sub-component. This creates a difficult search space for optimization methods, as the variables are not independent and the basic functions are often nonlinear. The results of the MROM algorithms and GROM on these hybrid functions provide important insight into their performance in this type of search space.

The MROM variants outperform GROM for functions f_{117} to f_{123}, likely because these functions have a moderate number of local optima and a not overly complex search space. Additionally, MROM_1 and MROM_2 perform better than other metal ratios for these functions.

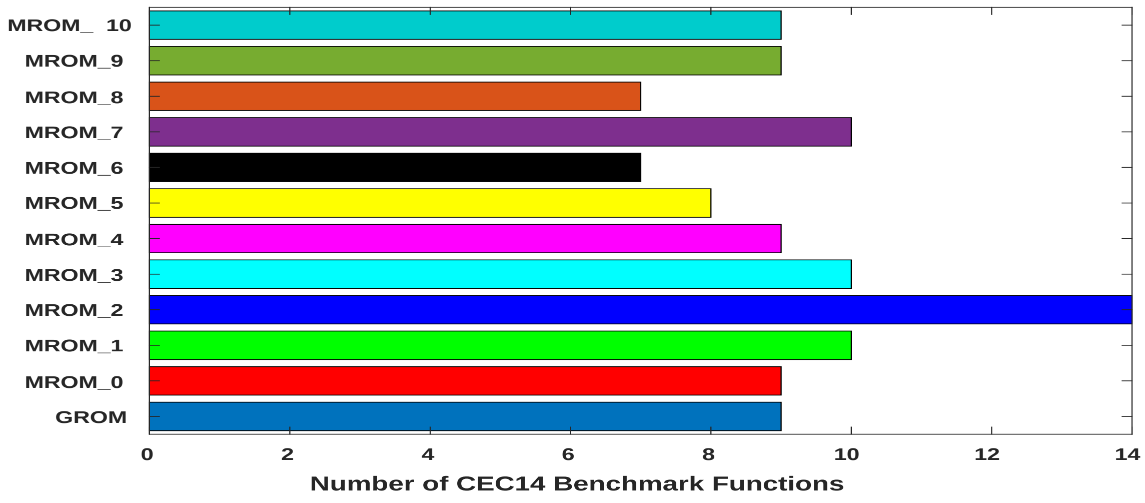
However, for functions f_{124} to f_{128}, both MROM variants and GROM algorithms get trapped in local minima. This is because these functions have a large number of local optima and a highly complex search space, making optimization particularly challenging. In these cases, MROM based algorithms and GROM may not effectively navigate the search space and find the global optimum. However, this does not imply that the algorithms are generally ineffective; it simply implies that they are not appropriate for these specific functions.

The results of the MROM-based algorithms and GROM on the hybrid functions of the CEC 2014 benchmark suite demonstrate that MROM is a highly effective method for global optimization in this type of search space. However, it is important to note that the MROM and GROM algorithms may not be suitable for certain highly complex and challenging hybrid functions where a large number of local optima are present.

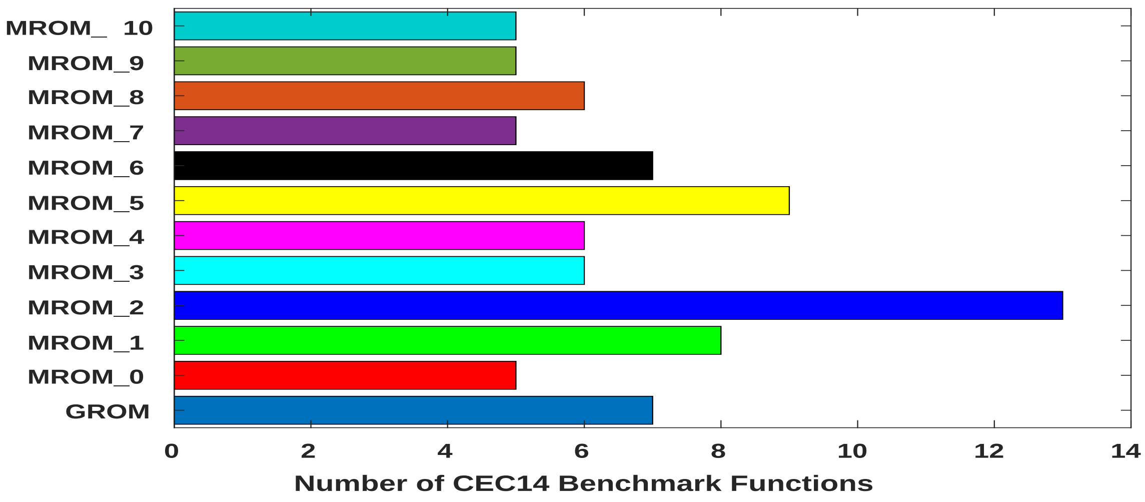
Category D: Composition Functions
The function f_{129} & f_{130} are Composition Functions. The



(a) Performance in Mean Objective Function Value Achieved by Various Metal ratios on CEC14 Benchmark Functions



(b) Performance in Best Objective Function Value Achieved by Various Metal ratios on CEC14 Benchmark Functions



(c) Performance in Standard Deviation in Mean Objective Function Value Achieved by Various Metal ratios on CEC14 Benchmark Functions

Fig. 11: Comparison of various metal ratios on CEC14 functions

Table 5: Results of FNN's Trained by different nature inspired optimization algorithms on First 15 benchmark data-sets

Data-Sets		MROM	GROM	BAT	CS	DE	FFA	GA	HHO	JAYA	MFO	PSO	SCA	SSA	WOA
Aggregation	MEAN	9.20E-01	9.20E-01	9.20E-01	9.40E-01	9.20E-01	9.10E-01	**9.77E-01**	9.20E-01	9.20E-01	9.20E-01	9.20E-01	9.20E-01	9.20E-01	9.20E-01
	STD	**3.53E-18**	**3.53E-18**	**3.53E-18**	1.10E-02	**3.53E-18**	3.05E-02	2.82E-17	**3.53E-18**	**3.53E-18**	**3.53E-18**	**3.53E-18**	**3.53E-18**	**3.53E-18**	**3.53E-18**
	BEST	9.20E-01	9.20E-01	9.20E-01	9.50E-01	9.20E-01	9.39E-01	**9.77E-01**	9.20E-01	9.20E-01	9.20E-01	9.20E-01	9.20E-01	9.20E-01	9.20E-01
Aniso	MEAN	**7.07E-01**	7.04E-01	6.74E-01	7.05E-01	7.06E-01	**7.07E-01**	3.53E-01	**7.07E-01**	7.05E-01	**7.07E-01**	**7.07E-01**	7.05E-01	**7.07E-01**	**7.07E-01**
	STD	**1.13E-16**	3.71E-03	8.50E-02	3.23E-03	2.03E-03	**1.13E-16**	5.65E-17	**1.13E-16**	2.53E-03	**1.13E-16**	**1.13E-16**	2.71E-03	**1.13E-16**	**1.13E-16**
	BEST	**7.07E-01**	**7.07E-01**	**7.07E-01**	**7.07E-01**	**7.07E-01**	**7.07E-01**	3.53E-01	**7.07E-01**	**7.07E-01**	**7.07E-01**	**7.07E-01**	**7.07E-01**	**7.07E-01**	**7.07E-01**
Appenditics	MEAN	**8.21E-01**	8.18E-01	8.12E-01	8.03E-01	7.99E-01	8.03E-01	7.27E-01	8.06E-01	7.84E-01	5.19E-01	7.84E-01	7.99E-01	8.18E-01	**8.21E-01**
	STD	**1.60E-02**	3.39E-16	2.30E-02	4.80E-02	5.00E-02	3.40E-02	2.26E-16	3.10E-02	6.50E-02	1.20E-02	5.50E-02	5.50E-02	3.39E-16	3.70E-02
	BEST	**9.09E-01**	8.18E-01	8.18E-01	**9.09E-01**	**9.09E-01**	8.18E-01	7.27E-01	8.18E-01	**9.09E-01**	5.39E-01	**9.09E-01**	**9.09E-01**	8.18E-01	**9.09E-01**
Balance	MEAN	8.62E-01	8.41E-01	8.97E-01	8.89E-01	8.77E-01	**9.15E-01**	8.60E-01	8.60E-01	8.77E-01	8.34E-01	**9.15E-01**	8.34E-01	9.06E-01	8.33E-01
	STD	5.80E-02	4.80E-02	2.40E-02	2.40E-02	4.10E-02	1.70E-02	5.65E-17	4.50E-02	3.70E-02	6.30E-02	2.00E-02	6.00E-02	1.90E-02	5.00E-02
	BEST	9.23E-01	9.07E-01	9.23E-01	9.23E-01	9.23E-01	**9.39E-01**	8.60E-01	**9.39E-01**	9.23E-01	9.23E-01	9.23E-01	9.23E-01	9.23E-01	9.23E-01
Banknote	MEAN	**9.92E-01**	9.20E-01	9.83E-01	9.61E-01	9.44E-01	9.90E-01	5.57E-01	9.79E-01	9.52E-01	9.88E-01	9.84E-01	9.43E-01	9.11E-01	9.64E-01
	STD	7.00E-03	4.70E-02	1.90E-02	2.20E-02	3.20E-02	9.00E-03	0.00E+00	2.60E-02	2.80E-02	6.00E-03	7.00E-03	3.30E-02	5.80E-02	2.80E-02
	BEST	**1.00E+00**	9.71E-01	**1.00E+00**	**1.00E+00**	**1.00E+00**	**1.00E+00**	5.57E-01	**1.00E+00**	9.92E-01	**1.00E+00**	9.92E-01	9.92E-01	9.92E-01	9.92E-01
Blobs	MEAN	6.52E-01	6.53E-01	6.53E-01	6.53E-01	6.53E-01	**8.33E-01**	3.20E-01	6.53E-01	6.52E-01	6.53E-01	6.53E-01	6.53E-01	6.53E-01	6.52E-01
	STD	2.00E-03	2.26E-16	1.00E-03	2.26E-16	2.26E-16	2.26E-16	**5.65E-17**	2.26E-16	3.00E-03	2.26E-16	2.26E-16	2.26E-16	2.26E-16	3.00E-03
	BEST	6.53E-01	6.53E-01	6.53E-01	6.53E-01	6.53E-01	**8.33E-01**	3.20E-01	6.53E-01	6.53E-01	6.53E-01	6.53E-01	6.53E-01	6.53E-01	6.53E-01
Blood	MEAN	7.96E-01	7.97E-01	8.01E-01	**8.03E-01**	7.94E-01	6.53E-01	8.00E-01	8.00E-01	7.97E-01	**8.03E-01**	7.99E-01	7.96E-01	8.00E-01	7.99E-01
	STD	1.10E-02	9.00E-03	7.00E-03	1.00E-02	2.10E-02	**2.26E-16**	**2.26E-16**	1.00E-02	7.00E-03	1.30E-02	1.10E-02	1.70E-02	1.00E-02	6.00E-03
	BEST	8.13E-01	8.13E-01	**8.40E-01**	**8.40E-01**	8.26E-01	6.53E-01	8.00E-01	8.13E-01	8.13E-01	**8.40E-01**	**8.40E-01**	**8.40E-01**	**8.40E-01**	8.13E-01
Circles	MEAN	**1.00E+00**	8.97E-01	8.55E-01	8.81E-01	8.80E-01	7.97E-01	8.00E-01	8.84E-01	9.26E-01	9.97E-01	**1.00E+00**	8.43E-01	9.86E-01	8.76E-01
	STD	**0.00E+00**	8.90E-02	6.00E-02	7.30E-02	7.10E-02	1.30E-02	2.26E-16	8.40E-02	6.40E-02	1.00E-02	**0.00E+00**	8.00E-02	3.70E-02	8.60E-02
	BEST	**1.00E+00**	**1.00E+00**	9.60E-01	**1.00E+00**	**1.00E+00**	8.26E-01	8.00E-01	**1.00E+00**	**1.00E+00**	**1.00E+00**	**1.00E+00**	**1.00E+00**	**1.00E+00**	**1.00E+00**
Diagnosis_II	MEAN	**8.33E-01**	**8.33E-01**	**8.33E-01**	**8.33E-01**	**8.33E-01**	**8.33E-01**	0.00E+00	**8.33E-01**	**8.33E-01**	**8.33E-01**	**8.33E-01**	**8.33E-01**	**8.33E-01**	**8.33E-01**
	STD	2.26E-16	2.26E-16	2.26E-16	2.26E-16	2.26E-16	2.26E-16	**0.00E+00**	2.26E-16	2.26E-16	2.26E-16	2.26E-16	2.26E-16	2.26E-16	2.26E-16
	BEST	**8.33E-01**	**8.33E-01**	**8.33E-01**	**8.33E-01**	**8.33E-01**	**8.33E-01**	0.00E+00	**8.33E-01**	**8.33E-01**	**8.33E-01**	**8.33E-01**	**8.33E-01**	**8.33E-01**	**8.33E-01**
Ecoli	MEAN	6.23E-01	6.40E-01	6.07E-01	7.90E-01	7.74E-01	7.18E-01	**9.58E-01**	8.20E-02	7.48E-01	7.19E-01	7.89E-01	6.78E-01	7.94E-01	7.00E-01
	STD	9.90E-02	1.13E-01	1.45E-01	3.50E-02	3.00E-02	1.50E-02	**0.00E+00**	1.07E-01	7.10E-02	1.00E-02	8.90E-02	8.70E-02	7.00E-03	9.00E-02
	BEST	7.94E-01	7.94E-01	8.91E-01	8.91E-01	8.23E-01	8.52E-01	**9.58E-01**	7.64E-01	8.82E-01	8.23E-01	8.23E-01	7.94E-01	8.23E-01	8.64E-01
Flame	MEAN	**9.19E-01**	8.15E-01	8.04E-01	8.51E-01	7.86E-01	8.56E-01	7.08E-01	7.98E-01	8.22E-01	8.16E-01	8.51E-01	8.00E-01	8.01E-01	7.48E-01
	STD	6.80E-02	1.14E-01	1.06E-01	7.70E-02	**9.60E-02**	1.15E-01	2.26E-16	1.04E-01	1.02E-01	8.90E-02	9.20E-02	1.15E-01	1.18E-01	1.06E-01
	BEST	**1.00E+00**	**1.00E+00**	**1.00E+00**	9.58E-01	9.16E-01	**1.00E+00**	7.08E-01	9.58E-01	**1.00E+00**	9.58E-01	**1.00E+00**	**1.00E+00**	**1.00E+00**	**1.00E+00**
Glass	MEAN	**9.00E-01**	**9.00E-01**	**9.00E-01**	**9.00E-01**	**9.00E-01**	2.72E-01	9.00E-01	**9.00E-01**	**9.00E-01**	**9.00E-01**	**9.00E-01**	**9.00E-01**	**9.00E-01**	**9.00E-01**
	STD	**1.41E-17**	**1.41E-17**	**1.41E-17**	**1.41E-17**	**1.41E-17**	5.65E-17	**1.41E-17**	**1.41E-17**	**1.41E-17**	**1.41E-17**	**1.41E-17**	**1.41E-17**	**1.41E-17**	**1.41E-17**
	BEST	**9.00E-01**	**9.00E-01**	**9.00E-01**	**9.00E-01**	**9.00E-01**	2.72E-01	**9.00E-01**	**9.00E-01**	**9.00E-01**	**9.00E-01**	**9.00E-01**	**9.00E-01**	**9.00E-01**	**9.00E-01**
Heart	MEAN	6.86E-01	6.95E-01	6.62E-01	6.95E-01	6.62E-01	6.97E-01	4.07E-01	**7.13E-01**	6.64E-01	6.87E-01	6.92E-01	6.80E-01	7.08E-01	6.48E-01
	STD	**3.40E-02**	3.70E-02	6.40E-02	5.30E-2	5.20E-02	5.20E-02	1.69E-16	6.80E-02	5.90E-02	4.80E-02	4.30E-02	5.30E-02	7.10E-02	8.00E-02
	BEST	7.40E-01	7.77E-01	8.14E-01	7.77E-01	8.14E-01	8.14E-01	4.07E-01	8.51E-01	7.77E-01	7.40E-01	8.14E-01	7.77E-01	**8.88E-01**	8.14E-01
Ionosphere	MEAN	**8.11E-01**	7.18E-01	7.39E-01	6.03E-01	5.62E-01	6.70E-01	5.83E-01	7.99E-01	5.95E-01	6.41E-01	6.50E-01	6.02E-01	7.12E-01	6.04E-01
	STD	**5.50E-02**	7.40E-02	8.50E-02	7.60E-02	7.40E-02	8.70E-02	0.00E+00	2.90E-02	8.60E-02	7.90E-02	1.03E-01	8.10E-02	6.00E-02	1.00E-01
	BEST	**9.16E-01**	6.53E-01	8.88E-01	8.05E-01	6.94E-01	8.61E-01	5.83E-01	8.33E-01	8.33E-01	8.33E-01	8.61E-01	7.50E-01	8.33E-01	7.50E-01
Iris	MEAN	**8.00E-01**	**8.00E-01**	**8.00E-01**	**8.00E-01**	**8.00E-01**	**8.00E-01**	4.00E-01	**8.00E-01**	**8.00E-01**	**8.00E-01**	**8.00E-01**	**8.00E-01**	**8.00E-01**	**8.00E-01**
	STD	2.26E-16	2.26E-16	2.26E-16	2.26E-16	2.26E-16	2.26E-16	**1.13E-16**	2.26E-16	2.26E-16	2.26E-16	2.26E-16	2.26E-16	2.26E-16	2.26E-16
	BEST	**8.00E-01**	**8.00E-01**	**8.00E-01**	**8.00E-01**	**8.00E-01**	**8.00E-01**	4.00E-01	**8.00E-01**	**8.00E-01**	**8.00E-01**	**8.00E-01**	**8.00E-01**	**8.00E-01**	**8.00E-01**

Table 6: Results of FNN's Trained by different nature inspired optimization algorithms on Last 15 benchmark data-sets

DataSet		MROM	GROM	BAT	CS	DE	FFA	GA	HHO	JAYA	MFO	PSO	SCA	SSA	WOA
	MEAN	**8.00E-01**	**8.00E-01**	**8.00E-01**	**8.00E-01**	**8.00E-01**	**8.00E-01**	4.00E-01	**8.00E-01**	**8.00E-01**	**8.00E-01**	**8.00E-01**	**8.00E-01**	**8.00E-01**	**8.00E-01**
Iris2D	STD	2.26E-16	2.26E-16	2.26E-16	2.26E-16	2.26E-16	2.26E-16	**1.13E-16**	2.26E-16	2.26E-16	2.26E-16	2.26E-16	2.26E-16	2.26E-16	2.26E-16
	BEST	**8.00E-01**	**8.00E-01**	**8.00E-01**	**8.00E-01**	**8.00E-01**	**8.00E-01**	4.00E-01	**8.00E-01**	**8.00E-01**	**8.00E-01**	**8.00E-01**	**8.00E-01**	**8.00E-01**	**8.00E-01**
	MEAN	**9.78E-01**	9.39E-01	9.63E-01	9.47E-01	9.42E-01	9.57E-01	3.42E-01	9.50E-01	9.40E-01	9.43E-01	9.67E-01	9.37E-01	9.57E-01	9.45E-01
Jain	STD	2.10E-02	2.40E-02	2.80E-02	3.40E-02	3.50E-02	2.50E-02	**1.69E-16**	2.70E-02	3.40E-02	2.00E-02	1.90E-02	3.80E-02	1.70E-02	2.50E-02
	BEST	**1.00E+00**	9.73E-01	**1.00E+00**	**1.00E+00**	**1.00E+00**	**1.00E+00**	3.42E-01	**1.00E+00**	**1.00E+00**	9.73E-01	**1.00E+00**	**1.00E+00**	9.73E-01	9.73E-01
	MEAN	**9.71E-01**	**9.71E-01**	9.70E-01	**9.71E-01**	**9.71E-01**	**9.71E-01**	9.00E-01	**9.71E-01**	**9.71E-01**	**9.71E-01**	**9.71E-01**	**9.71E-01**	**9.71E-01**	**9.71E-01**
Liver	STD	**1.13E-16**	**1.13E-16**	5.22E-03	**1.13E-16**	**1.13E-16**	**1.13E-16**	1.20E-16	**1.13E-16**	**1.13E-16**	**1.13E-16**	**1.13E-16**	**1.13E-16**	**1.13E-16**	**1.13E-16**
	BEST	**9.71E-01**	**9.71E-01**	**9.71E-01**	**9.71E-01**	**9.71E-01**	**9.71E-01**	9.00E-01	**9.71E-01**	**9.71E-01**	**9.71E-01**	**9.71E-01**	**9.71E-01**	**9.71E-01**	**9.71E-01**
	MEAN	9.48E-01	9.27E-01	9.24E-01	9.42E-01	9.25E-01	9.53E-01	4.39E-01	9.29E-01	9.45E-01	9.61E-01	**9.68E-01**	9.22E-01	9.51E-01	9.27E-01
Moons	STD	2.50E-02	1.70E-02	2.50E-02	1.70E-02	1.80E-02	2.30E-02	**1.29E-17**	1.50E-02	1.70E-02	1.30E-02	1.20E-02	1.50E-02	1.30E-02	1.70E-02
	BEST	**9.93E-01**	9.66E-01	9.66E-01	9.73E-01	9.73E-01	**9.93E-01**	4.40E-01	9.66E-01	9.86E-01	9.86E-01	9.86E-01	9.60E-01	9.80E-01	9.66E-01
	MEAN	8.12E-01	8.05E-01	8.05E-01	8.03E-01	8.05E-01	**8.32E-01**	7.91E-01	8.02E-01	8.06E-01	8.03E-01	8.12E-01	8.05E-01	8.26E-01	8.04E-01
Mouse	STD	3.10E-02	1.40E-02	1.40E-02	2.70E-02	3.10E-02	1.40E-02	1.13E-16	2.10E-02	2.10E-02	1.40E-02	1.10E-02	3.90E-02	**1.00E-02**	2.80E-02
	BEST	**8.93E-01**	8.32E-01	8.32E-01	8.53E-01	8.32E-01	8.53E-01	7.91E-01	8.32E-01	8.53E-01	8.32E-01	8.32E-01	**8.93E-01**	8.32E-01	8.12E-01
	MEAN	8.32E-01	8.33E-01	8.28E-01	8.31E-01	**8.40E-01**	8.25E-01	8.66E-01	8.26E-01	8.30E-01	8.34E-01	8.34E-01	8.26E-01	8.30E-01	8.28E-01
Pathbased	STD	**1.00E-02**	3.40E-02	2.00E-02	1.20E-02	2.50E-02	1.40E-02	0.00E+00	2.50E-02	2.00E-02	6.00E-03	6.00E-03	2.60E-02	**1.00E-02**	1.44E-01
	BEST	8.66E-01	8.66E-01	8.66E-01	8.66E-01	**9.00E-01**	8.33E-01	8.66E-01	8.33E-01	8.66E-01	8.66E-01	8.66E-01	8.66E-01	8.33E-01	8.33E-01
	MEAN	5.96E-01	5.80E-01	**5.98E-01**	5.85E-01	5.82E-01	5.84E-01	4.28E-01	5.87E-01	5.80E-01	5.85E-01	5.84E-01	**5.98E-01**	5.82E-01	5.85E-01
Seeds	STD	3.20E-02	3.80E-02	2.70E-02	3.10E-02	4.40E-02	2.10E-02	**1.13E-16**	3.10E-02	3.60E-02	2.20E-02	2.40E-02	3.20E-02	2.40E-02	4.10E-02
	BEST	6.19E-01	6.19E-01	**6.66E-01**	6.19E-01	**6.66E-01**	6.19E-01	4.28E-01	6.19E-01	6.19E-01	6.19E-01	6.19E-01	6.19E-01	6.19E-01	6.19E-01
	MEAN	5.40E-01	5.60E-01	**6.23E-01**	5.14E-01	5.58E-01	5.76E-01	6.40E-01	5.88E-01	5.33E-01	5.39E-01	5.38E-01	5.35E-01	5.44E-01	6.07E-01
Smiley	STD	5.10E-02	7.30E-02	4.60E-02	6.80E-02	1.00E-01	7.50E-02	1.13E-16	6.60E-02	9.60E-02	5.00E-02	6.20E-02	1.05E-01	5.50E-02	7.70E-02
	BEST	6.40E-01	6.40E-01	6.40E-01	7.40E-01	6.40E-01	**7.80E-01**	6.40E-01	6.60E-01	6.40E-01	6.40E-01	6.40E-01	6.40E-01	6.40E-01	**7.80E-01**
	MEAN	7.66E-01	7.31E-01	7.84E-01	7.26E-01	7.04E-01	7.34E-01	3.80E-01	**8.58E-01**	7.04E-01	7.55E-01	7.31E-01	7.04E-01	8.01E-01	6.57E-01
Sonar	STD	6.70E-02	1.00E-01	7.90E-02	6.60E-02	7.00E-02	7.90E-02	**1.13E-16**	5.20E-02	5.50E-02	6.80E-02	7.00E-02	9.10E-02	8.50E-02	7.90E-02
	BEST	8.57E-01	9.04E-01	9.04E-01	8.57E-01	8.57E-01	9.04E-01	3.80E-01	9.52E-01	8.57E-01	8.57E-01	8.57E-01	9.52E-01	9.52E-01	8.57E-01
	MEAN	**6.90E-01**	6.54E-01	6.59E-01	6.80E-01	6.70E-01	6.85E-01	3.53E-01	6.66E-01	6.67E-01	6.62E-01	6.87E-01	6.70E-01	6.89E-01	6.54E-01
Varied	STD	**7.00E-03**	2.70E-02	5.80E-02	1.30E-02	1.80E-02	1.40E-02	5.65E-17	2.30E-02	1.70E-02	1.90E-02	1.10E-02	1.80E-02	8.00E-03	2.60E-02
	BEST	**7.00E-01**	**7.00E-01**	6.93E-01	**7.00E-01**	6.93E-01	**7.00E-01**	3.53E-01	**7.00E-01**	**7.00E-01**	**7.00E-01**	**7.00E-01**	6.93E-01	**7.00E-01**	**7.00E-01**
	MEAN	**8.00E-01**	**8.00E-01**	**8.00E-01**	**8.00E-01**	**8.00E-01**	**8.00E-01**	4.00E-01	**8.00E-01**	7.95E-01	**8.00E-01**	**8.00E-01**	7.64E-01	**8.00E-01**	7.97E-01
Vary-Density	STD	**2.26E-16**	**2.26E-16**	**2.26E-16**	**2.26E-16**	**2.26E-16**	**2.26E-16**	**2.26E-16**	**2.26E-16**	1.60E-02	**2.26E-16**	**2.26E-16**	3.30E-02	**2.26E-16**	1.20E-02
	BEST	**8.00E-01**	**8.00E-01**	**8.00E-01**	**8.00E-01**	**8.00E-01**	**8.00E-01**	4.00E-01	**8.00E-01**	**8.00E-01**	**8.00E-01**	**8.00E-01**	**8.00E-01**	**8.00E-01**	**8.00E-01**
	MEAN	**8.08E-01**	7.90E-01	7.84E-01	8.04E-01	7.55E-01	7.99E-01	7.41E-01	7.94E-01	7.69E-01	7.73E-01	7.95E-01	7.65E-01	8.06E-01	7.54E-01
Vertebral2	STD	3.20E-02	6.40E-02	3.90E-02	4.70E-02	5.20E-02	4.50E-02	**0.00E+00**	4.50E-02	4.90E-02	6.60E-02	4.60E-02	4.40E-02	4.00E-02	5.80E-02
	BEST	**9.03E-01**	8.70E-01	8.70E-01	**9.03E-01**	8.38E-01	**9.03E-01**	7.41E-01	8.70E-01	8.38E-01	8.70E-01	8.38E-01	8.70E-01	8.70E-01	8.70E-01
	MEAN	8.59E-01	8.51E-01	8.53E-01	8.56E-01	8.48E-01	**8.61E-01**	7.93E-01	8.48E-01	8.51E-01	8.53E-01	8.56E-01	8.52E-01	8.54E-01	8.52E-01
Vertebral3	STD	**1.50E-02**	3.80E-02	2.90E-02	3.10E-02	3.50E-02	4.30E-02	5.65E-17	3.00E-02	3.30E-02	2.50E-02	2.70E-02	3.10E-02	3.00E-02	3.20E-02
	BEST	8.90E-01	**9.54E-01**	8.22E-01	9.22E-01	9.22E-01	**9.54E-01**	7.93E-01	8.90E-01	9.22E-01	8.90E-01	9.22E-01	9.22E-01	8.90E-01	8.90E-01
	MEAN	9.36E-01	9.33E-01	9.17E-01	9.36E-01	9.23E-01	**9.44E-01**	2.98E-01	9.41E-01	9.28E-01	9.32E-01	9.33E-01	9.21E-01	9.33E-01	9.14E-01
Wdbc	STD	1.50E-02	2.50E-02	2.60E-02	2.40E-02	2.50E-02	2.60E-02	**1.13E-16**	1.90E-02	1.40E-02	2.30E-02	4.30E-02	2.50E-02	1.50E-02	4.30E-02
	BEST	**9.82E-01**	**9.82E-01**	9.64E-01	**9.82E-01**	9.64E-01	**9.82E-01**	2.98E-01	**9.82E-01**	9.64E-01	**9.82E-01**	9.64E-01	9.64E-01	9.64E-01	**9.82E-01**
	MEAN	7.03E-01	6.92E-01	6.96E-01	7.05E-01	7.01E-01	7.29E-01	5.22E-01	6.92E-01	6.96E-01	7.53E-01	7.37E-01	**7.62E-01**	7.11E-01	6.96E-01
Wine	STD	3.50E-02	2.00E-02	2.40E-02	3.90E-02	3.70E-02	5.60E-02	**2.82E-17**	1.40E-02	3.10E-02	5.60E-02	5.20E-02	8.60E-02	4.20E-02	1.90E-02
	BEST	8.00E-01	8.00E-01	7.44E-01	8.00E-01	8.55E-01	8.55E-01	5.22E-01	7.44E-01	8.00E-01	**9.11E-01**	8.55E-01	**9.11E-01**	8.55E-01	7.44E-01

complexity of the composition functions in the CEC14 benchmark constitutes a substantial obstacle for optimization methods in general. In this regard, the efficacy of optimization methods such as GROM and MROM variants is assessed based on their capability to identify near-optimal solutions for these functions.

Despite the high complexity of composition functions, MROM variants were able to provide near-optimal solutions for both the functions. Notably, MROM demonstrated a remarkable improvement over GROM in the optimization problems represented by functions f_{129} and f_{130}, as evidenced by the superior best objective function value and lower standard deviation.

6.2.2. Comparison of various metal ratios on CEC14 test functions

In the field of optimization, it is important to identify methods that can consistently produce high-quality solutions to a wide range of problems. In the context of MROM framework, results presented in this study has shown that the silver ratio emerges as the best ratio for this purpose.

This conclusion is based on the evaluation of fitness across a set of 30 CEC14 benchmark test functions. The results in Table 4 show that the silver ratio consistently performs well across the CEC14 functions. Fig. 11 shows the comparison of distinct metal ratios used in this study on the basis of mean objective function value, best objective function value, and standard deviation in CEC 2014 benchmark functions. In particular, silver ratio based MROM method has the highest average fitness in 15 out of the 30 functions, the lowest standard deviation in 13 out of 30 functions, and the best minimum value in 14 out of 30 functions. This is a significant finding as it highlights the superiority of the silver ratio over other metal ratios in the MROM framework.

6.3. Result and Discussion in Feedforward Neural Network (Multi-layer Perceptron) Training

The problem of training FNN is discussed in this section through the use of the suggested MROM framework. Prior to presenting the experimental setup, findings, and analysis, the design of FNN and the representation of search agents is described.

6.3.1. Design of Feed-Forward Neural Network

The number of hidden layers and the number of nodes in each layer are the most important factors to consider while building a FNN. The strategy suggested in Mirjalili et al. (2014) for the design of FNN has been employed in this study. A single hidden layer FNN has been used and the number of hidden layer neurons has been calculated using the Eq. 41.

$$H = 2 \times I + 1 \tag{41}$$

Here, H is number of hidden layer neurons, and I in number of input feature.

6.3.2. Results and Discussions on Training of FNN

Table 7: Parameters for various algorithms

Algorithm	Parameter	Value
BAT	Loudness	0.5
	Pulse rate	0.5
	Frequency	[0,2]
CS	Discovery rate	0.25
	Beta	1.5
DE	Crossover probability	0.9
	Differential weight	0.5
FFA	Randomness	0.5
	Minimum value of Beta	0.2
	Absorption coefficient	1
GA	Crossover probability	0.9
	Mutation probability 0.01	
	Selection mechanism	Roulette wheel
GROM	No parameter	
HHO	Rabbit Energy	[2,0]
	Beta	1.5
JAYA	No parameter	
MFO	T	[-1,1]
	a	1-t/T
	b	1
	Flame	round(N_l × (N_1)/T)
MROM	No parameter	
PSO	Acceleration constants	[2,2]
	Inertia weights	[0.9, 0.6]
SCA	r1	[0,2]
	r2	[0,2]
	r3	[0,2]
	a	2
SSA	Flame number	round(N-Iter*((N-1)/Max_iter))
WOA	P	Rand(0,1)
	b	1

To examine the efficacy of the proposed method, a set of 30 data-sets taken from the University of California at Irvine (UCI) Machine Learning repository (Lichman (2013)) and DELVE repository (Rasmussen et al. (1996)) has been used. A short summary of these data sets is given in Table 8 that contains information about the number of classes, number of features, number of instances in the training set, and number of instances in the testing set. Every data-set used in the proposed study is split into 90% for training and 10% for testing. To maintain the class distribution, all data sets are stratified sampled. All input features are normalized using Eq. 42 to eliminate the impact of features with varying scales.

$$k_i' = \frac{k_i - min_d}{max_d - min_d} \tag{42}$$

Where, k_i' is the feature normalized value in the interval of $[min_d, max_d]$ and k_i is the i^{th} value in the dataset.

Table 8: Description of Various Benchmark Dataset

Dataset	Classes	Features	Training Sample	Testing sample
Aggregation	6	2	709	79
Aniso	3	2	1350	150
Appendicitis	2	7	96	10
Balance	3	4	563	62
Banknote	2	4	1235	137
Blobs	3	2	1350	150
Blood	2	4	673	75
Circles	2	2	1350	150
Diagnosis_II	2	6	108	12
Ecoli	5	7	302	34
Flame	2	2	216	24
Glass	4	9	193	21
Heart	2	13	243	27
Ionosphere	2	34	316	35
Iris	3	4	135	15
Iris2D	3	2	135	15
Jain	2	2	336	37
Liver	3	6	310	35
Moons	2	2	1350	150
Mouse	3	2	441	49
Path-based	3	2	270	30
Seeds	2	7	189	21
Smiley	4	2	450	50
Sonar	2	60	187	21
Varied	3	2	1350	150
Vary-Density	3	2	135	15
Vertebral2	2	6	279	31
Vertebral 3	3	6	279	31
Wdbc	2	30	512	57
Wine	3	13	160	18

In order to compare the proposed method to the existing techniques a set of 13 state-of-the-art techniques were used which includes Bat Optimization Algorithm (BAT) (Yang & Gandomi (2012)), Cuckoo Search Algorithm (CS) Gandomi et al. (2013), Differential Evolution (DE) (Feoktistov (2006)), Fire-Fly Algorithm (FFA) (Yang & Slowik (2020)), Genetic Algorithm (Mirjalili (2019)), Harris Hawk optimization Algorithm(HHO) (Heidari et al. (2019)), Jaya Optimization Algorithm (JAYA) (Rao (2016)), Moth-flame Optimization Algorithm (MFO) (Mirjalili (2015)), Particle Swarm Optimization (PSO) (Kennedy & Eberhart (1995)), Sine Cosine Algorithm (SCA) (Mirjalili (2016)), Salp Swarm Algorithm (SSA) (Mirjalili et al. (2017)), Whale Optimization Algorithm (WAO) (Mirjalili & Lewis (2016)) and the MROM's Counterpart GROM. For these experiments, only MROM with silver metal ratio has been used since it's the most promising among other variants of MROM framework. A comparison is made on basis of the arithmetic mean of the accuracy (MEAN), standard deviation (STD), and best accuracy (Best) for a total 30 runs, each run having a total of 100 iterations with a population size equal to 50. Table 7 gives tuning parameter for each algorithm.

Results of feed-forward neural networks trained by distinct algorithms are presented in Table 5 and Table 6. MROM, FFA, MFO, PSO, SSA, and WOA perform better than the others for the data-set Aniso. MROM has high accuracy than the others for the Banknote data set, even achieving 100 percent in the best instance. For Blobs dataset, FFA's performance is considerably better than its competitors. In the data set Blood, the CS performs better than the others. MROM once more proves to be the best solution for feed-forward neural network training for the circles dataset. In the Diagnosis II dataset, every algorithm performs adequately. The genetic algorithm was the most effective for the ecoli dataset. All algorithms, with the exception of genetic algorithms, have low accuracy for the data set aggregation. FFA had the best performance for the dataset Balance. Despite the fact that both MROM and WOA have higher accuracy for the data-set Appendics, MROM outperforms WOA because of its better lower standard deviation. According to this study conducted for the datasets Ansio, Appenditics, Banknote, Circles, Diagnosis II, Flame, Ionosphere, Iris, Iris2D, Jain, Liver, Varied, Vary-Density, and Vertebral2, the MROM trainer has shown superior performance compared to other optimizers. The mean accuracy achieved by the MROM trainer was 7.07E-01, 8.21E-01, 9.92E-01, 1.00E+00, 8.33E-01, 9.19E-01, 9.00E-01, 8.11E-01, 8.00E-01, 8.00E-01, 9.78E-01, 9.71E-01, 6.90E-01, 8.00E-01, and 8.00E-01 on these datasets, respectively. Out of 30 datasets, the suggested method MROM was able to outperform its counterpart on 16 datasets. The FFA algorithm emerged as the second-best trainer in our analysis.

This experimental study has demonstrated that the MROM trainer is a dependable method and achieve the best values for weights and biases of FNN, as evidenced by the high average accuracy and low standard deviation achieved. This implies that the MROM trainer can optimize FNN's performance proficiently without getting stuck in suboptimal solutions. This finding is noteworthy from an academic viewpoint, as it contributes to the expanding research on the efficacy of nature-inspired optimization algorithms in solving complex issues across different domains.

7. Conclusion and future work

In this paper, a novel optimization framework called the Metal Ratio Optimization Method (MROM), which is an extension of the popular Golden Ratio Optimization Method (GROM), is proposed. The proposed framework aims to improve the performance of GROM by introducing a new search mechanism and exploring various metal ratios. The performance of the MROM-based algorithms is compared with GROM on 23 well-known benchmark functions and the CEC 2014 benchmark function suite. The experimental results showed that MROM-based algorithms outperformed GROM in terms of achieving better mean fitness values of the objective function over several runs. The proposed framework algorithms achieved better convergence rates and higher accuracy in finding the optimal solutions. The complexity analysis of MROM revealed that it has the same complexity as GROM, which is an important factor to consider in real-world applications where computational efficiency is crucial. The study on the CEC 2014 benchmark function suite reveals that the silver ratio-based MROM_2 algorithm is the best among the other variants of the MROM framework. Moreover, the silver ratio-based MROM optimizer is utilized to train feed-forward neural networks and demonstrate its effectiveness and competitive performance against state-of-the-art optimization methods on thirty classification datasets.

In the future, MROM can be extended to handle multi-objective and constrained optimization problems. Furthermore, it would be interesting to explore the performance of the proposed method in training other types of neural networks, such as convolutional neural networks and recurrent neural networks. Additionally, more experiments can be performed to further improve the performance of MROM by incorporating various mathematical operators or heuristics, such as crossover, mutation, and chaotic perturbation, etc.

References

Abiodun, O. I., Jantan, A., Omolara, A. E., Dada, K. V., Mohamed, N. A., & Arshad, H. (2018). State-of-the-art in artificial neural network applications: A survey. *Heliyon*, *4*, e00938. URL: `https://www.sciencedirect.com/science/article/pii/S2405844018332067`. doi:`https://doi.org/10.1016/j.heliyon.2018.e00938`.

Amirsadri, S., Mousavirad, S. J., & Ebrahimpour-Komleh, H. (2018). A levy flight-based grey wolf optimizer combined with back-propagation algorithm for neural network training. *Neural Computing and Applications*, *30*, 3707–3720. URL: `https://link.springer.com/article/10.1007/s00521-017-2952-5`. doi:`https://doi.org/10.1007/s00521-017-2952-5`.

Azad, A. K., Wang, L., Guo, N., Tam, H.-Y., & Lu, C. (2016). Signal processing using artificial neural network for botda sensor system. *Opt. Express*, *24*, 6769–6782. URL: `https://opg.optica.org/oe/abstract.cfm?URI=oe-24-6-6769`. doi:`10.1364/OE.24.006769`.

Bebis, G., & Georgiopoulos, M. (1994). Feed-forward neural networks. *IEEE Potentials*, *13*, 27–31. URL: `https://ieeexplore.ieee.org/abstract/document/329294`. doi:`10.1109/45.329294`.

Bhoi, A. K., Mallick, P. K., Liu, C.-M., & Balas, V. E. (Eds.) (2021). *Bio-inspired Neurocomputing*. Springer Singapore. URL: `https://doi.org/10.1007%2F978-981-15-5495-7`. doi:`10.1007/978-981-15-5495-7`.

Bilal, Pant, M., Zaheer, H., Garcia-Hernandez, L., & Abraham, A. (2020). Differential evolution: A review of more than two decades of research. *Eng. Appl. Artif. Intell.*, *90*. URL: `https://doi.org/10.1016/j.engappai.2020.103479`. doi:`10.1016/j.engappai.2020.103479`.

Bohat, V. K., & Arya, K. (2018). An effective gbest-guided gravitational search algorithm for real-parameter optimization and its application in training of feedforward neural networks. *Knowledge-Based Systems*, *143*, 192–207.

Bridle, J. S. (1990). Probabilistic interpretation of feedforward classification network outputs, with relationships to statistical pattern recognition. In F. F. Soulié, & J. Hérault (Eds.), *Neurocomputing* (pp. 227–236). Berlin, Heidelberg: Springer Berlin Heidelberg.

Buhmann, M. D. (2000). Radial basis functions. *Acta Numerica*, *9*, 1–38. doi:`10.1017/S0962492900000015`.

Buhmann, M. D. (2003). *Radial basis functions: theory and implementations* volume 12. Cambridge university press.

Cachim, P., & Bezuijen, A. (2019). Modelling the torque with artificial neural networks on a tunnel boring machine. *KSCE Journal of Civil Engineering*, *23*, 4529–4537. URL: `https://link.springer.com/article/10.1007/s12205-019-0302-0`. doi:`https://doi.org/10.1007/s12205-019-0302-0`.

Chattopadhyay, S., Dey, A., Singh, P. K., Geem, Z. W., & Sarkar, R. (2021). Covid-19 detection by optimizing deep residual features with improved clustering-based golden ratio optimizer. *Diagnostics*, *11*. URL: `https://www.mdpi.com/2075-4418/11/2/315`. doi:`10.3390/diagnostics11020315`.

Chen, Y.-C., Ke, W.-C., & Chiu, H.-W. (2014). Risk classification of cancer survival using ann with gene expression data from multiple laboratories. *Computers in Biology and Medicine*, *48*, 1–7. URL: `https://www.sciencedirect.com/science/article/pii/S0010482514000377`. doi:`https://doi.org/10.1016/j.compbiomed.2014.02.006`.

Chopard, B., & Tomassini, M. (2018). *An introduction to metaheuristics for optimization*. Springer. URL: `https://link.springer.com/book/10.1007/978-3-319-93073-2`. doi:`https://doi.org/10.1007/978-3-319-93073-2`.

Dey, A., Chattopadhyay, S., Singh, P. K., Ahmadian, A., Ferrara, M., & Sarkar, R. (2020). A hybrid meta-heuristic feature selection method using golden ratio and equilibrium optimization algorithms for speech emotion recognition. *IEEE Access*, *8*, 200953–200970.

Fausett, L. V. (2006). *Fundamentals of neural networks: architectures, algorithms and applications*. Pearson Education India.

Feoktistov, V. (2006). *Differential evolution*. Springer.

Gandomi, A. H., Yang, X.-S., & Alavi, A. H. (2013). Cuckoo search algorithm: a metaheuristic approach to solve structural optimization problems. *Engineering with computers*, *29*, 17–35.

Gautam, P. (2016). System identification of nonlinear inverted pendulum using artificial neural network. In *2016 International Conference on Recent Advances and Innovations in Engineering (ICRAIE)* (pp. 1–5). URL: `https://ieeexplore.ieee.org/abstract/document/7939522`. doi:`10.1109/ICRAIE.2016.7939522`.

Ghosh-Dastidar, S., & Adeli, H. (2009). Spiking neural networks. *International journal of neural systems*, *19*, 295–308. URL: `https://www.worldscientific.com/doi/abs/10.1142/S0129065709002002`. doi:`https://doi.org/10.1142/S0129065709002002`.

Glover, F. W., & Kochenberger, G. A. (2006). *Handbook of metaheuristics* volume 57. Springer Science & Business Media.

Grosan, C., & Abraham, A. (2011). Artificial neural networks. In *Intelligent Systems: A Modern Approach* (pp. 281–323). Berlin, Heidelberg: Springer Berlin Heidelberg. URL: `https://doi.org/10.1007/978-3-642-21004-4_12`. doi:`10.1007/978-3-642-21004-4_12`.

Haji, S. H., & Abdulazeez, A. M. (2021). Comparison of optimization techniques based on gradient descent algorithm: A review. *PalArch's Journal of Archaeology of Egypt / Egyptology*, *18*, 2715–2743. URL: `https://www.archives.palarch.nl/index.php/jae/article/view/6705`.

Hamzaçebi, C. (2008). Improving artificial neural networks' performance in seasonal time series forecasting. *Information Sciences*, *178*, 4550–4559. URL: `https://www.sciencedirect.com/science/article/pii/S0020025508002958`. doi:`https://doi.org/10.1016/j.ins.2008.07.024`. Including Special Section: Genetic and Evolutionary Computing.

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. URL: `https://www.sciencedirect.com/science/article/pii/S0167739X18313530`. doi:`https://doi.org/10.1016/j.future.2019.02.028`.

Huang, M.-L., & Chou, Y.-C. (2019). Combining a gravitational search algorithm, particle swarm optimization, and fuzzy rules to improve the classification performance of a feed-forward neural network. *Computer Methods and Programs in Biomedicine*, *180*, 105016. URL: `https://www.sciencedirect.com/science/article/pii/S0169260719304614`.

doi:https://doi.org/10.1016/j.cmpb.2019.105016.

Jameson, A. (1995). Gradient based optimization methods. *MAE Technical Report No*, .

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

Ku, K. W., Mak, M.-W., & Siu, W.-C. (1995). A cellular genetic algorithm for training recurrent neural networks. In *Proceedings of the International Conference on Neural Networks and Signal Processing* (pp. 140–143).

Kuo, R., Hong, S., & Huang, Y. (2010). Integration of particle swarm optimization-based fuzzy neural network and artificial neural network for supplier selection. *Applied Mathematical Modelling*, *34*, 3976–3990. URL: https://www.sciencedirect.com/science/article/pii/S0307904X10001526. doi:https://doi.org/10.1016/j.apm.2010.03.033.

Li, X., Wen, J., & Bai, E.-W. (2016). Developing a whole building cooling energy forecasting model for on-line operation optimization using proactive system identification. *Applied Energy*, *164*, 69–88. URL: https://www.sciencedirect.com/science/article/pii/S0306261915015688. doi:https://doi.org/10.1016/j.apenergy.2015.12.002.

Liang, J. J., Qu, B. Y., & Suganthan, P. N. (2013). Problem definitions and evaluation criteria for the cec 2014 special session and competition on single objective real-parameter numerical optimization. *Computational Intelligence Laboratory, Zhengzhou University, Zhengzhou China and Technical Report, Nanyang Technological University, Singapore*, *635*, 490.

Lichman, M. (2013). UCI machine learning repository. URL: http://archive.ics.uci.edu/ml.

Lin, W., Wu, G., Wang, X., & Li, K. (2020). An artificial neural network approach to power consumption model construction for servers in cloud data centers. *IEEE Transactions on Sustainable Computing*, *5*, 329–340. URL: https://ieeexplore.ieee.org/abstract/document/8685195. doi:10.1109/TSUSC.2019.2910129.

Ling, Y., Zhou, Y., & Luo, Q. (2017). Lévy flight trajectory-based whale optimization algorithm for global optimization. *IEEE Access*, *5*, 6168–6186. URL: https://ieeexplore.ieee.org/abstract/document/7904636. doi:10.1109/ACCESS.2017.2695498.

Luo, Q., Li, J., Zhou, Y., & Liao, L. (2021). Using spotted hyena optimizer for training feedforward neural networks. *Cognitive Systems Research*, *65*, 1–16. URL: https://www.sciencedirect.com/science/article/pii/S1389041720300577. doi:https://doi.org/10.1016/j.cogsys.2020.09.001.

Luo, X., Oyedele, L. O., Ajayi, A. O., Akinade, O. O., Delgado, J. M. D., Owolabi, H. A., & Ahmed, A. (2020). Genetic algorithm-determined deep feedforward neural network architecture for predicting electricity consumption in real buildings. *Energy and AI*, *2*, 100015. URL: https://www.sciencedirect.com/science/article/pii/S266654682030015X. doi:https://doi.org/10.1016/j.egyai.2020.100015.

Mahmon, N. A., & Ya'acob, N. (2014). A review on classification of satellite image using artificial neural network (ann). In *2014 IEEE 5th Control and System Graduate Research Colloquium* (pp. 153–157). URL: https://ieeexplore.ieee.org/abstract/document/6908713. doi:10.1109/ICSGRC.2014.6908713.

Marinković, Z., Crupi, G., Caddemi, A., Marković, V., & Schreurs, D. M.-P. (2020). A review on the artificial neural network applications for small-signal modeling of microwave fets. *International Journal of Numerical Modelling: Electronic Networks, Devices and Fields*, *33*, e2668. URL: https://onlinelibrary.wiley.com/doi/abs/10.1002/jnm.2668. doi:https://doi.org/10.1002/jnm.2668.

Marugán, A. P., Márquez, F. P. G., Perez, J. M. P., & Ruiz-Hernández, D. (2018). A survey of artificial neural network in wind energy systems. *Applied Energy*, *228*, 1822–1836. URL: https://www.sciencedirect.com/science/article/pii/S0306261918311048. doi:https://doi.org/10.1016/j.apenergy.2018.07.084.

Mirjalili, S. (2015). Moth-flame optimization algorithm: A novel nature-inspired heuristic paradigm. *Knowledge-Based Systems*, *89*, 228–249. URL: https://www.sciencedirect.com/science/article/pii/S0950705115002580. doi:https://doi.org/10.1016/j.knosys.2015.07.006.

Mirjalili, S. (2016). Sca: A sine cosine algorithm for solving optimization problems. *Knowledge-Based Systems*, *96*, 120–133. URL: https://www.sciencedirect.com/science/article/pii/S0950705115005043. doi:https://doi.org/10.1016/j.knosys.2015.12.022.

Mirjalili, S. (2019). Genetic algorithm. In *Evolutionary algorithms and neural networks* (pp. 43–55). Springer. URL: https://link.springer.com/chapter/10.1007/978-3-319-93025-1_4. doi:https://doi.org/10.1007/s11042-020-10139-6.

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. URL: https://www.sciencedirect.com/science/article/pii/S0965997816307736. doi:https://doi.org/10.1016/j.advengsoft.2017.07.002.

Mirjalili, S., & Lewis, A. (2016). The whale optimization algorithm. *Advances in Engineering Software*, *95*, 51–67. URL: https://www.sciencedirect.com/science/article/pii/S0965997816300163. doi:https://doi.org/10.1016/j.advengsoft.2016.01.008.

Mirjalili, S., Mirjalili, S. M., & Lewis, A. (2014). Let a biogeography-based optimizer train your multi-layer perceptron. *Information Sciences*, *269*, 188–209. URL: https://www.sciencedirect.com/science/article/pii/S0020025514000747. doi:https://doi.org/10.1016/j.ins.2014.01.038.

Mohamed, A.-A. A., El-Gaafary, A. A. M., Mohamed, Y. S., & Hemeida, A. M. (2015). Design static var compensator controller using artificial neural network optimized by modify grey wolf optimization. In *2015 International Joint Conference on Neural Networks (IJCNN)* (pp. 1–7). URL: https://ieeexplore.ieee.org/abstract/document/7280704. doi:10.1109/IJCNN.2015.7280704.

Montana, D. J., & Davis, L. (1989). Training feedforward neural networks using genetic algorithms. In *Proceedings of the 11th International Joint Conference on Artificial Intelligence - Volume 1* IJCAI'89 (p. 762–767). San Francisco, CA, USA: Morgan Kaufmann Publishers Inc. URL: https://dl.acm.org/doi/abs/10.5555/1623755.1623876.

Nabil, E. (2016). A modified flower pollination algorithm for global optimization. *Expert Systems with Applications*, *57*, 192–203. URL: https://www.sciencedirect.com/science/article/pii/S0957417416301415. doi:https://doi.org/10.1016/j.eswa.2016.03.047.

Nematollahi, A. F., Rahiminejad, A., & Vahidi, B. (2020). A novel meta-heuristic optimization method based on golden ratio in nature. *Soft Computing*, *24*, 1117–1151. URL: https://link.springer.com/article/10.1007/s00500-019-03949-w. doi:https://doi.org/10.1007/s00500-019-03949-w.

Nusair, K., & Alasali, F. (2020). Optimal power flow management system for a power network with stochastic renewable energy resources using golden ratio optimization method. *Energies*, *13*. URL: https://www.mdpi.com/1996-1073/13/14/3671. doi:10.3390/en13143671.

Nwankpa, C., Ijomah, W., Gachagan, A., & Marshall, S. (2018). Activation functions: Comparison of trends in practice and research for deep learning. URL: https://arxiv.org/abs/1811.03378. doi:10.48550/ARXIV.1811.03378.

Ojha, V. K., Abraham, A., & Snášel, V. (2017). Metaheuristic design of feedforward neural networks: A review of two decades of research. *Engineering Applications of Artificial Intelligence*, *60*, 97–116. URL: https://www.sciencedirect.com/science/article/pii/S0952197617300234. doi:https://doi.org/10.1016/j.engappai.2017.01.013.

Price, K. V. (2013). Differential evolution. In I. Zelinka, V. Snášel, & A. Abraham (Eds.), *Handbook of Optimization: From Classical to Modern Approach* (pp. 187–214). Berlin, Heidelberg: Springer Berlin Heidelberg. URL: https://doi.org/10.1007/978-3-642-30504-7_8. doi:10.1007/978-3-642-30504-7_8.

Rao, R. (2016). Jaya: A simple and new optimization algorithm for solving constrained and unconstrained optimization problems. *International Journal of Industrial Engineering Computations*, *7*, 19–34. doi:10.5267/j.ijiec.2015.8.004.

Rasmussen, C. E., Neal, R. M., Hinton, G. E., van Camp, D., Revow, M., Ghahramani, Z., Kustra, R., & Tibshirani, R. (1996). The delve manual. *URL http://www. cs. toronto. edu/˜ delve*, .

Ruder, S. (2016). An overview of gradient descent optimization algorithms. URL: https://arxiv.org/abs/1609.04747. doi:10.48550/ARXIV.1609.04747.

Shalev-Shwartz, S., Shamir, O., & Shammah, S. (2017). Failures of gradient-based deep learning. In D. Precup, & Y. W. Teh (Eds.), *Proceedings of the 34th International Conference on Machine Learning* (pp. 3067–3075). PMLR volume 70 of *Proceedings of Machine Learning Research*. URL: https://proceedings.mlr.press/v70/shalev-shwartz17a.html.

Shaw, D., & Kinsner, W. (1996). Chaotic simulated annealing in multilayer feedforward networks. In *Proceedings of 1996 Canadian Conference on Electrical and Computer Engineering* (pp. 265–269 vol.1). volume 1. URL: https://ieeexplore.ieee.org/abstract/document/548088. doi:10.1109/CCECE.1996.548088.

Sherstinsky, A. (2020). Fundamentals of recurrent neural network (rnn) and long short-term memory (lstm) network. *Physica D: Nonlinear Phenomena*, *404*, 132306. URL: https://www.sciencedirect.com/science/article/pii/S0167278919305974. doi:https://doi.org/10.1016/j.physd.2019.132306.

Singh, S., & Bansal, J. C. (2022). Mutation-driven grey wolf optimizer with modified search mechanism. *Expert Systems with Applications*, *194*, 116450.

Smith, T., & Boning, D. (1997). A self-tuning ewma controller utilizing artificial neural network function approximation techniques. *IEEE Transactions on Components, Packaging, and Manufacturing Technology: Part C*, *20*, 121–132. URL: https://ieeexplore.ieee.org/abstract/document/622882. doi:10.1109/3476.622882.

Svozil, D., Kvasnicka, V., & Pospichal, J. (1997). Introduction to multi-layer feed-forward neural networks. *Chemometrics and Intelligent Laboratory Systems*, *39*, 43–62. URL: https://www.sciencedirect.com/science/article/pii/S0169743997000610. doi:https://doi.org/10.1016/S0169-7439(97)00061-0.

Tealab, A., Hefny, H., & Badr, A. (2017). Forecasting of nonlinear time series using ann. *Future Computing and Informatics Journal*, *2*, 39–47. URL: https://www.sciencedirect.com/science/article/pii/S2314728817300144. doi:https://doi.org/10.1016/j.fcij.2017.05.001.

Wang, S., Zhang, Y., Dong, Z., Du, S., Ji, G., Yan, J., Yang, J., Wang, Q., Feng, C., & Phillips, P. (2015). Feed-forward neural network optimized by hybridization of pso and abc for abnormal brain detection. *International Journal of Imaging Systems and Technology*, *25*, 153–164. URL: https://onlinelibrary.wiley.com/doi/abs/10.1002/ima.22132. doi:https://doi.org/10.1002/ima.22132. arXiv:https://onlinelibrary.wiley.com/doi/pdf/10.1002/ima.22132.

Yang, X.-S., & Gandomi, A. H. (2012). Bat algorithm: a novel approach for global engineering optimization. *Engineering computations*, .

Yang, X.-S., & Slowik, A. (2020). Firefly algorithm. In *Swarm Intelligence Algorithms* (pp. 163–174). CRC Press.

Yegnanarayana, B. (2009). *Artificial neural networks*. PHI Learning Pvt. Ltd.

Zhang, J.-R., Zhang, J., Lok, T.-M., & Lyu, M. R. (2007). A hybrid particle swarm optimization–back-propagation algorithm for feedforward neural network training. *Applied Mathematics and Computation*, *185*, 1026–1037. URL: https://www.sciencedirect.com/science/article/pii/S0096300306008277. doi:https://doi.org/10.1016/j.amc.2006.07.025. Special Issue on Intelligent Computing Theory and Methodology.

Zhang, Y., Phillips, P., Wang, S., Ji, G., Yang, J., & Wu, J. (2016). Fruit classification by biogeography-based optimization and feedforward neural network. *Expert Systems*, *33*, 239–253. URL: https://onlinelibrary.wiley.com/doi/abs/10.1111/exsy.12146. doi:https://doi.org/10.1111/exsy.12146. arXiv:https://onlinelibrary.wiley.com/doi/pdf/10.1111/exsy.12146.

Appendix A. Details of benchmark Functions

Table A.9: List of first group of benchmark functions

Function	Mathematical Representation	Dim	Global Minima	Range
f_1	$f_1(X) = \sum_{i=1}^{Dim} x_i^2$	30	0	$(-100, 100)$
f_2	$f_2(X) = \sum_{i=1}^{Dim} \lvert x_i \rvert + \prod_{i=1}^{Dim} \lvert x_i \rvert$	30	0	$(-10, 10)$
f_3	$f_3(X) = \sum_{i=1}^{Dim} \left(\sum_{i=1}^{Dim} x_i \right)^2$	30	0	$(-100, 100)$
f_4	$f_4(X) = \max_i \left\{ \lvert x_i \rvert, 1 \leq i \leq D \right\}$	30	0	$(-100, 100)$
f_5	$f_5(X) = \sum_{i=1}^{Dim-1} [100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2]$	30	0	$(-30, 30)$
f_6	$f_6(X) = \sum_{i=1}^{Dim} (\lfloor x_i + 0.5 \rfloor)^2$	30	0	$(-100, 100)$
f_7	$f_7(X) = \sum_{i=1}^{Dim} ix_i + rand[0, 1)$	30	0	$(-1.28, 1.28)$
f_8	$f_8(X) = -\sum_{i=1}^{Dim} (x_i sin(\sqrt{\lvert x_i \rvert}))$	30	-12569.5	$(-500, 500)$
f_9	$f_9(X) = \sum_{i=1}^{Dim} [x_i^2 - 10cos(2\pi x_i) + 10]$	30	0	$(-5.12, 5.12)$
f_{10}	$f_{10}(X) = -20exp\left(-0.2 \sqrt{\frac{1}{D} \sum_{i=1}^{Dim} x_i^2} \right)$	30	0	$(-32, 32)$
f_{11}	$f_{11}(X) = \frac{1}{4000} \sum_{i=1}^{Dim} x_i^2 - \prod_{i=1}^{Dim} cos\left(\frac{x_i}{\sqrt{i}} \right) + 1$	30	0	$(-600, 600)$
f_{12}	$f_{12}(X) = \frac{\pi}{Dim} \left\{ sin^2(3\pi y_1) + \sum_{i=1}^{Dim-1} (y_i - 1)^2 [1 + sin^2(3\pi y_{i+1})] \right.$	30	0	$(-50, 50)$
	$\left. +(y_{Dim} - 1)^2 \right\} + \sum_{i=1}^{Dim} u(x_i, 10, 100, 4)$			
f_{13}	$f_{13}(X) = 0.1 \left\{ sin^2(3\pi x_1) + \sum_{i=1}^{Dim-1} (x_i - 1)^2 [1 + sin^2(3\pi x_{i+1})] \right.$	30	0	$(-50, 50)$
	$\left. +(x_{Dim} - 1)^2 [1 + sin^2(3\pi x_{Dim})] \right\} + \sum_{i=1}^{Dim} u(x_i, 5, 100, 4)$			
f_{14}	$f_{14}(X) = \left[\frac{1}{500} + \sum_{j=1}^{25} \frac{1}{j + \sum_{i=1}^{2} (x_i - a_{ij})^6} \right]^{-1}$	2	1	$(-65.536, 65.536)$
f_{15}	$f_{15}(X) = \sum_{i=1}^{11} \left[a_i - \frac{x_1(b_i^2 + b_i x_2)}{b_i^2 + b_i x_2 + x_4} \right]^2$	4	0.0003	$(-5, 5)$
f_{16}	$f_{16}(X) = 4x_1^2 - 2.1x_1^4 + \frac{1}{3}x_1^6 + x_1 x_2 - 4x_2^2 + 4x_2^4$	2	-1.0316	$(-5, 5)$
f_{17}	$f_{17}(X) = \left(x_2 - \frac{5.1}{4\pi^2} x_1 + \frac{5}{\pi} x_1 - 6 \right)^2 + 10 \left(1 - \frac{1}{8\pi} \right) cosx_1 + 10$	2	0.398	$(-5, 10) \& (0, 15)$
f_{18}	$f_{18}(X) = \left[1 + (x_1 + x_2 + 1)^2 (19 - 14x_1 + 3x_1^2 - 14x_2 + 6x_1 x_2 + 3x_2^2) \right]$	2	3	$(-2, 2)$
	$\times \left[30 + (2x_1 - 3x_2)^2 \times (18 - 32x_1 + 12x_1^2 + 48x_2 - 36x_1 x_2 + 27x_2^2) \right]$			
f_{19}	$f_{19}(X) = -\sum_{i=1}^{4} exp \left[-\sum_{j=1}^{3} a_{ij} (x_j - p_{ij})^2 \right]$	3	-3.86	$(0, 1)$
f_{20}	$f_{20}(X) = -\sum_{i=1}^{4} exp \left[-\sum_{j=1}^{6} a_{ij} (x_j - p_{ij})^2 \right]$	6	-3.32	$(0, 1)$
f_{21}	$f_{21}(X) = -\sum_{i=1}^{5} \left[(x - a_i)(x - a_i)^T + c_i \right]$	4	-10.1532	$(0, 10)$
f_{22}	$f_{22}(X) = -\sum_{i=1}^{7} \left[(x - a_i)(x - a_i)^T + c_i \right]$	4	-10.4028	$(0, 10)$
f_{23}	$f_{23}(X) = -\sum_{i=1}^{10} \left[(x - a_i)(x - a_i)^T + c_i \right]$	4	-10.5363	$(0, 10)$

Table A.10: CEC 2014 benchmark functions (summary)

Type	Function ID	Function	Global minima
Unimodal	f_{101}	Rotated high conditioned elliptic function	100
	f_{102}	Rotated bent cigar function	200
	f_{103}	Rotated discus function	300
Multimodal	f_{104}	Shifted and rotated Rosenbrock function	400
	f_{105}	Shifted and rotated Ackley's function	500
	f_{106}	Shifted and rotated Weierstrass function	600
	f_{107}	Shifted and rotated Griewank's function	700
	f_{108}	Shifted Rastrigin function	800
	f_{109}	Shifted and rotated Rastrigin function	900
	f_{110}	Shifted Schwefel's function	1000
	f_{111}	Shifted and rotated Schwefel's function	1100
	f_{112}	Shifted and rotated Katsuura function	1200
	f_{113}	Shifted and rotated HappyCat function	1300
	f_{114}	Shifted and rotated HGbat function	1400
	f_{115}	Shifted and rotated Expanded Griewank's plus Rosenbrock's function	1500
	f_{116}	Shifted and rotated Expanded Scaffer's F6 function	1600
Hybrid	f_{117}	Hybrid function 1 (f_{109}, f_{108}, f_{101})	1700
	f_{118}	Hybrid function 2 (f_{102}, f_{112}, f_{108})	1800
	f_{119}	Hybrid function 3 (f_{107}, f_{106}, f_{104}, f_{114})	1900
	f_{120}	Hybrid function 4 (f_{112}, f_{103}, f_{113}, f_{108})	2000
	f_{121}	Hybrid function 5 (f_{114}, f_{112}, f_{104}, f_{109}, f_{101})	2100
	f_{122}	Hybrid function 6 (f_{110}, f_{111}, f_{113}, f_{109}, f_{105})	2200
Composition	f_{123}	Composition function 1 (f_{104}, f_{101}, f_{102}, f_{103}, f_{101})	2300
	f_{124}	Composition function 2 (f_{110}, f_{109}, f_{114})	2400
	f_{124}	Composition function 3 (f_{111}, f_{109}, f_{101})	2500
	f_{126}	Composition function 4 (f_{111}, f_{113}, f_{1}, f_{106}, f_{107})	2600
	f_{127}	Composition function 5 (f_{114}, f_{109}, f_{111}, f_{106}, f_{101})	2700
	f_{128}	Composition function 6 (f_{115}, f_{113}, f_{111}, f_{116}, f_{101})	2800
	f_{129}	Composition function 7 (f_{117}, f_{118}, f_{119})	2900
	f_{130}	Composition function 8 (f_{120}, f_{121}, f_{122})	3000