# MSMA: Merged Slime Mould Algorithm for Solving Engineering Design Problems

Khaled Mohammad Alhashash[1], Hussein Samma[2], Shahrel Azmin Suandi[3*]

Intelligent Biometric Group, School of Electrical and Electronic Engineering, USM Engineering Campus, Universiti Sains Malaysia, Nibong Tebal 14300, Penang, Malaysia[1, 3]

SDAIA-KFUPM Joint Research Center for Artificial Intelligence (JRCAI), King Fahd University of Petroleum and Minerals, Dhahran 31261, Saudi Arabia[2]

***Abstract*—The Slime Mould Algorithm (SMA) has effectively solved various real-world problems such as image segmentation, solar photovoltaic cell parameter estimation, and economic emission dispatch. However, SMA and its variants still face limitations when dealing with low-dimensional optimization problems, including slow convergence and local optima traps. This study aims to develop an optimized algorithm, the Merged Slime Mould Algorithm (MSMA), to overcome these limitations and improve performance in low-dimensional optimization tasks. Additionally, MSMA introduces a novel approach by merging the Adaptive Opposition Slime Mould Algorithm (AOSMA) and the Smart Switching Slime Mould Algorithm (S2SMA), simplifying the hybridization process and enhancing optimization performance. MSMA eliminates the need for multiple initializations, avoids memory-switching requirements, and employs adaptive and smart switching rules to harness the strengths of both algorithms. The performance of MSMA is evaluated using the CEC 2005 benchmark and ten real-world applications. The Wilcoxon rank-sum test verifies the effectiveness of the proposed approach, with results compared to various SMA variations and related optimization methods. Numerical findings demonstrate superior fitness values achieved by the proposed strategy, while statistical results indicate MSMA's outperformance with a rapid convergence curve.**

***Keywords*—Slime mould algorithm; engineering design problems; metaheuristic; optimization***

## I. Introduction

Metaheuristic algorithms (MAs) offer valuable tools for solving complex engineering problems in a reasonable time [1]. These algorithms provide a flexible and efficient approach to optimization, enabling engineers to find near-optimal solutions in diverse domains. MAs have two main elements: exploration and exploitation abilities [2]. Exploration capability is the ability to converge to a possible global optimum with increasing solution space and randomness. On the other hand, exploitation capability refers to the ability to search more precisely in the region that the algorithm's exploration phase has identified. There are two categories of metaheuristics: population-based and single-solution-based metaheuristics [3]. Population-based approaches involve utilizing a collection of solutions, referred to as a population, to generate and substitute candidate solutions throughout the optimization procedure. Some of the popular population-based metaheuristic approaches are Particle Swarm Optimization (PSO) [4], whale optimization algorithm (WOA) [5], and Harris Hawk Optimizer (HHO) [6]. In contrast, metaheuristics that rely on a single solution-based approach involve generating a set of potential solutions derived from the current solution. Subsequently, the current solution is substituted with one of the candidate solutions during each iteration. This category involves the local search (LS)[7], Tabu search (TS)[8], and simulated annealing (SA) [9].

Single-based and population-based algorithms have benefits and are widely utilized to address various issues. Nevertheless, no single approach can solve all optimization problems [10]. Developing an optimization algorithm to address these issues is necessary, but researchers have found it challenging to design new optimization algorithms from scratch. In this direction, hybridizing meta-heuristic algorithms is the most common and successful technique. For example, on hybridizing meta-heuristic algorithms [11]–[13].

In the literature, several optimizers have emerged recently, such as SMA [14], Fitness Dependent Optimizer (FDO) [15], Black Widow Optimization Algorithm (BWO) [16], and Reptile Search Algorithm (RSA) [17]. SMA has captured considerable attention due to its smooth structure, limited parameter requirements, robustness, and flexibility in implementation. It presents itself as a valuable and efficient approach for addressing a wide range of real-world optimization problems [18], such as image segmentation [19], estimation of solar photovoltaic cell parameters [20], and economic emission dispatch [21]. Nevertheless, similar to other metaheuristic algorithms, SMA encounters challenges related to local optimality and premature convergence in some optimization problems [22], [23]. Moreover, Utilizing two random search agents from the entire population to determine the future displacement and direction based on the best search agents restricts SMA's exploitation and exploration capabilities [24]. Researchers suggested hybridized and modified variants of SMA to address these limitations.

This research article presents the hybridization of two variants of SMA, namely $S^2$SMA [25] and AOSMA [24]. The integration involves incorporating a set of vertical smart switching rules to govern the transition process between AOSMA and S2SM. The two algorithms were combined intelligently, where the invocation procedure exclusively occurs during the update of slime locations. This merger is unique and distinct from SAM's previous integration due to the following three advantages: no necessity for multiple initializations for different algorithms, no memory-switching needs, and employing adaptive and intelligent switching rules to leverage the strengths of both algorithms. The main contributions of this work are outlined as follows:

- MSMA introduces a novel optimization approach by intelligently merging AOSMA and $S^2$SMA, setting it apart from previous SMA integrations through its streamlined operational framework, which simplifies the algorithm hybridization process.

- The MSMA eliminates the necessity for multiple initializations and memory-switching, significantly enhancing computational efficiency. This innovation reduces the algorithm's complexity and resource consumption, facilitating a more seamless optimization experience.

- Incorporating Vertical Smart Switching Rules (VSSR) enables MSMA to facilitate dynamic algorithmic switches based on problem-specific attributes, amplifying adaptability and operational efficiency. VSSR represents a critical innovation, ensuring effective navigation through complex problem spaces and significantly improving optimization performance.

- The MSMA has been rigorously validated through extensive experiments and numerical studies, demonstrating its superiority in solving optimization problems.

This paper's remaining sections are organized as follows. The pertinent studies on SMA and engineering design problems are summarized in Section II. Section III illustrates the slime mould algorithm and the proposed work in detail. Section IV presents the numerical experiment and statistical analysis. This paper's conclusion is given in Section V.

## II. RELATED WORK

As mentioned previously, SMA can be categorized into hybridized and modified forms of SMA. Many studies have investigated the idea of hybridizing SMA with other metaheuristic algorithms [26]–[30]. Among these advancements, Naik et al. [26] introduced the Equilibrium Slime Mould Algorithm (ESMA), merging the Slime Mould Algorithm (SMA) with the Equilibrium Optimizer (EO) for enhanced multilevel thresholding in breast thermogram images. ESMA aims to reduce entropic dependencies between image classes, showing improved exploration capability and efficient analysis over other optimization methods. Although it outperforms in breast thermogram analysis, suggesting potential benefits for medical diagnostics, ESMA faces challenges in specific clinical contexts and broader medical imaging applications. Further contributing to the field, Chen et al. [27] introduced CHDESMA, an improved Slime Mould Algorithm (SMA) using chaotic maps and Differential Evolution (DE). CHDESMA mitigates SMA's local optima and population diversity issues by integrating chaotic maps for initialization and DE strategies for enhanced search. Evaluations against benchmarks and real-world problems show CHDESMA's competitive performance against advanced algorithms and DE variants, emphasizing its effectiveness and contributions in diverse scenarios. Moreover, Bhandakkar and Mathew [28] proposed using Integrated Slime Mould Algorithm (ISMA) for optimal placement of a Hybrid Power Flow Controller (HPFC). ISMA combines the Slime Mould Algorithm (SMA) with WOA for enhanced searching behavior. This optimization aims to minimize system power loss and generation cost by determining optimal locations for Unified Power Flow Controllers (UPFCs) and their capacities while considering system stability constraints. Chen et al. [29] presented RCLSMAOA, merging SMA and AOA to improve optimization. Through extensive testing, it effectively combines global exploration and local exploitation strategies. Despite the success, challenges persist in high-dimensional spaces and convergence accuracy. Future work aims to refine RCLSMAOA's performance in practical engineering problems and high dimensions, potentially exploring a binary version of the algorithm for further enhancement. Finally, Ewees et al. [30] introduced GBOSMA, a hybrid method merging Gradient-Based Optimizer (GBO) and Slime Mould Algorithm (SMA) to improve global optimization and feature selection. GBOSMA enhances exploration by using SMA as a local search within GBO, achieving better performance than standard GBO, SMA, and recent algorithms in both speed and accuracy across diverse benchmarks. The results showcase GBOSMA's superiority, achieving top fitness values in 66% of global optimization functions and the highest accuracy in 93% of feature selection benchmarks. This approach holds potential for various applications like medical imaging, object detection, and weather prediction tasks.

In many investigations, modified SMA methods were presented [24], [25], [31]–[34]. The Adaptive Opposition Slime Mould Algorithm (AOSMA), as introduced by Naik et al. [24], represents an advancement in the Slime Mould Algorithm (SMA) through the integration of adaptive opposition-based learning. This enhancement significantly boosts the algorithm's exploration and exploitation capabilities, making it a powerful tool for solving complex problems. However, AOSMA is not without its limitations. It shows a marked reliance on specific problem types, indicating that its effectiveness may be constrained to particular domains. Additionally, there is a noted requirement for further validation to confirm its broader applicability across a wider range of problem scenarios. Alhashash et al. [25] introduced an enhanced optimizer named Smart Switching Slime Mould Algorithm ($S^2$SMA) that enhances the accuracy of face sketch recognition by fine-tuning pre-trained deep learning models, which is challenging due to limited sketch datasets. $S^2$SMA simultaneously fine-tunes multiple deep learning models and uses embedded rules and search operations for adaptive switching between search operations during execution. The proposed algorithm was evaluated on CEC's 2010 large-scale benchmark and two face sketch databases and outperformed other optimization techniques with a faster convergence rate. The outcomes revealed the superiority of $S^2$SMA in the majority of experiments. Ewees et al. [31] presented a modified version of the slime mould algorithm (SMA) called SMAMPA, which incorporates the Marine Predators Algorithm (MPA) operators as a local search strategy. The proposed feature selection technique was evaluated on twenty UCI datasets and compared with other state-of-the-art FS methods, showing superior performance in terms of efficiency and performance metrics. The SMAMPA method was also applied to real-world problems, such as QSAR modeling and chemometrics, with promising results. Future work includes investigating SMAMPA in more complicated problems, such as multi-optimization problems and big data mining. Abid et al. [32] proposed an enhanced slime mould optimization algorithm (ESMOA) to optimize tuning parameters for a cascaded proportional derivative-proportional integral (PD-PI) controller in order to solve frequency stability problems (FSP) in multi-area power systems (MAPSs) with two-area non-reheat thermal systems. ESMOA surpassed current PID and PI controllers. Cascaded PD-PI controller designs are more reliable than GSO and CO algorithms due to ESMOA's chaotic dynamic and elite group. In time domain simulations, ESMOA beat both GSO and CO. Deng and Liu [33] proposed AGSMA, an improved variant of the slime mould algorithm, to address limitations such as insufficient exploration, slow convergence, and an imbalance between diversity and convergence. AGSMA achieved a balance between convergence and diversity through adaptive grouping, a new search mechanism, and an efficient learning operator. Experiments demonstrated that it outperformed other methods and is able to solve complex nonlinear problems. However, premature convergence in some multimodal problems needs additional study. Sharma et al. [34] presented modifications to the Slime Mould Algorithm (SMA) to make it more effective for engineering design tasks, including opposition theory and a sine cosine-based position update mechanism. These modifications were found to significantly enhance the performance of SMA on standard benchmark functions and make it suitable for demand-side management tasks.

In the evolving field of metaheuristic algorithms, recent studies have made significant strides in addressing complex engineering design problems. Samma et al. [13] pioneered the Q-learning-based Simulated Annealing (QLSA) algorithm, setting a precedent for dynamic parameter control and adaptability, albeit with scalability and exploration scope limitations. Building on this, Nadimi-Shahraki et al. [1] introduced the Gaze Cues Learning-based Grey Wolf Optimizer (GGWO), which incorporated novel search strategies inspired by wolf behavior, showing promise despite challenges in selective pressure optimization. Further contributions, such as Wang et al. [35]'s Artificial Rabbits Optimization (ARO) and Yildiz et al. [36]'s Elite Opposition-Based Learning Grasshopper Optimization (EOBL-GOA), demonstrated the algorithms' strengths in diverse engineering problems but also highlighted the need for domain-specific adaptability. Zhang et al. [37] and Yıldız et al. [38] proposed enhancements to the Slime Mould Algorithm (SMA) and introduced the Chaotic Lévy flight distribution (CLFD) algorithm, respectively, achieving improved solution quality and exploration-exploitation balance. Recent developments saw Yang et al. [39] focus on the ARSCA algorithm, addressing computational complexity while improving convergence accuracy. Abdel-Basset et al. [40] applied the Nutcracker Optimization Algorithm (NOA) to engineering problems, demonstrating the ease of implementation and high convergence speed but facing challenges in exploration-exploitation balance. Gharehchopogh et al. [41] introduced the Chaotic Quasi-oppositional Farmland Fertility Algorithm (CQFFA), which enhanced exploration and convergence via chaotic maps and the Quasi-Oppositional Binary Leader strategy, albeit with hybridization challenges. Deng and Liu [42] showcased the Multi-strategy Improved Slime Mould Algorithm (MSMA), signaling a need for enhancements in multi-objective optimization and broader domain adaptability.

Despite significant advancements in developing SMA variants, current methods still face challenges in broader applicability and often struggle with slow convergence and local optima traps in low-dimensional optimization problems. This gap highlights the need for improved solutions that can overcome these limitations. The proposed approach addresses these challenges by integrating SMA variants with adaptive mechanisms, enhancing computational efficiency, reducing the reliance on multiple initializations, and simplifying the hybridization process, offering a more robust and effective solution for complex optimization tasks.

## III. Proposed SMA-Based Method

### A. The original Slime Mould Algorithm (SMA)

Li et al. [14] introduced the SMA as an innovative optimization mechanism for global optimization. SMA focuses on the behavior and morphological changes that the slime mould Physarum polycephalum undergoes during nutrient acquisition. Approaching, wrapping, and grabbing food are the three stages of SMA.

*1) Approaching food:* The concentration of odor in the air is essential for a slime mould to approach food. This contraction pattern when nearing food is defined by Eq. (1):

$$\overrightarrow{X(t+1)} = \begin{cases} \overrightarrow{X_b(t)} + \overrightarrow{vb} \cdot \left(\overrightarrow{W} \cdot \overrightarrow{X_A(t)} - \overrightarrow{X_B(t)}\right), r_2 < p \\ \overrightarrow{vc} \cdot \overrightarrow{X(t)}, r_2 \geq p \end{cases} \quad (1)$$

where the parameter $\overrightarrow{vb}$ takes values within the range of [-a, a], while $\overrightarrow{vc}$ gradually decreases from one to zero in a linearly. The position $\overrightarrow{X_b}$ refers to the current location of an individual with the highest concentration of odor detected. $\vec{X}$ represents the current location of slime mould. $\overrightarrow{X_A}$ and $\overrightarrow{X_B}$ denote two randomly selected individuals from a population of size n. The variables $t$ and $r_2$ indicate the current iteration number and a random value between 0 and 1, respectively. The weight of slime mould is represented by $\overrightarrow{W}$. The parameter p is computed using Eq. (2):

$$p = tanh|S(i) - DF| \quad (2)$$

where $i \in 1,2, \ldots, n$. The fitness of the current location $\vec{X}$ is denoted by $S(i)$, while DF denotes the best fitness achieved across all iterations. The formula for computing $\overrightarrow{vb}$ can be found in Eq. (3), and the value of $a$ is provided in Eq. (4).

$$\overrightarrow{vb} = [-a, a]) \quad (3)$$

$$a = arctanh(-\left(\frac{t}{max\_t}\right) + 1) \quad (4)$$

Here, $max\_t$ refers to the maximum number of iterations. The formula for calculating $\overrightarrow{W}$ is presented in Eq. (5), while its $SmellIndex$ is defined in Eq. (6).

$$\overrightarrow{W(SmellIndex(\iota))} = \begin{cases} 1 + r_3 \cdot log\left(\frac{bF-S(i)}{bF-wF}+1\right), condition \\ 1 - r_3 \cdot log\left(\frac{bF-S(i)}{bF-wF}+1\right),\ others \end{cases} \tag{5}$$

$$SmellIndex = sort(S) \tag{6}$$

where, condition represents that S(i) must be ranked within the top fifty percent of the entire population. The variable $r_3$ denotes a random value ranging from 0 to 1. The best fitness value achieved during the current iteration process is represented as $bF$, while the worst fitness value is denoted as $wF$. SmellIndex corresponds to the sequence of fitness values arranged in ascending order.

*2) Wrapping food:* Updates to the position of slime mould can be calculated using the formula given in Eq. (7):

$$\overrightarrow{X^*(t+1)} = \begin{cases} r_4 \cdot (UB - LB) + LB, r_1 < z \\ \overrightarrow{X_b(t)} + \overrightarrow{vb} \cdot \left(W \cdot \overrightarrow{X_A(t)} - \overrightarrow{X_B(t)}\right), r_2 < p \ \ and\ r_1 \geq z \\ \overrightarrow{vc} \cdot \overrightarrow{X(t)}, r_2 \geq p \ \ and\ r_1 \geq z \end{cases} \tag{7}$$

where, $r_1$, $r_2$ and $r_4$ are randomly selected from the interval [0,1]. $B$ and $UB$ represent the lower and upper bounds of the search range, respectively. The $p$ value signifies the probability associated with the presence of slime mould, while $z$ is a parameter with a constant value of 0.03.

*3) Grabbling food:* To represent the slime mould venous width changes, SMA utilizes the vectors $\overrightarrow{W}$, $\overrightarrow{vb}$, and $\overrightarrow{vc}$. $\overrightarrow{W}$ reflects the oscillating frequency of slime mould, which is determined by analyzing the quality of the food source. It helps update the speed of movement towards the food source, aiding the slime mould in selecting the most suitable food source.

The values of $\overrightarrow{vb}$ and $\overrightarrow{vc}$ undergo random oscillations within specific ranges. The vector $\overrightarrow{vb}$ ranges from -$a$ to $a$, while $\overrightarrow{vc}$ ranges from -1 to 1. As the iterative process progresses, these vectors converge towards zero.

The variation in $\overrightarrow{vb}$ replicates the slime mould's behaviour when it encounters a new food source. Even if an improved food supply has been identified, the slime mould continues to explore other locations by separating some organic matter. This behaviour increases the chances of finding higher-quality food sources and improves the optimization of local problems. For further details on the SMA, refer to the study conducted by Li et al. [14].

## *B. The Proposed Merged Slime Mould Algorithm*

MSMA is a novel optimization method that combines two variants SMA: AOSMA and $S^2$SMA. This merger distinguishes itself from previous integrations by providing three primary advantages: the elimination of the necessity for multiple initializations for different algorithms, avoidance of memory-switching requirements, and the incorporation of adaptive and intelligent switching rules, known as the Vertical Smart Switching Rules (VSSR).

The formulation of VSSR involves incorporating four embedded vertical smart switching rules to control the recall ratio between AOSMA and $S^2$SMA during slime position updates. The activation of VSSR is dependent on the occurrence of specific events, comprising seven parameters: $AOSMA\_EN$ parameter, $S^2SMA\_EN$ parameter, $EVAL\_C$ parameter, $AOSMA\_C$ parameter, $S^2SMA\_C$ parameter, $VER\_SUCCESS_{LEADER}$ parameter, and $per$. The first parameter is $AOSMA\_EN$. It will take either zero or one. If AOSMA is chosen to update slime positions, it will be one; otherwise, it will be zero. The second parameter is $S^2SMA\_EN$. It will take either zero or one. If $S^2$SMA is chosen to update slime positions, it will be one; otherwise, it will be zero. The third parameter, $EVAL\_C$, evaluation counter represents the number of iterations required to evaluate the performance of the two algorithms and is computed using Eq. (8). The fourth and fifth parameters, $AOSMA\_C$ and $S^2SMA\_C$, respectively, count the number of times each algorithm successfully finds a new leader within $EVAL\_P$ iterations when used to update slime positions. The sixth parameter, $VER\_SUCCESS_{LEADER}$, assumes one value if any methods can find a new leader and zero otherwise. The final parameter, $per$, takes on a value within the range of [0,1], and its value depends on the rules to be applied, which will be further expounded in the ensuing section.

$$EVAL\_P = \epsilon * max\_t \tag{8}$$

where, $\epsilon$ is a constant parameter of 0.02, its value is affordable, which was selected to ensure timely switching. However, increasing this value would result in slower switching, perhaps introducing bias. Conversely, decreasing the value would lead to faster switching, hence increasing complexity. Moreover, this parameter is adjustable based on the nature of a given problem.

The first and second rules are depicted in Fig. 1 and Fig. 2 respectively. They were developed to update the value of $AOSMA\_C$ and $S^2SMA\_C$, which indicates the number of successes for each approach during the process of finding a new leader. Both rules will be checked in every iteration. The first rule will apply if AOSMA is called while updating the slime position and a new leader is found. Thus, $AOSMA\_C$ will be updated. The second rule will apply if $S^2$SMA is called while updating the slime position and a new leader is found. Thus, $S^2SMA\_C$ will be updated. It should be noted that the proposed method will give AOSMA and $S^2$SMA equal priority to change slime positions during the first $EVAL\_P$. During the process, the values of both $AOSMA\_C$ and $S^2SMA\_C$ will be updated as explained in Rule1 and Rule2.

**RULE 1: IF** $AOSMA\_EN == 1$ **AND** $VER\_SUCCESS_{LEADER} == 1$
**THEN** $AOSMA\_C = AOSMA\_C + 1$

Fig. 1. RULE 1 To count the number of times AOSMA was successful during a given period.

**RULE 2: IF** $S^2SMA\_EN == 1$ **AND** $VER\_SUCCESS_{LEADER} == 1$
**THEN** $S^2SMA\_C = S^2SMA\_C + 1$

Fig. 2. RULE 2 To count the number of times $S^2$SMA was successful during a given period.

The third and fourth rules are shown in Fig. 3 and Fig. 4, respectively. They are formulated to calculate $per$, which is the ratio of AOSMA and S²SMA calling to update slime positions during the subsequent $EVAL\_P$. This value depends on the values of $AOSMA\_C$ and $S^2SMA\_C$, as explained in Rule1 and Rule2. If AOSMA and S²SMA cannot find a new leader during the current $EVAL\_P$, both methods will be given an equal chance over the subsequent $EVAL\_P$; otherwise, the third and fourth rules will be applied. The third rule is applied if the value of $AOSMA\_C$ is greater than $S^2SMA\_C$; otherwise, the fourth rule will be applied.

$$\textbf{RULE 3: IF } AOSMA\_C > S^2SMA\_C$$
$$\textbf{THEN } per = AOSMA\_C/(AOSMA\_C + S^2SMA\_C)$$

Fig. 3. RULE 3 to compute the probability of AOSMA being called within the specified period.

$$\textbf{RULE 4: IF } AOSMA\_C \leq S^2SMA\_C$$
$$\textbf{THEN } per = 1 - (S^2SMA/(AOSMA\_C + S^2SMA\_C))$$

Fig. 4. RULE 4 to compute the probability of S²SMA being called within the specified period.

In the SMA algorithm, the arctanh function is utilized to calculate the value of parameter $a$ in Eq. (4). However, it has been observed that the arctanh function can lead to programming warnings/errors [43]-[45]. To enhance the performance and stability of the standard SMA algorithm and achieve faster convergence with reduced warnings/errors during program execution, alternative controlling equations, such as the cosine function, have been proposed as viable solutions [44]. In this study, the value of parameter $a$ was computed using Eq. (9), which was directly obtained from [45].

$$a = 1 + cos\,(\frac{t}{max\_t} \cdot \pi) \tag{9}$$

where $max\_t$ is the maximum number of iterations and t is the current iteration.

Fig. 5 depicts the complete stages of the proposed MSMA algorithm.

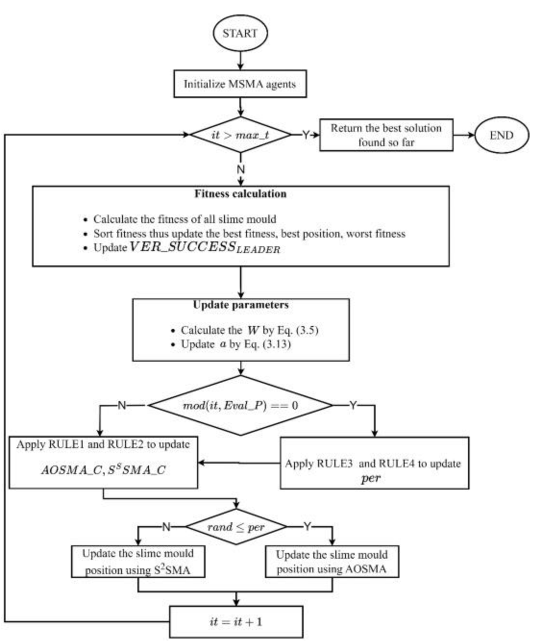



Fig. 5. Flow chart of the proposed MSMA algorithm.

## IV. EXPERIMENTAL RESULTS AND ANALYSIS

In this analysis section, several experiments were conducted to demonstrate MSMA's efficacy. Three case studies were investigated, comprising basic benchmark problems CEC 2005 [46] and seven engineering designs.

### A. Evaluation on Basic Benchmark Functions

In this section, a total of 23 CEC 2005 [46] continuous benchmarks were used, categorized into seven unimodal (F1-F7), six multimodal (F8-F13), and ten fixed-dimensional multimodal functions (F14-F23), as illustrated in Table I, Table II, and Table III. Unimodal functions assess exploitation efficiency with one global optimum, while multimodal functions (F8-F13) evaluate exploration and local optima avoidance. Fixed-dimensional tests (F14-F23) provide a middle ground with fewer local optima, gauging the algorithm's balance between exploitation and exploration.

TABLE I. DESCRIPTION OF UNIMODAL BENCHMARK FUNCTIONS

| Function | Description | Dim | Range | $f_{min}$ |
|---|---|---|---|---|
| $F_1(X) = \sum_{j=1}^{D} x_j^{\,2}$ | Sphere | 30 | [-100,100] | 0 |
| $F_2(X) = \sum_{j=0}^{D} \lvert x_j \rvert + \prod_{j=0}^{D} \lvert x_j \rvert$ | Schwefel 2.22 | 30 | [-10,10] | 0 |
| $F_3(X) = \sum_{j=1}^{D} (\sum_{k=1}^{j} x_k)^2$ | Schwefel 1.2 | 30 | [-100,100] | 0 |
| $F_4(X) = max_j\{\lvert x_j \rvert, 1 \leq j \leq D\}$ | Schwefel 2.21 | 30 | [-100,100] | 0 |
| $F_5(X) = \sum_{j=1}^{D-1} [\,100(x_{j+1} - x_j^{\,2})^2 + (x_j - 1)^2]$ | Rosenbrock | 30 | [-30,30] | 0 |
| $F_6(X) = \sum_{j=1}^{D} ([x_j + 0.5])^2$ | Step | 30 | [-100,100] | 0 |
| $F_7(X) = \sum_{j=0}^{D} j x_j^{\,4} + random[0,1]$ | Quartic | 30 | [-128,128] | 0 |

TABLE II. DESCRIPTION OF MULTIMODAL BENCHMARK FUNCTIONS

| Function | Description | Dim | Range | $f_{min}$ |
|---|---|---|---|---|
| $F_8(X) = \sum_{j=1}^{D} -x_j \sin(\sqrt{\lvert x_j \rvert})$ | Schwefel | 30 | [-500,500] | −418.98 29 ∗n |
| $F_9(X) = \sum_{j=1}^{D} [x_j^{\,2} - 10\cos(2\pi x_j) + 10]$ | Rastrigin | 30 | [−5.12,5.12] | 0 |
| $F_{10}(X) = -20\exp\left(-0.2\sqrt{\frac{1}{D}\sum_{j=1}^{D} x_j^{\,2}}\right) - \exp\left(\frac{1}{D}\sum_{j=1}^{D}\cos(2\pi x_j)\right) + 20 + e$ | Ackley | 30 | [-32,32] | 0 |

| Function | Description | Dim | Range | $f_{min}$ |
|---|---|---|---|---|
| $F_{11}(X) = \frac{1}{4000}\sum_{j=1}^{D} {x_j}^2 - \prod_{j=1}^{D} \cos\left(\frac{x_j}{\sqrt{j}}\right) + 1$ | Griewank | 30 | [-600,600] | 0 |
| $F_{12}(X) = \frac{\pi}{D}\left\{10\sin(\pi y_1) + \sum_{j=1}^{D-1}(y_j-1)^2\left[1 + 10 sin^2(\pi y_{j+1})\right] + (y_D-1)^2\right\} + \sum_{j=1}^{D} u(x_j,10,100,4)$; $y_j = 1 + \frac{x_j+1}{4}$; $u(x_j,a,k,m) = \begin{cases} k(x_j-a)^m & x_j > \\ 0 & -a < x_j \\ k(-x_j-a)^m & x_j < \end{cases}$ | Penalized | 30 | [-50,50] | 0 |
| $F_{13}(X) = 0.1\left\{sin^2(3\pi x_1) + \sum_{j=1}^{D}(x_j-1)^2\left[1 + sin^2(3\pi x_j + 1)\right] + (x_D-1)^2[1 + sin^2(2\pi x_D)]\right\} + \sum_{j=1}^{D} u(x_j,5,100,4)$ | Penalize 2 | 30 | [-50,50] | 0 |

TABLE III. DESCRIPTION OF FIXED-DIMENSION MULTIMODAL BENCHMARK FUNCTIONS

| Function | Description | Dim | Range | $f_{min}$ |
|---|---|---|---|---|
| $F_{14}(X) = \left(\frac{1}{500} + \sum_{j=1}^{25}\frac{1}{j + \sum_{k=1}^{2}(x_k - a_{kj})^6}\right)^{-1}$ | Foxholes | 2 | [-65,65] | 1 |
| $F_{15}(X) = \sum_{j=1}^{11}\left[a_j - \frac{x_1(b_j^2 - b_j x_2)}{b_j^2 + b_j x_3 + x_4}\right]^2$ | Kowalik | 4 | [−5,5] | 0.0003 |
| $F_{16}(X) = 4x_1^2 - 2.1x_1^2 + \frac{1}{3}x_1^6 + x_1x_2 - 4x_2^2 + 4x_2^4$ | Six-hump Camel-Back | 2 | [-5,5] | -1.0316 |
| $F_{17}(X) = \left(x_2 - \frac{5.1}{4\pi^2}x_1^2 + \frac{5}{\pi}x_1 - 6\right)^2 + 10\left(1 - \frac{1}{8\pi}\right)cos\, x_1 + 10$ | Branin | 2 | [-5,5] | 0.398 |
| $F_{18}(X) = [1 + (x_1 + x_2 + 1)^2(19 - 14x_1 + 3x_1^2 - 14x_2 + 6x_1x_2 + 3x_2^2)] * [30 + (2x_1 - 3x_2)^2(18 - 32x_1 + 12x_1^2 + 48x_2 - 36x_1x_2 + 27x_2^2)]$ | Goldstein-Price | 2 | [-2,2] | 3 |
| $F_{19}(X) = -\sum_{j=1}^{4} c_j \exp(-\sum_{k=1}^{3} a_{jk}(x_k - p_{jk})^2)$ | Hartman 3 | 3 | [1,3] | -3.86 |
| $F_{20}(X) = -\sum_{j=1}^{4} c_j \exp(-\sum_{k=1}^{6} a_{jk}(x_k - p_{jk})^2)$ | Hartman 6 | 6 | [0,1] | -3.32 |
| $F_{21}(X) = -\sum_{j=1}^{5}[(X - a_j)(X - a_j)^T + c_j]^{-1}$ | Shekel 5 | 4 | [0,10] | -10.1532 |
| $F_{22}(X) = -\sum_{j=1}^{7}[(X - a_j)(X - a_j)^T + c_j]^{-1}$ | Shekel 7 | 4 | [0,10] | -10.4028 |
| $F_{23}(X) = -\sum_{j=1}^{10}[(X - a_j)(X - a_j)^T + c_j]^{-1}$ | Shekel 10 | 4 | [0,10] | -10.5363 |

*1) Comparison with SMA and SMA variants:*

*a) Performance analysis:* This section compares the efficacy of MSMA to that of SMA [14] and SMA variants. Specifically, $S^2$SMA [25], ESMA [26], LSMA [19], and AOSMA [24] are executed based on the parameters shown in Table IV. The mean, and standard deviation of MSMA and other algorithms are reported in Table V. The ranking was determined by using an average of 30 runs. MSMA achieved the optimal value, zero, or the best result in most functions relative to other algorithms. This is due to the application of rules that aid in selecting the optimal algorithm, which in turn enables the achievement of optimal results. However, the outcomes were not satisfactory due to the nature of the functions F5, F6, F7, F19, and F20.

TABLE IV. CONFIGURATION PARAMETERS FOR THE EXAMINED ALGORITHMS

| Method | Population size | The maximum number of iterations | Other parameters |
|---|---|---|---|
| ***MSMA (Proposed)*** | 30 | $10^3$ | $z$ = 0.03, $\mu$ =0.5, ϵ =0.02, α = 5 and $\beta$ =3/2 |
| ***SMA*** [14] | 30 | $10^3$ | $z$ = 0.03 |
| ***LSMA*** [19] | 30 | $10^3$ | $z$ = 0.03 |

| | | | |
|---|---|---|---|
| ***AOSMA*** [24] | 30 | $10^3$ | $z = 0.03$ |
| ***ESMA*** [26] | 30 | $10^3$ | $z = 0.03$ |

TABLE V. RESULTS OF CEC 2005 FUNCTIONS

| Function | Fitness | Algorithm | | | | | |
|---|---|---|---|---|---|---|---|
| | | ***MSMA*** | ***S²SMA*** | ***SMA*** | ***ESMA*** | ***LSMA*** | ***AOSMA*** |
| ***F1*** | Mean | 0.000E+00 | 6.525E-06 | 0.000E+00 | 0.000E+00 | 0.000E+00 | 0.000E+00 |
| | Std | 0.000E+00 | 2.863E-06 | 0.000E+00 | 0.000E+00 | 0.000E+00 | 0.000E+00 |
| ***F2*** | Mean | 0.000E+00 | 2.250E-03 | 1.233E-196 | 2.988E-227 | 0.000E+00 | 0.000E+00 |
| | Std | 0.000E+00 | 8.600E-04 | 0.000E+00 | 0.000E+00 | 0.000E+00 | 0.000E+00 |
| ***F3*** | Mean | 0.000E+00 | 7.737E-04 | 0.000E+00 | 0.000E+00 | 0.000E+00 | 0.000E+00 |
| | Std | 0.000E+00 | 3.599E-04 | 0.000E+00 | 0.000E+00 | 0.000E+00 | 0.000E+00 |
| ***F4*** | Mean | 0.000E+00 | 9.313E-02 | 6.527E-201 | 2.532E-295 | 0.000E+00 | 0.000E+00 |
| | Std | 0.000E+00 | 7.777E-02 | 0.000E+00 | 0.000E+00 | 0.000E+00 | 0.000E+00 |
| ***F5*** | Mean | 3.129E-02 | 8.328E-01 | 1.174E+00 | 9.674E-01 | **1.866E-02** | 2.254E-02 |
| | Std | 1.165E-01 | 4.461E+00 | 4.931E+00 | 4.676E+00 | 1.451E-02 | 6.369E-02 |
| ***F6*** | Mean | 4.397E-06 | 3.152E-05 | 9.883E-04 | 5.087E-04 | 1.710E-04 | **3.987E-06** |
| | Std | 2.498E-06 | 1.238E-05 | 4.196E-04 | 2.147E-04 | 6.327E-05 | 1.589E-06 |
| ***F7*** | Mean | 4.950E-05 | 3.833E-03 | 7.968E-05 | 7.292E-05 | 5.945E-05 | **2.761E-05** |
| | Std | 6.893E-05 | 4.007E-03 | 7.901E-05 | 7.692E-05 | 6.367E-05 | 2.676E-05 |
| ***F8*** | Mean | **-1.257E+04** | -1.257E+04 | -1.257E+04 | -1.257E+04 | -1.257E+04 | -1.257E+04 |
| | Std | 1.563E-04 | 1.724E-04 | 8.373E-02 | 2.321E-02 | 4.497E-03 | 1.784E-04 |
| ***F9*** | Mean | 0.000E+00 | 3.574E-06 | 0.000E+00 | 0.000E+00 | 0.000E+00 | 0.000E+00 |
| | Std | 0.000E+00 | 1.111E-06 | 0.000E+00 | 0.000E+00 | 0.000E+00 | 0.000E+00 |
| ***F10*** | Mean | 8.882E-16 | 6.185E-04 | 8.882E-16 | 8.882E-16 | 8.882E-16 | 8.882E-16 |
| | Std | 0.000E+00 | 1.318E-04 | 0.000E+00 | 0.000E+00 | 0.000E+00 | 0.000E+00 |
| ***F11*** | Mean | 0.000E+00 | 5.694E-03 | 0.000E+00 | 0.000E+00 | 0.000E+00 | 0.000E+00 |
| | Std | 0.000E+00 | 1.168E-02 | 0.000E+00 | 0.000E+00 | 0.000E+00 | 0.000E+00 |
| ***F12*** | Mean | 1.193E-06 | **7.800E-07** | 8.589E-04 | 4.542E-04 | 3.119E-05 | 3.051E-05 |
| | Std | 7.749E-07 | 3.096E-07 | 1.335E-03 | 5.612E-04 | 1.612E-05 | 1.576E-04 |
| ***F13*** | Mean | **9.972E-06** | 1.106E-05 | 7.211E-04 | 4.259E-04 | 2.254E-04 | 1.110E-03 |
| | Std | 5.603E-06 | 4.735E-06 | 4.926E-04 | 2.342E-04 | 1.175E-04 | 3.352E-03 |
| ***F14*** | Mean | **9.980E-01** | 9.980E-01 | 9.980E-01 | 9.980E-01 | 9.980E-01 | 9.980E-01 |
| | Std | 1.526E-15 | 1.517E-15 | 1.508E-13 | 8.343E-14 | 2.859E-13 | 6.661E-14 |
| ***F15*** | Mean | **3.448E-04** | 4.613E-04 | 5.015E-04 | 4.679E-04 | 5.629E-04 | 5.246E-04 |
| | Std | 8.868E-05 | 3.506E-04 | 2.224E-04 | 1.964E-04 | 2.814E-04 | 3.570E-04 |
| ***F16*** | Mean | **-1.032E+00** | -1.032E+00 | -1.032E+00 | -1.032E+00 | -1.032E+00 | -1.032E+00 |
| | Std | 6.421E-13 | 2.593E-11 | 1.878E-10 | 3.863E-10 | 1.630E-09 | 5.656E-12 |
| ***F17*** | Mean | **3.979E-01** | 3.979E-01 | 3.979E-01 | 3.979E-01 | 3.979E-01 | 3.979E-01 |
| | Std | 2.171E-09 | 1.531E-07 | 3.122E-08 | 4.086E-08 | 7.897E-08 | 5.215E-09 |
| ***F18*** | Mean | 3.000E+00 | 3.000E+00 | 3.000E+00 | 3.000E+00 | 3.000E+00 | 3.000E+00 |
| | Std | 3.720E-10 | 1.187E-09 | 6.592E-12 | 9.030E-13 | 7.936E-08 | 7.377E-10 |
| ***F19*** | Mean | -3.863E+00 | -3.863E+00 | **-3.863E+00** | -3.863E+00 | -3.863E+00 | -3.863E+00 |
| | Std | 3.073E-06 | 1.824E-05 | 3.073E-08 | 8.330E-07 | 2.485E-06 | 9.483E-07 |
| ***F20*** | Mean | -3.254E+00 | -3.230E+00 | -3.239E+00 | -3.231E+00 | -3.237E+00 | **-3.270E+00** |
| | Std | 6.014E-02 | 5.138E-02 | 5.543E-02 | 5.133E-02 | 5.636E-02 | 6.038E-02 |
| ***F21*** | Mean | **-1.015E+01** | -1.015E+01 | -1.015E+01 | -1.015E+01 | -1.015E+01 | -1.015E+01 |
| | Std | 5.139E-07 | 9.000E-05 | 9.021E-05 | 1.180E-04 | 3.903E-05 | 5.340E-06 |
| ***F22*** | Mean | **-1.040E+01** | -1.040E+01 | -1.040E+01 | -1.040E+01 | -1.040E+01 | -1.040E+01 |
| | Std | 7.056E-07 | 7.015E-05 | 9.711E-05 | 1.015E-04 | 3.713E-05 | 6.514E-06 |
| ***F23*** | Mean | **-1.054E+01** | -1.054E+01 | -1.054E+01 | -1.054E+01 | -1.054E+01 | -1.054E+01 |
| | Std | 1.544E-07 | 7.791E-05 | 1.054E-04 | 1.122E-04 | 4.224E-05 | 6.612E-06 |

*b) Analysis of execution time:* The computer's software and hardware specifications used to conduct the investigations in this study are elaborated on in Table VI. Table VII displays the computational time (in seconds) for three different algorithms: MSMA, SMA, and AOSMA. According to Table VIII, the SMA algorithm achieved a computation time of 0.51318182 seconds, while the AOSMA algorithm recorded a shorter time at 0.260618 seconds. As a result of hybridizing AOSMA and S²SMA, MSMA achieved a computation time of 0.380505 seconds and thus outperformed the original SMA algorithm. These results indicate that MSMA shows promise in improving task-specific computational time compared to the traditional SMA approach. Notably, the programming language, programmer proficiency, and machine configuration influence the CPU time utilized by each method.

TABLE VI. SETTING INFORMATION FOR HARDWARE AND SOFTWARE

| Item | Component | Setting |
|---|---|---|
| ***Hardware*** | CPU | Intel(R) Core (TM) i7-10700 |
| | Frequency | 2.9 GHz |
| | RAM | 16GB |
| | GPU | Nvidia GeForce GTX 1660 Super |
| | SSD | 256 GB |
| | Hard Drive | 2 TB |
| ***Software*** | Operating system | Windows 10 |
| | Language | MATLAB R2021a |

TABLE VII. COMPUTATIONAL TIME ANALYSIS

| | MSMA (Proposed) | SMA | AOSMA |
|---|---|---|---|
| **Time (Second)** | 0.380505 | 0.51318182 | 0.260618 |

2) *Comparison with conventional algorithms:*

*a) Performance analysis:* This section compares the performance of the MSMA algorithm with six popular metaheuristic algorithms: WOA [5], Multi-Verse Optimizer (MVO) [47], Grey Wolf Optimizer (GWO) [48], Sine Cosine Algorithm (SCA) [49], Arithmetic Optimization Algorithm (AOA) [50], and PSO [4] across unimodal and multimodal functions (F1-F13). The primary parameter configurations of these algorithms are displayed in Table VIII below. It has been demonstrated that the MSMA variant outperforms the original SMA and other SMA variants. Therefore, the upcoming comparative experiment will not include SMA for comparison.

According to Table IX, MSMA's ability to achieve highly competitive best fitness values frequently converges to zero or near-zero fitness on unimodal functions such as F1 and F2, emphasizing its exceptional exploitation efficiency. Moreover, on multimodal functions like F13, MSMA exhibits worthy exploration capabilities, navigating intricate landscapes and converging to optimal solutions. These findings collectively highlight MSMA as an effective metaheuristic algorithm with the potential for solving real-world problems in various domains.

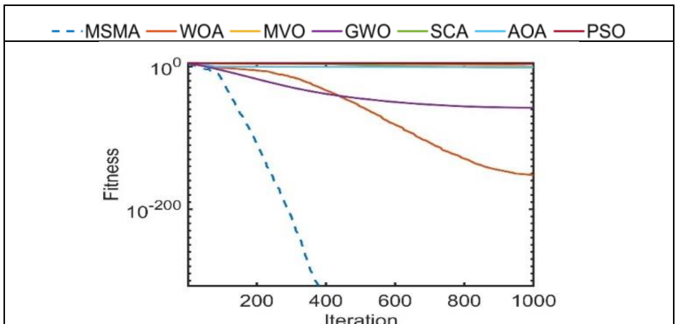
TABLE VIII. THE SETTING OF ALGORITHMS' PARAMETERS

| Method | Population size | The maximum number of iterations | Other parameters |
|---|---|---|---|
| *MSMA (Proposed)* | 30 | $10^3$ | $z$ = 0.03, $\mu$ =0.5, $\epsilon$ =0.02, $\alpha$ = 5 and $\beta$ =3/2 |
| *(WOA)* [5] | 30 | $10^3$ | $a1 = 2-0$; $a2 = -1--2$; $b = 1$ |
| *(MVO)* [47] | 30 | $10^3$ | Wormhole Existence Probability WEPMax = 1; WEPMin = 0.2; |
| *(GWO)* [48] | 30 | $10^3$ | $a$: 2– 1 |
| *(SCA)* [49] | 30 | $10^3$ | $a = 2$ |
| *AOA* [50] | 30 | $10^3$ | $\mu$ =0.5 and $\alpha$ = 5 |
| *(PSO)* [4] | 30 | $10^3$ | c1 = 2.5 – 0.5, c2 = 0.5–2.5, w = 0.9–0.5. |

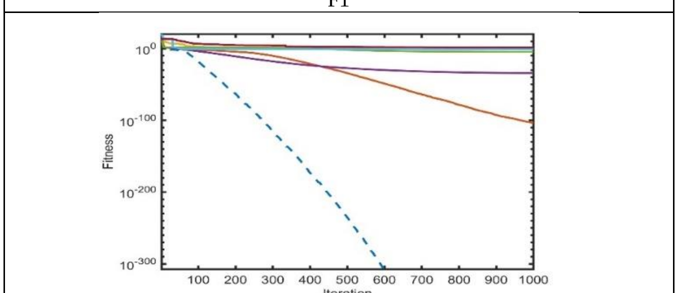
TABLE IX. COMPARISON MSMA WITH CONVENTIONAL ALGORITHMS

| Function | Fitness | Algorithm | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | | ***MSMA*** | ***WOA*** | ***MO*** | ***GWO*** | ***SCA*** | ***AOA*** | ***PSO*** |
| ***F1*** | Mean | **0** | 6.9E-153 | 0.313323 | 1.81E-58 | 0.078575 | 2.23E-32 | 5866.422 |
| | Std | 0 | 3.3E-152 | 0.09829 | 7.72E-58 | 0.23702 | 1.22E-31 | 1311.936 |
| ***F2*** | Mean | **0** | 4E-104 | 0.42004 | 8.46E-35 | 3.22E-05 | **0** | 48.39758 |
| | Std | 0 | 1.9E-103 | 0.093926 | 1.14E-34 | 6.67E-05 | 0 | 10.43552 |
| ***F3*** | Mean | **0** | 23143.08 | 44.96597 | 1.64E-15 | 3319.778 | 0.004737 | 24646.9 |
| | Std | 0 | 9726.587 | 20.69163 | 6.89E-15 | 2581.694 | 0.009056 | 5292.921 |
| ***F4*** | Mean | **0** | 41.81578 | 0.899134 | 2.87E-14 | 21.89053 | 0.024824 | 37.87362 |
| | Std | 0 | 31.17343 | 0.283396 | 1.05E-13 | 12.17732 | 0.020985 | 3.295181 |
| ***F5*** | Mean | **0.85445** | 27.21463 | 204.3283 | 26.77613 | 184.4234 | 28.26249 | 2735175 |
| | Std | 4.564721 | 0.553541 | 224.4148 | 0.829596 | 300.5298 | 0.40652 | 1167378 |
| ***F6*** | Mean | **5.59E-06** | 0.093981 | 0.315247 | 0.555492 | 4.81354 | 2.778531 | 6032.94 |
| | Std | 5.29E-06 | 0.116612 | 0.079513 | 0.329167 | 0.783288 | 0.27915 | 1379.102 |
| ***F7*** | Mean | 9.38E-05 | 0.002088 | 0.021331 | 0.000915 | 0.035331 | **3.45E-05** | 2.15723 |
| | Std | 8.94E-05 | 0.00177 | 0.008588 | 0.000562 | 0.030723 | 3.86E-05 | 0.798218 |
| ***F8*** | Mean | **-12569.5** | -11210.8 | -7972.48 | -6055.25 | -3921.06 | -5738.88 | -3277.07 |
| | Std | 0.00015 | 1489.677 | 621.0526 | 956.9094 | 261.2019 | 492.3883 | 419.031 |
| ***F9*** | Mean | **0** | 1.89E-15 | 109.1013 | 0.848748 | 25.83291 | **0** | 267.1143 |
| | Std | 0 | 1.04E-14 | 30.77091 | 2.414291 | 37.67391 | 0 | 19.37847 |
| ***F10*** | Mean | **8.88E-16** | 4.56E-15 | 1.278301 | 1.6E-14 | 13.02186 | **8.88E-16** | 13.72874 |
| | Std | 0 | 2.38E-15 | 0.96858 | 2.79E-15 | 8.510519 | 0 | 0.871003 |
| ***F11*** | Mean | **0** | **0** | 0.568299 | 0.001077 | 0.177192 | 0.090559 | 57.97914 |
| | Std | 0 | 0 | 0.091516 | 0.003314 | 0.204652 | 0.067312 | 11.14415 |
| ***F12*** | Mean | **1.19E-06** | 0.005829 | 1.557851 | 0.040531 | 2.313704 | 0.414541 | 653080.9 |
| | Std | 6.37E-07 | 0.0047 | 1.298217 | 0.019803 | 3.730789 | 0.04977 | 636192.7 |
| ***F13*** | Mean | **3.09E-05** | 0.192608 | 0.073073 | 0.521447 | 12.55231 | 2.78815 | 5119619 |
| | Std | 0.00011 | 0.115827 | 0.037876 | 0.208239 | 35.92383 | 0.095486 | 3507365 |

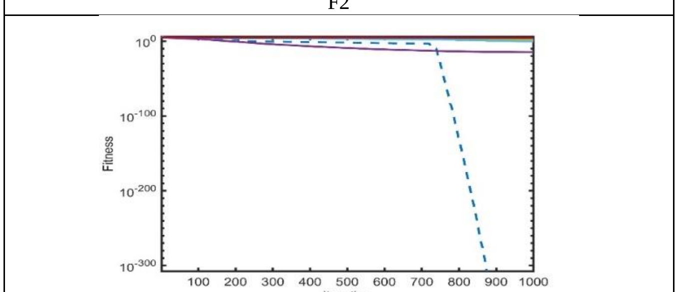
*b) Convergence curve:* In this section, Fig. 6 shows the convergence curves of MSMA compared to WOA [5], MVO [47], GWO [48], SCA [49], AOA [50], and PSO [4]. Fig. 6 displays convergence curves derived from the average best objective function value achieved over 30 runs, as detailed in Table IX. The x-axis represents 1000 iterations, while the y-axis represents the maximum score achieved. The results show that MSAM is superior to its competitors in most of the unimodal functions (1–7), and this reflects its high ability in the exploitation phase. Furthermore, MSAM's exploratory capabilities were showcased in multimodal functions (8–13), highlighting its superiority in all functions. Overall, it demonstrates that the convergence of MSMA is significantly superior to that of other algorithms across most functions. This is due to VSRR, which intelligently switches between algorithms in MSMA to take advantage of its exploitation and exploration capabilities.



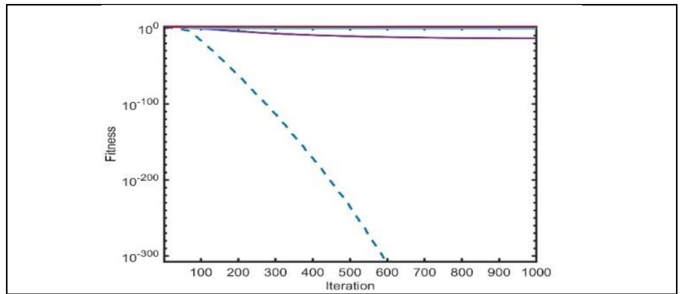

F1



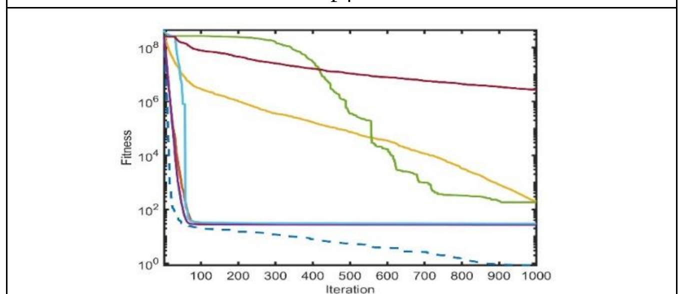

F2



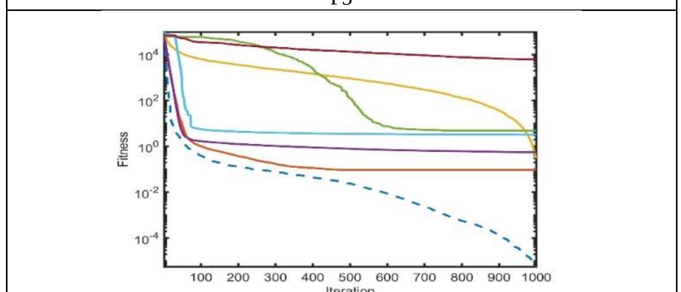

F3



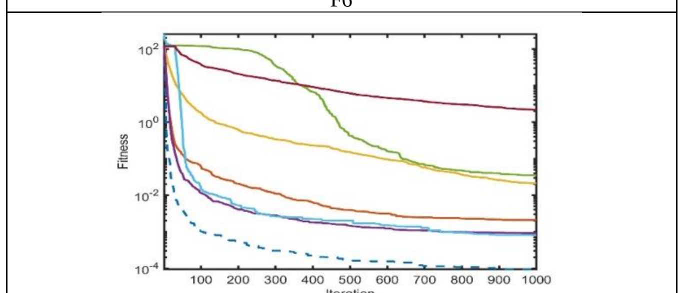

F4



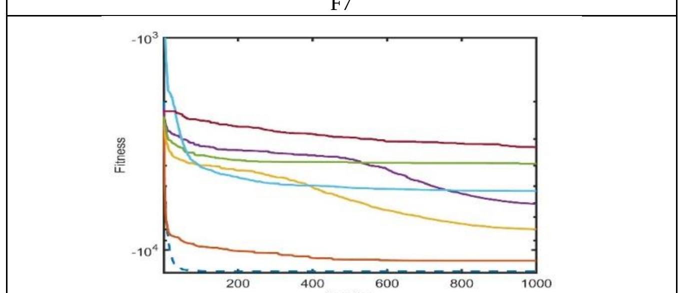

F5



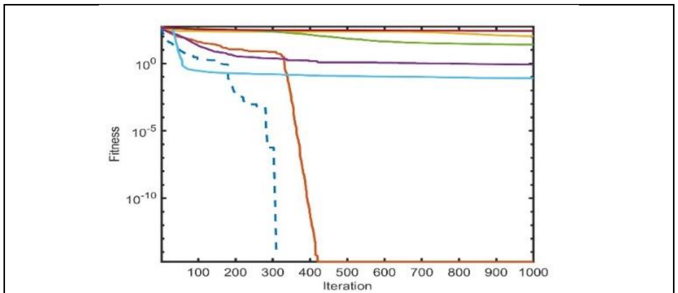

F6



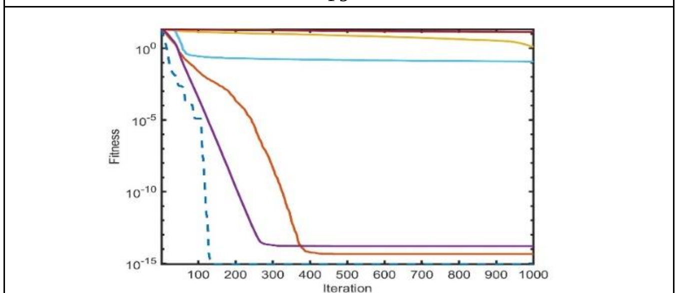

F7



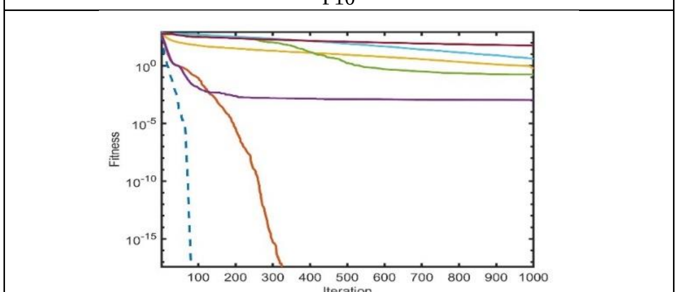

F8



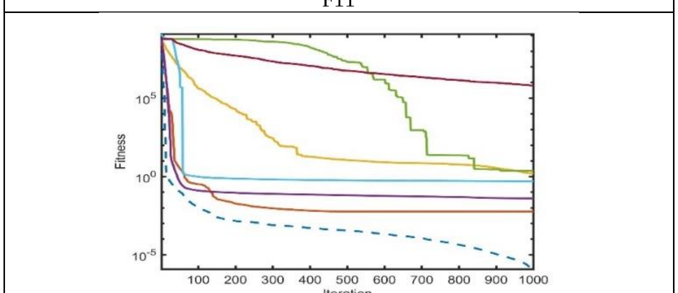

F9



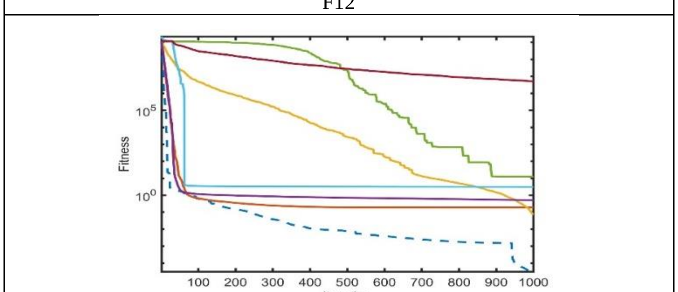

F10




F11




F12




F13

Fig. 6. The convergence curves for unimodal and multimodal functions.

### B. *Experimental Results on Constraints Problems (Engineering Design Problems)*

The performance of MSMA was assessed by applying the method to solve various engineering design problems. These included a cantilever beam problem, a welded beam design problem, a pressure vessel problem, a compression coil spring design problem, a multiple disc clutch brake problem, a speed reducer problem, and a gear train design problem. The mathematical formulas relating to these problems are provided in "Appendix A". These validations evaluated the effectiveness and suitability of MSMA in tackling different design challenges.

This experiment standardized the parameters for all optimization techniques to ensure a fair comparison. The maximum number of function iterations was set to 10,000, and the population size was set to 30. For statistically reliable results, each method underwent 30 runs independently.

*1) Performance analysis:* This section compares the efficacy of MSMA to that of SMA [14] and SMA variants ($S^2$SMA [25], LSMA [36], AOSMA [24], and ESMA [26]). Table X shows performance metrics for MSMA and the other algorithms on Engineering design problems, including the mean, and the standard deviation.

Based on the obtained results, in the problem of Cantilever Beam analysis, MSMA, boasting a mean of 13.36523309, clearly outperforms its counterparts. ESMA, LSMA, and SMA yield closely clustered means of 13.36532, 13.36531551, and 13.36536, respectively, while AOSMA displays a slightly elevated average. This highlights MSMA's superior effectiveness. Likewise, in Welded Beam problem assessments, MSMA's mean of 1.724885178 is notably superior to its peers. ESMA closely trails with a mean of 1.724979, while other algorithms register marginally higher averages, underscoring the unmistakable dominance of MSMA in this context. Transitioning to the Pressure Vessel problem, MSMA stands out as the top-performing algorithm. Its mean of 6766.643344 significantly outperforms SMA, ESMA, and LSMA, all of which yield notably higher means. This underscores MSMA's exceptional suitability for this specific problem. In the Compression Coil Spring Design problem, MSMA's mean of 0.01284604 distinctly outshines alternative algorithms, which yield significantly higher averages. This glaring disparity underscores the exceptional performance of MSMA in this scenario.

Moreover, in the Multiple Disk Clutch Brake problem, MSMA, SMA, and ESMA exhibit closely aligned means, with MSMA marginally leading. While LSMA and AOSMA register slightly higher values, MSMA's marginal lead implies superior efficacy for this problem. In the Speed Reducer problem, MSMA, ESMA, and AOSMA stand out with proximate mean values, with MSMA in the lead. In contrast, SMA and LSMA yield substantially higher averages, reinforcing the notable performance of MSMA. Lastly, in the Gear Train Design problem, MSMA's mean value of 3.25763E-20 is strikingly lower than those of alternative algorithms, which produce considerably higher means, unequivocally solidifying its unparalleled suitability for this specific function. These results demonstrate MSMA's superior performance across various engineering problems, affirming its pivotal role in optimization endeavours.

TABLE X. STATISTICAL RESULTS OF ENGINEERING DESIGN PROBLEMS

| Function | Fitness | Algorithm | | | | | |
|---|---|---|---|---|---|---|---|
| | | *MSMA* | *$S^2$SMA* | *SMA* | *ESMA* | *LSMA* | *AOSMA* |
| **Cantilever beam** | Mean | **13.36523** | 13.36534 | 13.36536 | 13.36532 | 13.36532 | 13.36526 |
| | Std | 2.23E-05 | 0.0001 | 7.36E-05 | 9.2E-05 | 5.69E-05 | 3.74E-05 |
| **Cantilever beam** | Mean | **1.724885** | 1.725027 | 1.725076 | 1.724979 | 1.725094 | 1.725079 |
| | Std | 7.13E-05 | 0.000174 | 0.000276 | 0.000113 | 0.00021 | 0.000229 |
| **Pressure Vessel** | Mean | **6766.643** | 6861.241 | 78181.51 | 68808.52 | 74919.22 | 37554.12 |
| | Std | 565.1678 | 501.2 | 40226.18 | 44866.76 | 42302.85 | 44912.3 |
| **Compression Coil Spring design** | Mean | **0.012846** | 0.013245 | 50000.01 | 40000.01 | 53333.34 | 56666.67 |
| | Std | 0.000151 | 0.000301 | 50854.76 | 49827.28 | 50741.62 | 50400.69 |
| **Multiple disk clutch brake** | Mean | **0.25977** | 0.259784 | 0.259774 | 0.259774 | 0.259785 | 0.259771 |
| | Std | 3.18E-06 | 1.26E-05 | 2.98E-06 | 4.5E-06 | 1.47E-05 | 1.34E-06 |
| **Speed reducer** | Mean | **2996.351** | 2996.352 | 100000 | 96766.54 | 93533.09 | 96766.54 |
| | Std | 0.009292 | 0.003918 | 0 | 17710.36 | 24610.61 | 17710.36 |
| **Gear train design** | Mean | **3.26E-20** | 2E-14 | 3.74E-14 | 4.35E-14 | 3.13E-14 | 5.45E-15 |
| | Std | 8E-20 | 2.84E-14 | 1.31E-13 | 1.08E-13 | 6.49E-14 | 1.44E-14 |

*2) Statistical analysis:* To statistically evaluate the performance of MSMA and the compared algorithms, including SMA [14] S and SMA variants ($S^2$SMA [25], LSMA [36], AOSMA [24], and ESMA [26]), on various engineering design problems. Calculating the p-value of the Wilcoxon signed-rank test [51]. Each value greater than 0.05 is displayed in bold font, indicating that the difference is not statistically significant. The calculated p-values indicate substantial evidence of differentiation, as shown in Table XI MSMA's p-values are smaller than 0.05 in the majority of cases, indicating significant differences. Notably, the "Pressure Vessel" problem exhibits a relatively large p-value (approximately 0.76) when comparing MSMA to AOSMA, indicating that the difference with AOSMA is not statistically significant. In contrast, for the "Gear train design" problem, the p-values consistently indicate significant differences, indicating that MSMA outperforms all compared algorithms. These results demonstrate the superior performance of MSMA and its potential as an efficient optimization method for complex engineering design problems.

TABLE XI. P-VALUES FOR MSMA VERSUS OTHER COMPETITORS ON ENGINEERING DESIGN PROBLEMS

| Function | MSMA vs. $S^2$SMA | MSMA vs. SMA | MSMA vs. ESMA | MSMA vs. LSMA | MSMA vs. AOSMA |
|---|---|---|---|---|---|
| ***Cantilever beam*** | 7.77255E-09 | 7.38029E-10 | 2.37682E-07 | 2.19474E-08 | 0.000471375 |
| ***Welded Beam*** | 3.64589E-08 | 1.42942E-08 | 1.15665E-07 | 1.10234E-08 | 2.83145E-08 |
| ***Pressure Vessel*** | **0.157975689** | 2.66709E-06 | 0.000244046 | 0.000377215 | **0.761297126** |
| ***Compression Coil Spring design*** | 1.8731E-07 | 0.005708009 | **0.619007153** | 0.031019514 | 0.001643841 |
| ***Multiple disk clutch brake*** | 8.10136E-10 | 3.96477E-08 | 7.69496E-08 | 7.38029E-10 | 7.59915E-07 |
| ***Speed reducer*** | 1.35943E-07 | 1.21178E-12 | 4.21155E-12 | 3.68819E-12 | 8.15959E-12 |
| ***Gear train design*** | 3.01986E-11 | 7.38029E-10 | 3.01986E-11 | 3.01986E-11 | 3.01986E-11 |

*3) Comparison with conventional algorithms:* This section aims to evaluate the performance of MSMA through a comprehensive comparison with six popular metaheuristic algorithms: WOA [5], MVO [47], GWO [48], SCA [49], AOA [50], and PSO [4]. The comparison is conducted across seven distinct engineering design problems to thoroughly assess their capabilities in solving engineering problems. The main parameter settings for each algorithm are outlined in Table VIII.

Beginning with the Cantilever Beam Design Problem, the analysis reveals that MSMA exhibits competitive performance, achieving optimal values for variables (x1 to x5) and an optimal cost of 13.36520828, as demonstrated in Table XII. This outcome underscores the effectiveness of MSMA in addressing structural engineering challenges, where precise optimization is paramount for ensuring structural integrity and efficiency.

Similarly, in the Welded Beam Problem, MSMA demonstrates notable performance with an optimal cost of 1.724852759, as presented in Table XIII, indicating its capability to navigate the complexities inherent in welding design optimization. The results further validate the robustness of MSMA in handling diverse engineering scenarios, where intricate design considerations must be balanced to achieve optimal outcomes.

The Pressure Vessel Problem, as presented in Table XIV, further emphasizes the diversity of MSMA's capabilities. It showcases optimal values for variables and an optimal cost of 5885.332794. It highlights MSMA's adaptability to multifaceted challenges in pressure vessel design optimization, where complex geometrical and operational constraints influence the design space.

In the Compression Coil Spring Design Problem, MSMA continues to demonstrate competitive results, achieving an optimal cost of 0.012665319, as presented in Table XV. This performance highlights the efficacy of MSMA in optimizing mechanical components, where precision in design parameters is crucial for achieving desired spring characteristics and performance metrics.

Table XVI illustrates the optimal values for variables (x1, x2, x3, x4, x5) and their respective optimal costs achieved by various algorithms in the Multiple Disk Clutch Brake scenario. MSMA outperforms competitors by attaining an optimal cost of 0.259768995. In contrast, other algorithms exhibit slightly different values for the variables. It highlights the effectiveness of MSMA in this context.

Similarly, Table XVII provides a comparative analysis for the Speed Reducer Problem, where MSMA excels in achieving an optimal cost of 2996.348166. Competing algorithms, on the other hand, are unable to reach the same degree of accuracy. MSMA's reliability and effectiveness are demonstrated by its ability to handle the complexity of this problem.

In the context of gear train design optimization, Table XVIII highlights the effectiveness of the MSAM algorithm with an optimal cost of 4.29529E-26. Additionally, WOA achieves a noteworthy optimal cost of 0, emphasizing its competitive performance. These findings underscore the capabilities of MSAM and WOA in addressing complex engineering optimization challenges.

TABLE XII. COMPARISON RESULTS OF THE CANTILEVER BEAM DESIGN PROBLEM

| Algorithms | Optimal values for variables | | | | | Optimal cost |
|---|---|---|---|---|---|---|
| | *x1* | *x2* | *x3* | *x4* | *x5* | |
| ***MSMA*** | 6.017085383 | 5.311288687 | 4.488476538 | 3.507273784 | 2.149604785 | **13.36520828** |
| ***WOA*** | 5.700449107 | 5.397613593 | 4.814600886 | 3.522731364 | 2.119436619 | 13.38920896 |
| ***MVO*** | 6.033559535 | 5.30753025 | 4.440377082 | 3.528653236 | 2.165715561 | 13.36529853 |
| ***GWO*** | 6.030295996 | 5.311933104 | 4.485345216 | 3.494561661 | 2.151681834 | 13.36520866 |
| ***SCA*** | 6.256390441 | 6.10830251 | 4.446189485 | 3.086518141 | 2.009635551 | 13.44560967 |
| ***AOA*** | 6.314864496 | 5.699730447 | 3.986353036 | 3.898579909 | 2.105572722 | 13.49454764 |
| ***PSO*** | 5.716182614 | 5.394840612 | 4.923773757 | 3.426754426 | 2.132218051 | 13.39554248 |

TABLE XIII. COMPARISON RESULTS OF THE WELDED BEAM PROBLEM

| Algorithms | Optimal values for variables | | | | Optimal cost |
|---|---|---|---|---|---|
| | *x1* | *x2* | *x3* | *x4* | |
| ***MSMA*** | 0.20572668 | 3.470554907 | 9.036623951 | 0.205729641 | **1.724852759** |
| ***WOA*** | 0.416992489 | 2.031718826 | 6.338092508 | 0.421197407 | 1.757628876 |
| ***MVO*** | 0.204572029 | 3.495334409 | 9.042511959 | 0.205708116 | 1.725770029 |
| ***GWO*** | 0.20548343 | 3.475777453 | 9.036731042 | 0.205741147 | 1.724921656 |
| ***SCA*** | 0.206779645 | 3.346727647 | 9.463618339 | 0.210822386 | 1.74420762 |
| ***AOA*** | 0.208707168 | 3.156291235 | 10 | 0.247529632 | 1.854111628 |
| ***PSO*** | 0.211161682 | 3.416026615 | 8.890902048 | 0.21255867 | 1.733532831 |

TABLE XIV. COMPARISON RESULTS OF THE PRESSURE VESSEL PROBLEM

| Algorithms | Optimal values for variables | | | | Optimal cost |
|---|---|---|---|---|---|
| | *x1* | *x2* | *x3* | *x4* | |
| ***MSMA*** | 1.258828444 | 0.622239552 | 65.22427172 | 10.00413828 | **5885.332794** |
| ***WOA*** | 74.39811474 | 34.47896969 | 46.42166663 | 73.12900591 | 5913.484457 |
| ***MVO*** | 89.65475933 | 74.80944321 | 18.909546 | 166.6995299 | 6432.102507 |
| ***GWO*** | 0.77873673 | 0.385013082 | 40.34876174 | 199.595325 | 5886.112827 |
| ***SCA*** | 0.799649958 | 0.428055843 | 40.71461012 | 200 | 5968.711993 |
| ***AOA*** | 34.09159014 | 87.13884443 | 13.35195248 | 51.93686087 | 9424.698317 |
| ***PSO*** | 36.82855393 | 79.53717197 | 52.42213373 | 69.31768329 | 6155.484164 |

TABLE XV. COMPARISON RESULTS OF THE COMPRESSION COIL SPRING DESIGN PROBLEM

| Algorithms | Optimal values for variables | | | Optimal cost |
|---|---|---|---|---|
| | *x1* | *x2* | *x3* | |
| ***MSMA*** | 0.055584231 | 0.457867024 | 7.138747128 | **0.012665319** |
| ***WOA*** | 0.059352736 | 0.570654818 | 4.793787974 | 0.012672374 |
| ***MVO*** | 0.057411627 | 0.510629027 | 5.858130197 | 0.012702184 |
| ***GWO*** | 0.030415911 | 0.746376689 | 2.882659855 | 0.01266583 |
| ***SCA*** | 0.049565332 | 0.307684113 | 15 | 0.012751116 |
| ***AOA*** | 0.076649751 | 1.3 | 2 | 0.015289034 |
| ***PSO*** | 0.050062779 | 0.315733397 | 14.66722903 | 0.012701516 |

TABLE XVI. MULTIPLE DISK CLUTCH BRAKE

| Algorithms | Optimal values for variables | | | | | Optimal cost |
|---|---|---|---|---|---|---|
| | *x1* | *x2* | *x3* | *x4* | *x5* | |
| ***MSMA*** | 69.9999999 | 90.00000002 | 1.000000004 | 1000 | 2.312782041 | **0.259768995** |
| ***WOA*** | 70 | 90 | 1 | 1000 | 2.312782578 | 0.259769039 |
| ***MVO*** | 70.00153892 | 90.00180212 | 1 | 999.7168915 | 2.313482158 | 0.259781877 |
| ***GWO*** | 69.99852945 | 90 | 1 | 1000 | 2.312864645 | 0.259774817 |
| ***SCA*** | 69.63372395 | 90 | 1 | 1000 | 2.347561762 | 0.260725078 |
| ***AOA*** | 80 | 100.7047151 | 1 | 1000 | 2.32781793 | 0.277270665 |
| ***PSO*** | 69.99879821 | 90 | 1 | 1000 | 2.312844506 | 0.259783161 |

TABLE XVII. COMPARISON RESULTS OF THE SPEED REDUCER PROBLEM

| Algorithms | Optimal values for variables | | | | | | | Optimal cost |
|---|---|---|---|---|---|---|---|---|
| | *x1* | *x2* | *x3* | *x4* | *x5* | *x6* | *x7* | |
| ***MSMA*** | 3.500000002 | 0.7 | 17 | 7.300000013 | 7.80000075 | 3.350214675 | 5.286683234 | **2996.348166** |
| ***WOA*** | 3.5 | 0.7 | 17 | 8.086052026 | 8.061415721 | 3.356439258 | 5.348209953 | 3001.995774 |

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| ***MVO*** | 3.501974802 | 0.7 | 17 | 7.405570651 | 8.070442656 | 3.353599328 | 5.286782167 | 2998.474657 |
| ***GWO*** | 3.500234799 | 0.7 | 17.00022304 | 7.323444697 | 7.801214073 | 3.350567429 | 5.286801157 | 2996.877194 |
| ***SCA*** | 3.551329239 | 0.7 | 17 | 7.781838931 | 8.3 | 3.422479892 | 5.310732943 | 3034.002185 |
| ***AOA*** | 3.6 | 0.7 | 17 | 7.3 | 8.3 | 3.516263029 | 5.294372667 | 3074.222921 |
| **PSO** | 2.626401315 | 0.728946859 | 20.50362653 | 8.249249684 | 8.218561811 | 3.380030355 | 5.348181951 | 2997.389625 |

TABLE XVIII. COMPARISON RESULTS OF THE GEAR TRAIN DESIGN PROBLEM

| **Algorithms** | **Optimal values for variables** | | | | **Optimal cost** |
|---|---|---|---|---|---|
| | ***x1*** | ***x2*** | ***x3*** | ***x4*** | |
| ***MSMA*** | 20.51707896 | 14.28150743 | 12 | 57.89428103 | 4.29529E-26 |
| ***WOA*** | 56.38376759 | 12.21452818 | 33.2686516 | 49.95209432 | **0** |
| ***MVO*** | 18.0025215 | 12.49738546 | 12 | 57.73816517 | 1.03484E-18 |
| ***GWO*** | 50.71522562 | 17.09261159 | 24.15294502 | 56.42037706 | 2.93402E-17 |
| ***SCA*** | 58.2894748 | 40.59354904 | 12 | 57.92398093 | 5.63401E-16 |
| ***AOA*** | 59.97591416 | 12.00002286 | 43.26335633 | 60 | 2.99093E-15 |
| ***PSO*** | 42.42788592 | 30.55243375 | 12 | 59.89162803 | 2.5461E-15 |

### *C. Discussion*

The Merged Slime Mould Algorithm (MSMA) results demonstrate its effectiveness across benchmark functions and engineering design problems. Evaluating 23 continuous benchmark functions from the CEC 2005 revealed that MSMA excels in achieving optimal results, particularly in unimodal functions where exploitation is crucial. Its performance in multimodal functions illustrates robust exploration capabilities, effectively navigating complex landscapes and avoiding local optima.

Comparisons with other Slime Mould Algorithm (SMA) variants and established metaheuristic algorithms like WOA, GWO, and PSO showed that MSMA consistently outperforms its peers. The mean and standard deviation metrics analysis highlight MSMA's ability to frequently achieve optimal or near-optimal fitness values. The convergence curves indicate that MSMA delivers rapid convergence, leveraging Vertical Smart Switching Rules (VSRR) for intelligent algorithm switching, thus enhancing both exploitation and exploration strategies.

MSMA's superiority in engineering design problems is further validated. For instance, the Cantilever Beam problem achieved significantly lower mean values compared to other algorithms. Similar trends were noted in the Welded Beam and Pressure Vessel problems, with statistical significance confirmed through the Wilcoxon signed-rank test. These results underscore MSMA's reliability and efficiency in tackling complex engineering challenges.

The promising outcomes of MSMA open several exciting avenues for future research. Exploring hybridization techniques that combine MSMA with advanced optimization algorithms could further enhance its performance. Additionally, adapting Vertical Smart Switching Rules (VSRR) for dynamic problem landscapes may improve efficiency. Future studies could also validate MSMA through real-world case studies, ensuring its practical applicability across diverse industries. Such explorations would significantly contribute to the optimization field and enhance MSMA's utility in addressing complex challenges, instilling a sense of optimism and hope for its continuous improvement.

## V. CONCLUSION

In conclusion, this paper introduced MSMA as a dynamic hybridization approach engineered to significantly enhance the performance of the traditional SMA in tackling low-dimensional optimization problems compared to other algorithms. The proposed technique merges two existing SMA variants, AOSMA and $S^2$SMA, through the incorporation of embedded Vertical Smart Switching Rules (VSSR). VSSR enables dynamic switching between algorithms based on problem-specific attributes, thereby boosting adaptability and operational efficiency. The MSMA's unique integration strategy eliminates the need for multiple algorithm initializations as well as avoids the need for memory-based switching. Instead, it relies on adaptive and intelligent switching rules to exploit the strengths of both algorithms. This represents a notable advancement compared to previous integrations of SMA.

The proposed MSMA has been fully validated on ten real-world engineering challenges and basic benchmark problems CEC 2005 using statistical and numerical analyses. The experimental results highlight MSMA's superiority over current approaches and demonstrate its potential to provide innovative solutions for complex engineering designs. This study provides additional evidence that MSMA consistently achieves the highest mean fitness values and shows the fastest rates of convergence among the algorithms evaluated, demonstrating its superior performance in addressing engineering design problems. Remarkably, when compared to SMA, MSMA also showed improved computational efficiency, particularly in the Cantilever Beam problem. The Wilcoxon signed-rank test has statistically validated MSMA's outstanding performance in a variety of engineering problems, confirming its superiority and efficacy in resolving complex engineering design problems.

These findings validate the proposed MSMA's superiority over existing techniques, showcasing its potential to provide promising solutions for complex engineering design problems. Future research directions could pivot towards enhancing the VSSR mechanism to further improve MSMA's adaptability and robustness. Moreover, extending the exploration to other problem domains and conducting comparative studies with other state-of-the-art optimization algorithms would yield additional insights, paving the way for further advancements in optimization technology.

ACKNOWLEDGMENT

Conceptualization, S.A.S. and H.S.; Methodology, K.M.A.; Software, K.M.A.; Validation, S.A.S. and H.S.; Formal analysis, K.M.A.; Investigation, K.M.A.; Resources, K.M.A.; Data curation, K.M.A.; Writing – original draft, K.M.A.; Writing – review & editing, K.M.A. and H.S.; Visualization, S.A.S. and

H.S.; Supervision, S.A.S. and H.S.; Project administration, S.A.S. and H.S.; Funding acquisition, S.A.S. All authors have read and agreed to the published version of the manuscript.

## REFERENCES

[1] M. H. Nadimi-Shahraki, S. Taghian, S. Mirjalili, H. Zamani, and A. Bahreininejad, “GGWO: Gaze cues learning-based grey wolf optimizer and its applications for solving engineering problems,” J. Comput. Sci., vol. 61, no. June 2021, p. 101636, 2022.

[2] Y. Duan and X. Yu, “A collaboration-based hybrid GWO-SCA optimizer for engineering optimization problems,” Expert Syst. Appl., vol. 213, no. PB, p. 119017, 2023.

[3] S. Chauhan and G. Vashishtha, “A synergy of an evolutionary algorithm with slime mould algorithm through series and parallel construction for improving global optimization and conventional design problem,” Eng. Appl. Artif. Intell., vol. 118, no. December 2022, p. 105650, Feb. 2023.

[4] J. Kennedy and R. Eberhart, “Particle swarm optimization,” in Proceedings of ICNN’95-international conference on neural networks, 1995, vol. 4, pp. 1942–1948.

[5] S. Mirjalili and A. Lewis, “The Whale Optimization Algorithm,” Adv. Eng. Softw., vol. 95, pp. 51–67, 2016.

[6] A. A. Heidari, S. Mirjalili, H. Faris, I. Aljarah, M. Mafarja, and H. Chen, “Harris hawks optimization: Algorithm and applications,” Futur. Gener. Comput. Syst., vol. 97, pp. 849–872, 2019.

[7] W. Michiels, E. H. L. Aarts, and J. Korst, Theoretical aspects of local search, vol. 13. Springer, 2007.

[8] F. Glover, “Tabu search—part I,” ORSA J. Comput., vol. 1, no. 3, pp. 190–206, 1989.

[9] S. Kirkpatrick, C. D. Gelatt Jr, and M. P. Vecchi, “Optimization by simulated annealing,” Science (80-. )., vol. 220, no. 4598, pp. 671–680, 1983.

[10] D. H. Wolpert and W. G. Macready, “No free lunch theorems for optimization,” IEEE Trans. Evol. Comput., vol. 1, no. 1, pp. 67–82, 1997.

[11] X. Zhong, Z. You, and P. Cheng, “A hybrid optimization algorithm and its application in flight trajectory prediction,” Expert Syst. Appl., vol. 213, no. PB, p. 119082, 2023.

[12] Q. S. Hamad, H. Samma, S. A. Suandi, and J. Mohamad-Saleh, “Q-learning embedded sine cosine algorithm (QLESCA),” Expert Syst. Appl., vol. 193, no. November 2021, p. 116417, 2022.

[13] H. Samma, J. Mohamad-Saleh, S. A. Suandi, and B. Lahasan, “Q-learning-based simulated annealing algorithm for constrained engineering design problems,” Neural Comput. Appl., vol. 32, no. 9, pp. 5147–5161, 2020.

[14] S. Li, H. Chen, M. Wang, A. A. Heidari, and S. Mirjalili, “Slime mould algorithm: A new method for stochastic optimization,” Futur. Gener. Comput. Syst., vol. 111, pp. 300–323, Oct. 2020.

[15] J. M. Abdullah and T. Ahmed, “Fitness Dependent Optimizer: Inspired by the Bee Swarming Reproductive Process,” IEEE Access, vol. 7, pp. 43473–43486, 2019.

[16] V. Hayyolalam and A. A. P. Kazem, “Black widow optimization algorithm: a novel meta-heuristic approach for solving engineering optimization problems,” Eng. Appl. Artif. Intell., vol. 87, p. 103249, 2020.

[17] L. Abualigah, M. Abd Elaziz, P. Sumari, Z. W. Geem, and A. H. Gandomi, “Reptile Search Algorithm (RSA): A nature-inspired meta-heuristic optimizer,” Expert Syst. Appl., vol. 191, p. 116158, 2022.

[18] T. Kundu and H. Garg, “LSMA-TLBO: A hybrid SMA-TLBO algorithm with lévy flight based mutation for numerical optimization and engineering design problems,” Adv. Eng. Softw., vol. 172, no. May, p. 103185, 2022.

[19] M. K. Naik, R. Panda, and A. Abraham, “Normalized square difference based multilevel thresholding technique for multispectral images using leader slime mould algorithm,” J. King Saud Univ. - Comput. Inf. Sci., vol. 34, no. 7, pp. 4524–4536, Jul. 2022.

[20] H. Lin et al., “Adaptive slime mould algorithm for optimal design of photovoltaic models,” Energy Sci. Eng., vol. 10, no. 7, pp. 2035–2064, 2022.

[21] M. H. Hassan, S. Kamel, L. Abualigah, and A. Eid, “Development and application of slime mould algorithm for optimal economic emission dispatch,” Expert Syst. Appl., vol. 182, no. May, p. 115205, 2021.

[22] S. Zhao et al., “Multilevel threshold image segmentation with diffusion association slime mould algorithm and Renyi’s entropy for chronic obstructive pulmonary disease,” Comput. Biol. Med., vol. 134, no. April, p. 104427, 2021.

[23] A. A. Ewees et al., “Improved Slime Mould Algorithm based on Firefly Algorithm for feature selection: A case study on QSAR model,” Eng. Comput., no. 0123456789, 2021.

[24] M. K. Naik, R. Panda, and A. Abraham, “Adaptive opposition slime mould algorithm,” Soft Comput., vol. 25, no. 22, pp. 14297–14313, 2021.

[25] K. M. Alhashash, H. Samma, and S. A. Suandi, “Fine-Tuning of Pre-Trained Deep Face Sketch Models Using Smart Switching Slime Mold Algorithm,” Appl. Sci., vol. 13, no. 8, p. 5102, Apr. 2023.

[26] M. K. Naik, R. Panda, and A. Abraham, “An entropy minimization based multilevel colour thresholding technique for analysis of breast thermograms using equilibrium slime mould algorithm,” Appl. Soft Comput., vol. 113, p. 107955, 2021.

[27] H. Chen, X. Li, S. Li, Y. Zhao, and J. Dong, “Improved Slime Mould Algorithm Hybridizing Chaotic Maps and Differential Evolution Strategy for Global Optimization,” IEEE Access, vol. 10, no. May, pp. 66811–66830, 2022.

[28] A. A. Bhandakkar and L. Mathew, “Merging slime mould with whale optimization algorithm for optimal allocation of hybrid power flow controller in power system,” J. Exp. Theor. Artif. Intell., vol. 35, no. 7, pp. 973–1000, Oct. 2022.

[29] H. Chen, Z. Wang, H. Jia, X. Zhou, and L. Abualigah, “Hybrid Slime Mold and Arithmetic Optimization Algorithm with Random Center Learning and Restart Mutation,” Biomimetics, vol. 8, no. 5, p. 396, 2023.

[30] A. A. Ewees, F. H. Ismail, and A. T. Sahlol, “Gradient-based optimizer improved by Slime Mould Algorithm for global optimization and feature selection for diverse computation problems,” Expert Syst. Appl., vol. 213, no. September 2022, 2023.

[31] A. A. Ewees et al., “Enhanced feature selection technique using slime mould algorithm: a case study on chemical data,” Neural Comput. Appl., vol. 35, no. 4, pp. 3307–3324, 2023.

[32] S. Abid et al., “Development of Slime Mold Optimizer with Application for Tuning Cascaded PD-PI Controller to Enhance Frequency Stability in Power Systems,” Mathematics, vol. 11, no. 8, 2023.

[33] L. Deng and S. Liu, “An enhanced slime mould algorithm based on adaptive grouping technique for global optimization,” Expert Syst. Appl., vol. 222, no. February, p. 119877, 2023.

[34] A. K. Sharma, A. Saxena, and D. K. Palwalia, “Oppositional Slime Mould Algorithm: Development and application for designing demand side management controller,” Expert Syst. Appl., vol. 214, no. January 2022, p. 119002, 2023.

[35] L. Wang, Q. Cao, Z. Zhang, S. Mirjalili, and W. Zhao, “Artificial rabbits optimization: A new bio-inspired meta-heuristic algorithm for solving engineering optimization problems,” Eng. Appl. Artif. Intell., vol. 114, no. April, p. 105082, 2022.

[36] B. S. Yildiz, N. Pholdee, S. Bureerat, A. R. Yildiz, and S. M. Sait, “Enhanced grasshopper optimization algorithm using elite opposition-based learning for solving real-world engineering problems,” Eng. Comput., vol. 38, no. 5, pp. 4207–4219, 2022.

[37] Y. Zhang, S. Du, and Q. Zhang, “Improved Slime Mold Algorithm with Dynamic Quantum Rotation Gate and Opposition-Based Learning for Global Optimization and Engineering Design Problems,” Algorithms, vol. 15, no. 9, p. 317, Sep. 2022.

[38] B. S. Yıldız, S. Kumar, N. Pholdee, S. Bureerat, S. M. Sait, and A. R. Yildiz, “A new chaotic Lévy flight distribution optimization algorithm for solving constrained engineering problems,” Expert Syst., vol. 39, no. 8, 2022.

[39] X. Yang et al., “An adaptive quadratic interpolation and rounding mechanism sine cosine algorithm with application to constrained engineering optimization problems,” Expert Syst. Appl., vol. 213, no. PB, p. 119041, Mar. 2023.

[40] M. Abdel-Basset, R. Mohamed, M. Jameel, and M. Abouhawwash, "Nutcracker optimizer: A novel nature-inspired metaheuristic algorithm for global optimization and engineering design problems," Knowledge-Based Syst., vol. 262, p. 110248, Feb. 2023.

[41] F. S. Gharehchopogh, M. H. Nadimi-Shahraki, S. Barshandeh, B. Abdollahzadeh, and H. Zamani, "CQFFA: A Chaotic Quasi-oppositional Farmland Fertility Algorithm for Solving Engineering Optimization Problems," J. Bionic Eng., vol. 20, no. 1, pp. 158–183, Jan. 2023.

[42] L. Deng and S. Liu, "A multi-strategy improved slime mould algorithm for global optimization and engineering design problems," Comput. Methods Appl. Mech. Eng., vol. 404, p. 115764, 2023.

[43] A. Bala Krishna, S. Saxena, and V. K. Kamboj, hSMA-PS: a novel memetic approach for numerical and engineering design challenges, vol. 38, no. 4. Springer London, 2022.

[44] J. Zhao, Z. M. Gao, and H. F. Chen, "The Simplified Aquila Optimization Algorithm," IEEE Access, vol. 10, pp. 22487–22515, 2022.

[45] Z. M. Gao, J. Zhao, and S. R. Li, "The Improved Slime Mould Algorithm with Cosine Controlling Parameters," J. Phys. Conf. Ser., vol. 1631, no. 1, 2020.

[46] P. N. Suganthan et al., "Problem definitions and evaluation criteria for the CEC 2005 special session on real-parameter optimization," Tech. Report, Nanyang Technol. Univ. Singapore, May 2005 KanGAL Rep. 2005005, IIT Kanpur, India, no. May, pp. 1–50, 2005.

[47] S. Mirjalili, S. M. Mirjalili, and A. Hatamlou, "Multi-Verse Optimizer: a nature-inspired algorithm for global optimization," Neural Comput. Appl., vol. 27, no. 2, pp. 495–513, Feb. 2016.

[48] S. Mirjalili, S. M. Mirjalili, and A. Lewis, "Grey Wolf Optimizer," Adv. Eng. Softw., vol. 69, pp. 46–61, Mar. 2014.

[49] S. Mirjalili, "SCA: A Sine Cosine Algorithm for solving optimization problems," Knowledge-Based Syst., vol. 96, pp. 120–133, Mar. 2016.

[50] L. Abualigah, A. Diabat, S. Mirjalili, M. Abd Elaziz, and A. H. Gandomi, "The Arithmetic Optimization Algorithm," Comput. Methods Appl. Mech. Eng., vol. 376, p. 113609, 2021.

[51] S. García, A. Fernández, J. Luengo, and F. Herrera, "Advanced nonparametric tests for multiple comparisons in the design of experiments in computational intelligence and data mining: Experimental analysis of power," Inf. Sci. (Ny)., vol. 180, no. 10, pp. 2044–2064, 2010.

## APPENDIX A. ENGINEERING DESIGN PROBLEMS

*A. Cantilever structure problem*

*Minimize* $f(x) = 0.6224(x_1 + x_2 + x_3 + x_4 + x_5)$

*subject to,* $g(x) = \frac{61}{x_1^3} + \frac{37}{x_2^3} + \frac{19}{x_3^3} + \frac{7}{x_4^3} + \frac{1}{x_5^3} - 1 \leq 0$

*Variable ranges:* $0.01 \leq x_1, x_2, x_3, x_4, x_5 \leq 100$

*B. The welded beam design problem*

*Minimize* $f(\vec{x}) = 1.10471x_1^2x_2 + 0.04811x_3x_4(14.0 + x_2)$

*subject to,*

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

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

$g_3(\vec{x}) = \delta(\vec{x}) - \delta_{max}$

$g_4(\vec{x}) = x_1 - x_4 \leq 0$

$g_5(\vec{x}) = P - P_c(x) \leq 0$

$g_6(\vec{x}) = 0.125 - x_1 \leq 0$

$g_7(\vec{x}) = 0.1047x_1^2 + 0.04811x_3x_4(14.0 + x_2) - 5.0 \leq 0$

*Variable ranges:* $0.1 \leq x_1 \leq 2.0$, $0.1 \leq x_2 \leq 10.0$, $0.1 \leq x_3 \leq 10.0$, $0.1 \leq x_4 \leq 2.0$

*where,*

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

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

$$P_c(x) = \frac{4.103E\sqrt{\frac{x_3^2x_4^6}{36}}}{L^2}\left(1 - \frac{x_3}{2L}\sqrt{\frac{E}{4G}}\right), P = 6000lb, L = 14in, E = 30 \times 10^{06}psi$$

$$G = 12 \times 10^{06}psi. \tau_{max} = 136000psi, \sigma(x) = 30000psi, \delta_{max} = 0.25in$$

*C. Pressure Vessel problem*

*Minimize* $f(\vec{x}) = 0.6224x_1x_3x_4 + 1.7781x_2x_3^2 + 3.1661x_1^2x_4 + 19.84x_1^2x_3$

*Subject to,*

$g_1(\vec{x}) = -x_1 + 0.0193x_3 \leq 0,$

$g_2(\vec{x}) = -x_3 + 0.00954x_3 \leq 0,$

$g_3(\vec{x}) = -\pi x_3^2x_4 - \frac{4}{3}\pi x_3^3 + 1{,}296{,}000 \leq 0,$

$g_4(\vec{x}) = x_4 - 240 \leq 0,$

*Variable ranges:* $0 \leq x_1 \leq 99$, $0 \leq x_2 \leq 99, 10 \leq x_3 \leq 200, 10 \leq x_4 \leq 200$

*D. Compression Coil Spring design problem*

*Minimize* $f(\vec{x}) = (x_3 + 2)x_2x_1^2$

*subject to,*

$g_1(\vec{x}) = 1 - \frac{x_2^3x_3}{71785x_1^4} \leq 0$

$g_2(\vec{x}) = \frac{4x_2^2 - x_1x_2}{12566(x_1^3x_2 - x_1^4)} + \frac{1}{5108x_1^2} - 1 \leq 0$

$g_3(\vec{x}) = 1 - \frac{140.45x_1}{x_2^2x_3} \leq 0$

$g_4(\vec{x}) = \frac{x_1 + x_2}{1.5} - 1 \leq 0$

*Variable ranges:* $0.05 \leq x_1 \leq 2.0$, $0.25 \leq x_2 \leq 1.3$, $2 \leq x_3 \leq 15.0$

*E. Multiple disk clutch brake problem*

*Minimize* $f(\vec{x}) = \pi(x_2^2 - x_1^2)x_3(x_5 + 1)\rho,$

*Subject to,*

$g_1(x) = x_2 - x_1 - \Delta R \geq 0$

$g_2(x) = L_{max} - (x_5 + 1)(x_3 + \delta) \geq 0$

$g(x) = P_{max} - P_{rz} \geq 0$

$g(x) = P_{max} * Vsr_{max} - P_{rz} * Vsr \geq 0,$

$g_5(x) = Vsr_{max} - Vsr \geq 0,$

$g_6(x) = T_{max} - T \geq 0,$

$g_7(x) = M_h - sM_s \geq 0,$

$g_8(x) = T \geq 0,$

*Variable ranges:* $60 \leq x_1 \leq 80, 90 \leq x_2 \leq 110, 1 \leq x_3 \leq 3, 0 \leq x_4 \leq 1000, 2 \leq x_5 \leq 9, i = 1,2,3,4,5.$

*where,*

$$M_h = \frac{2}{3}\mu x_4x_5\frac{x_2^3 - x_1^3}{x_2^2 - x_1^2}N.mm, W = \frac{\pi n}{30}rad/s, A = \pi(x_2^2 - x_1^2)mm^2$$

$$P_{rz} = \frac{x_4}{A}N/mm^2, Vsr = \frac{PiR_{sr}n}{30}mm/s, R_{sr} = \frac{2(x_2^3 - x_1^3)}{3(x_2^2x_1^2)}mm$$

$$\Delta R = 20mm, L_{max} = 30mm, \mu = 0.6, P_{max} = 1MP_a, p = 0.0000078\frac{kg}{mm^3}, Vsr_{max} = 10\frac{m}{s},$$

$$\delta = 0.5mm, s = 1.5, T_{max} = 15s, n = 250rpm, I_z = 55Kg.m^2, M_s = 40Nm, Mf = 3Nm$$

*F. Speed reducer problem.*

*Minimize* $f(\vec{x}) = 0.7854x_1x_2^2(3.3333x_3^2 + 14.9334x_3 - 43.0934) - 1.508x_1(x_6^2 + x_7^2) + 7.4777(x_6^3 + x_7^3) + 0.7854(x_4x_6^2 + x_5x_7^2)$

*Subject to,*

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

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

$g_3(\vec{x}) = \frac{1.93x_4^3}{x_2x_3x_6^4} - 1 \leq 0,$

$g_4(\vec{x}) = \frac{1.93x_5^3}{x_2x_3x_7^4} - 1 \leq 0,$

$$g_5(\vec{x}) = \frac{\sqrt{(\frac{745x_4}{x_2x_3})^2 + 16.9 \times 10^6}}{110x_6^3} - 1 \leq 0$$

$$g_6(\vec{x}) = \frac{\sqrt{(\frac{745x_5}{x_2x_3})^2 + 157.5 \times 10^6}}{85x_7^3} - 1 \leq 0$$

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

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

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

$$g_{10}(\vec{x}) = \frac{1.5x_6 + 1.9}{12x_2} - 1 \leq 0,$$

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

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

*Variable ranges:* $2.6 \leq x_1 \leq 3.6$, $0.7 \leq x_2 \leq 0.8$, $17 \leq x_3 \leq 28$, $7.3 \leq x_4 \leq 8.3$, $7.8 \leq x_5 \leq 8.3$, $2.9 \leq x_6 \leq 3.9, 5.5 \leq x_7 \leq 5$

### G. Gear train engineering design problem

*Minimize* $f(\vec{x}) = \left(\frac{1}{6.931} - \frac{x_1x_2}{x_3x_4}\right)^2$

*Variable ranges:* $12 \leq x_1, x_2, x_3, x_4 \leq 60$